NTNU

Det skapende universitet

En genetisk algoritme for navigering i arktiske regioner.

Thomas Marstrander

Master i datateknikk

Innlevert: juni 2014

Hovedveileder: Anders Kofod-Petersen, IDI

Norges teknisk-naturvitenskapelige universitet

Institutt for datateknikk og informasjonsvitenskap

Thomas Marstrander

A genetic algorithm for weather routing in Arctic regions.

Master Thesis, Spring 2014

Artificial Intelligence Group
Department of Computer and Information Science
Faculty of Information Technology, Mathematics and Electrical Engineering

NTNU

Abstract

Arctic regions presents great risks for navigators, because of high speed and rapid changing winds. High speed winds can cause damages and ultimately loss of vessel and personnel. The aim of this thesis is to use an artificial intelligence method to find the least hazardous route between two geographical points. The thesis will take into consideration the wind and the effects of wind on a modeled vessel, statistically representative of fishing vessels in Norway, 2013. A genetic algorithm is applied to the multi-objective path-finding problem and Blendermann's method is used for finding wind loads on the vessel. Findings are compared to traditional ways of navigation. The results indicates that the genetic algorithm is a good method for optimizing multi-objective path-planning in maritime environments.

Sammendrag

Arktiske områder innebærer store trusler for navigering, grunnet kraftige og raskt skiftende vinder. Kraftige vinder kan forårsake skader på personell og fartøy. Målet med tesen er å bruke en kunstig intelligens metode for å finne den tryggeste veien mellom to geografiske koordinater. Tesen analyserer vind og den effekten vind har på en gitt skipsmodell, som er representativ for fiskebåter i Norge i 2013. En genetisk algoritme blir brukt for å løse problemet og Blendermanns metode er brukt for å finne den effekten vind har på fartøyet. Resultater fra eksperimentering med programmet er sammenlignet med mer tradisjonelle metoder for navigering. Resultatene fra tesen indikerer at den genetiske algoritmen er en god metode for å optimalisere en rute i maritime miljø, der flere hensyn kan være i konflikt med hverandre.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) as part of the requirements for fulfilling the degree of Master of Science.

This thesis has been performed for the Department of Computer and Information Science, NTNU, Trondheim, with Anders Kofod-Petersen as supervisor.

Thomas Marstrander
Trondheim, June 26, 2014

Acknowledgements

I would like to thank the whole staff at SINTEF Nord, the managing director Jørn Eldby and co-supervisor Ståle Walderhaug for including me in a great work environment in Tromsø, and Karl Gunnar Aarsæther for valuable help on maritime vessel stability.

I would also like to extend thanks to "Meteorologisk Institutt" and Gunnar Noer for taking interest in my project and providing me with wind data.

Finally, I would like to thank my supervisor at the Department of Computer and Information Science, Anders Kofod-Petersen, for giving me the opportunity to take part in this interesting project and guiding me through it.

Abbreviations

GA Genetic Algorithm

GRT Gross Registered Tonnage

HIRLAM High Resolution Limited Area Model

IMO International Maritime Organization

MEWRA Multi Evolutionary Weather Routeing Algorithm

NETCDF Network Common Data Form

RQ Research Question

SAR Synthetic Aperture Radar

SM Systematic Mapping

UML Unified Modeling Language

Symbols

$\mathbf{A_L}$ Lateral-plane area of a maritime vessel

B Buoyancy centre of a maritime vessel

CK Rolling moment coefficient for a maritime vessel

CY Side-force moment coefficient for a maritime vessel

G Gravity centre of a maritime vessel

$\mathbf{GM_t}$ Transverse metacentric height of a maritime vessel

$\mathbf{H_M}$ Mean height of a maritime vessel

K Rolling moment of a maritime vessel

$\mathbf{M_t}$ Transverse metacentre of a maritime vessel

R Earth's radius

$\mathbf{S_H}$ Lateral-plane centroid above the waterline of a maritime vessel

Y Side-force moment of a maritime vessel

W Displacement of a maritime vessel

g Gravity

ϵ Wind attack angle in relation to ship

θ Rolling angle of a maritime vessel

ρ_a Air density

ρ_w Water density

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter describes the research method of the thesis. The chapter starts by introducing objective and scope of the thesis, then proceeds with defining a set of research questions that will be addressed. Finally, the research method applied throughout the thesis is described.

1.1 Objective

The objective of this thesis is to present a genetic algorithm for optimizing multi-objective path-planning in arctic regions using wind data. Calculations from the algorithm will be used to estimate the route with focus on user preferences.

1.2 Scope

Risk analysis of the complete navigation process of a maritime vessel is a big task with an overwhelming amount of parameters factoring into it, which makes it necessary to simplify the process accordingly. The thesis focuses on the wind related risk to a maritime vessel by estimating the rolling moment on a fishing vessel model. Other wind moments will not be investigated, neither will risk factors contributed by waves and currents.

Quantification by simulation will be investigated in the sea around Norway during the time of a polar low. Since the polar low is on top of landmasses, and the application of the algorithm on the polar low is the most interesting, landmasses will be ignored in the results. Simulations is run of the maritime vessel navigation path and maneuvering process over time, and will use existing models for calculating rolling moment on the maritime vessel. Standard models for the maritime vessel rolling coefficient and angle of rolling will be applied, together with a multi-objective genetic algorithm for minimizing the rolling moment and length of a path.

The model of navigation will be presented with wind data and geographical data, and will not take into account interfering traffic. The resulting paths are displayed and compared to more traditional methods, comparing the tradeoff between performance of algorithms and desired outcome given specified weights and parameters.

1.3 Research Goals

For this thesis a main goal is defined, with complementing research questions.

Goal *The main goal of this thesis is to develop and study how the genetic algorithm can improve on traditional methods for path planning in maritime environments, with focus on wind data.*

The goal is defined from a motivation of combining artificial intelligence and maritime path-planning, with focus on safety. The main goal is complemented by the following research questions, in order to address the various aspects of developing and testing such an algorithm:

Research question 1 *Can a genetic algorithm find an optimal path in a maritime environment ?*

The first research question is defined to investigate if the genetic algorithm approach is applicable to the maritime environment. The research question will be addressed by closely investigating what challenges exists in a maritime environment, and how the GA can deal with these challenges, together with an analysis of how the genetic algorithm evolves for a specific experiment.

Research question 2 *How will the genetic algorithm compare to more traditional methods of path-planning?*

The second research question is designed to evaluate the GA against more traditional methods. Existing methods will be investigated, and the final implementation of the GA will be compared against some of them.

Research question 3 *How will the genetic algorithm handle particularly interesting applications, such as navigating through a polar low, and navigating out of a polar low?*

Finally, the third research question deal with the usefulness of the algorithm. Particularly interesting maritime environment scenarios are investigated, and the algorithm will be applied to these scenarios in order to test the utility of the algorithm.

1.4 Research Method

Prior to the thesis a specialization project was conducted, which included a literature study using the systematic mapping(SM) method. SM uses thematic analysis to find patterns and map the literature found. The main findings from the specialization project was a gap in artificial intelligence used in maritime path-planning. There was found no literature on path-planning in relation to the special environment in arctic regions, such as the heavy wind in a polar lows. The background theory and design builds on the work done in the specialization project, and the identified challenges in maritime path-planning.

This thesis implements the genetic algorithm using ideas for design from earlier work. The genetic algorithm is implemented iteratively in an agile environment, while continuously conducting experiment. The latest version should always be a working system.

1.4.1 General Research Approach

The general research approach is conducted in accordance to the quantitative research methodology, the following points are presented in succession throughout the thesis:

1. Presentation of models, theories and hypotheses – in chapter 1 and 3.
2. Development and adaption of methods for measurements of experiments – in chapter 4.
3. Experimental control and manipulation of algorithm variables – in chapter 6.
4. Collection of empirical simulation data – in chapter 6.
5. Modelling and analysis of the collected data – in chapter 6.

1.4.2 Quantification of Simulations

Quantification of risk by simulations is an established analysis method. The main results from the thesis will be through analysis of simulations. The thesis will highlight observations of how the algorithm solutions are found, how it performs, and results in comparison to more traditional navigation methods.

Chapter 2

Background and Motivation

This thesis builds on the results of the authors' specialization project in computer science, conducted in the fall of 2013. The purpose of this chapter is to provide the necessary background and motivation for this project, and to put the work done in this thesis in a wider context and in relation to the work of others. The chapter starts off with a presentation of motivation and background for the thesis, it then proceeds to present the main findings from the specialization project and similar research that has been conducted.

2.1 Motivation

A meeting with representatives from "Meteorologisk Institutt"[1], "Barentswatch"[2], "Yr"[3], "SINTEF Nord"[4] and "Kystverket"[5] was held at Kystverket in Tromsø the 16. October 2013. There was identified an interest for a maritime vessel path-planning tool that could help ship owners navigate safely in the extreme environment that the Arctic regions present.

In Hollnagel [1996] it is identified that a root cause of up to 90% of accidents is attributed to human elements. In 1983 IMO adopted resolution A.528(13), Recommendation on Weather Routeing, which recognize that routeing advice such as "optimum routes" has proved to benefit ship operations and safety for crew and cargo, and recommends the usage of weather routeing. This is why there is a need for computer calculated algorithms, which can aid in the process

[1] http://www.met.no/
[2] http://www.barentswatch.no/
[3] http://www.yr.no/
[4] http://www.sintef.no/SINTEF-Nord-AS/
[5] http://www.kystverket.no/

of path-planning and finding improvements upon your own route.

One of the biggest dangers to maritime vessel, on-board equipment and crew are when ships rolls too much from side to side. High rolling angles can result in capsizing, and loss of equipment, vessel and ultimately personnel. Rolling moment is highly contributed to by wind force on the side of the ship.

In light of the meeting a literature review was conducted to map out the state-of-the-art of maritime navigation, using artificial intelligence methods, in Arctic regions. The research method used in the specialization project was a thematic analysis of the literature found using systematic mapping, to get an overview of existing literature and identifying research gaps. Systematic mapping is structured from a user specified set of search strings applied to a set of databases. The resulting literature is manually filtered iteratively with increasingly strict constraints. The final set of literature were used as a basis to propose the initial design for the multi-objective genetic algorithm used in this thesis.

2.2 Best path

In the past, best paths has been associated with the shortest path. The definition has since evolved and is now extended to consider optimizing a given set of constraints that is important for traversing the path, e.g. minimizing risk for vessel and crew, least amount of energy consumed and a minimal amount of time expended.

Searching for optimal solutions to problems is a common and highly valued task both in personal life as well as work environment. People use optimization without even thinking about it to schedule weekly activities. It is also essential in advanced science and technology, economics and business. Global optimization is the quest to find the best solution to a problem.

2.3 Path-finding Algorithms

Path-finding algorithms can usually be divided into two categories: pregenerative and reactive. In pregenerative algorithms the planning is done before the object starts traveling, and is without course corrections, as in Chen et al. [1995], while in reactive algorithms the path is found by the vehicle as it proceeds through the environment [Kamon and Rivlin, 1995]. This thesis will focus on the pregenerative approach.

There exists many methods for finding optimal paths. In Garau et al. [2005] an application of the A* algorithm is proposed as a search algorithm for path planning in the ocean. Energy optimal paths are calculated in a simulated environment. The most important drawback of this paper is that the ocean currents are assumed to be static, wind forces are also not taken into account. An alternative path-planning based on potential fields are described in Barraquand et al. [1991] and Kwon et al. [2005]. This method works well for avoiding obstacles, but is susceptible to getting stuck at local minima. A comparison between A* search, rapidly-exploring random tree(RTT) and distance transforms is presented in Jarvis [2006]. Artificial intelligence methods such as case-based reasoning have also been used in path-planning problems [Vasudevan and Ganesan, 1996], which has the benefit of reusing solutions and minimize computational redundancy. Alvarez et al. [2004] provides a genetic algorithm for path planning in strong ocean currents, and can optimize the route based on several conflicting constraints. It is shown through rigorous comparisons, that their genetic algorithm produces significantly superior trajectories than an implementation of particle swarm optimization in Roberge et al. [2013].

Strong results from genetic algorithms in combination with path-planning problems, together with the author's motivation for using the genetic algorithm, resulted in genetic algorithm being the choice of method for solving the path-planning problem in this thesis.

Genetic algorithm uses principles of the Darwinian approach to natural selection, and is part of a group of algorithms called evolutionary computing. Evolutionary Computing was made known by Rechenberg [1965] and Holland [1975] by exploiting the mechanics of natural evolution. Today it is used in diverse artificial intelligence fields such as system optimization, hardware design, computer-assisted manufacturing and robotics, and more described in Floreano and Mattiussi [2008]. Evolutionary algorithms has the benefit of being able to design patterns and solutions that are hard for humans to find. A good example is the evolved antenna at NASA [Hornby et al., 2006].

In genetic algorithms an initial population of paths are improved iteratively by genetic operators. Unlike dynamic programming the computational complexity of genetic algorithms increase linearly with the solution space [Petkovi, 2011]. The drawback of genetic algorithms is that they can not guarantee an optimal solution in finite time.

2.4 Maritime Path-Planning Using Advanced Computations

As we have seen from the last section there exists many ways of dealing with the path-planning problem, and they are very general problems. They can be used in diverse domains, e.g. for aerial vehicles, ships, underwater vehicles, ground robots and particles, or for routeing data packets in a network.

The primary goal of ship routeing is to reduce a voyage cost in various ways.

From ancient times a captain has been in charge of selecting the best course using experience. Maritime vessel weather routeing tries to find an optimum path for ocean routes based on weather forecast, sea conditions and ship characteristics. Finding an optimum path means to focus on important factors, e.g. maximize crew comfort and safety on the voyage, minimize fuel consumption, time and distance used. The maritime navigation problem in Arctic regions is a multi-objective optimization problem, which means finding a compromise between these features that usually are in conflict with each other.

Some of the first work done in complex maritime navigation was done using the isochrone method proposed by James [1957]. This method recursively defines time-fronts, finds the best solution for each time front and expands upon these. Several variations of the isochrone method has been proposed since this.

Kobayashi et al. [2011] uses Powell's method to find an optimal route for fuel expressed by a bezier curve and calculates the wind coefficients using Fujiwara's method. Powell's method is not very well suited for the problem described in this thesis because it must be a real-valued function of a fixed number of real-valued inputs.

The evolutionary approach has been successfully applied in the maritime navigation domain. Several implementations of genetic algorithms in optimization of maritime problem exists, e.g. anti-collision avoidance [Ito et al., 1999], evolution of controllers for autonomous vessels [Manley, 2008], traffic organization in terminals [Smierzchalski, 1999] and cargo stowage problem [Dubrovsky et al., 2002].

A multi-objective version of the evolutionary algorithm, called MEWRA, for navigation has been applied in Szlapczynska and Smierzchalski [2009], which was designed specifically for a static model hybrid-propulsion ship and execution was close to 20 minutes.

2.5 Challenges In Maritime Path-Planning

From the specialization project a set of challenges in relation to path-planning in maritime environment was identified. The challenges can be divided into two groups, static and dynamic challenges. Static challenges are objects or constraints that remains the same throughout traversing the path.
Static challenges:

- Canals
- Shallow waters
- Land masses

- Traffic restricted zones
- Structures, for instance oil rigs or bridges.

Dynamic challenges are objects or fields that change over time:

- Other vessels
- Wind
- Waves
- Currents
- Ice

On a spherical surface, such as the earth, the shortest path between two coordinates is found by traversing the Great Circle. However if there is storm in the middle of these coordinates, traveling around the storm can save time and reduce safety risks.

In Krata [2008] it is identified that the first feature influencing the capsizing rate of a vessel is the size of the vessel. Smaller vessels has a higher chance of capsizing, as illustrated in figure 2.1.

2.6 Existing Software

A simple google search for "weather routeing software" provided an overview of the weather routeing software that exists today. Following is a list of the software that was found and what kind of method they implement for finding the optimal route, if specified by the provider:

- QtVlm: isochrone method
- MaxSea: isochrone method

These two softwares and variations of them were the prevailing weather routeing programs that was found. Several monitoring and onboard-systems were also found, however these have different purposes than weather routeing. Both listed programs use isochrone method. The isochrone method is an old and outdated way of searching through the solution space, and is prone to "isochronic loops" as explained in James [1957]. Drawbacks include having to search through the whole search space for each time front and only allowing for single-objective optimization.

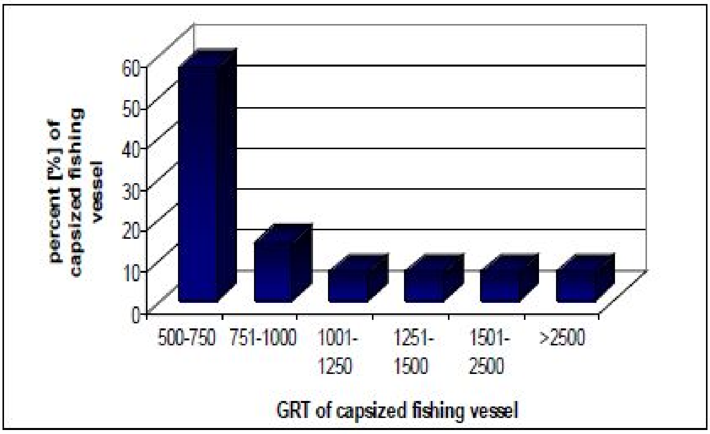


Figure 2.1: Graphs showing the percentage of capsized fishing vessels, grouped by Gross Registered Tonnage(GRT). The smallest fishing vessels has over 57% of the statistically recorded capsizing. The figure is reprinted from Krata [2008]

2.7 Refined Problem Description and Scope

The broad motivation behind this thesis is the need for better safety for maritime vessels. Knowing how wind will affect a maritime vessel at a certain course requires extensive calculations, education and experience. Having a tool that can calculate how much your ship will roll during a trip will let you choose the safest path for maritime vessels in diverse weather.

Earlier work show that genetic algorithms has been successfully applied to path-planning problems in several domains. Latest it was applied to MEWRA using a hybrid propulsion-motor for a bulk carrier.

The major drawbacks of MEWRA includes that it is restricted to a single ship model, a bulk carrier, and shows no results for small scale navigation. The MEWRA does not specify the resolution or format of the wind data that is used. Further this thesis is specifically tested in polar lows, which is a dangerous environmental phenomena in arctic regions. The literature review from the specialization project show that polar lows navigation is not much described in literature.

The search space in a maritime environment, can be overwhelming depending on the desired precision. Because of this the author wanted to try to use a genetic algorithm, which does not have to search through the complete search space in order to find solutions. This thesis takes genetic algorithm in maritime environment a step further by performing extensive experiments in an arctic environment, specifically containing a polar low. The ship model is chosen to represent an especially threatened group of ships, small fishing vessels.

The algorithm focus on safety of the maritime ship model that is chosen in relevance to wind, and the wind forces that act on ship model. Wind force and rolling moment are therefore the main focus of the algorithm, while other dynamic factors, such as waves and currents are not taken into account.

From the systematic mapping and existing software search there was identified a research gap for artificial intelligence algorithms used with maritime path-planning. This resulted in that the path-planning system had to be developed from scratch, using the principles of GA and maritime seakeeping theory.

The outcome of this project includes a working system for computing the safest path given wind data, and how this path compares to more traditional methods of plotting a path. The outcome also includes suggestions for further improvements of the system.

Chapter 3

Theory and Methodology

Predicting stability of maritime vessels and path planning has been and is still a major research field. A variety of models has been used. The model used in this research is Werner Blendermann's method for calculating wind loads on ships, which is based on empirical data from extensive testing in wind tunnels. This chapter contains the meteorological wind data, an overview of the physics acting on a ship, and the method used for implementing the program in this thesis.

3.1 Meteorological Wind Data

The meteorological wind data is essential for calculating the wind forces acting on a maritime vessel, and thus the risk that the windy environment poses to our maritime vessel.

3.1.1 Representation

Meteorological data are represented using array-oriented scientific data. The data provides a set of geographical locations and corresponding wind in longitudinal direction and latitudinal direction. Together these form a wind vector at the specified geographical location. The meteorological data is represented using a standard scientific data format called netcdf which is explained in detail in chapter 5.

3.1.2 Time-Domain and Area

The time-domain of the data spans from 7.january up to and including 9.january 2009, and has measurements for every complete hour, which makes for a total of

72 timesteps

The geographical zone in which there exists wind data, is limited by geographical coordinate 43E 003 in the southern corner and 83N 91'30 in the northern corner, illustrated in figure 3.1.

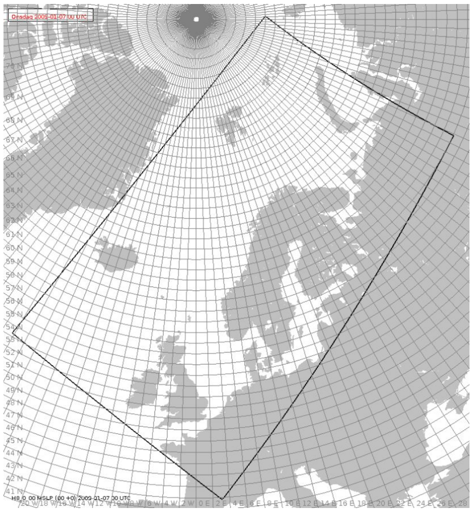


Figure 3.1: Area containing wind data, limited by 43E 003 in the southern corner and 83N 91'30 in the northern corner

3.1.3 Polar Low

Polar lows only appear in certain regions of the world during cold air outbreaks, such as the Barents Sea and the Norwegian Sea. A polar low is a low pressure system that can contain high speed winds with fast changing directions, which can be dangerous to maritime vessels. The reader is reffered to Rasmussen [1979] for a more general discussion of polar lows.

The wind field is normal, with an asymmetric wind field coming from the west and quite sharp contrasts between the different sections. Figure 3.2 displays a Synthetic Aperture Radar(SAR) picture, which shows the sharp contrasts well.

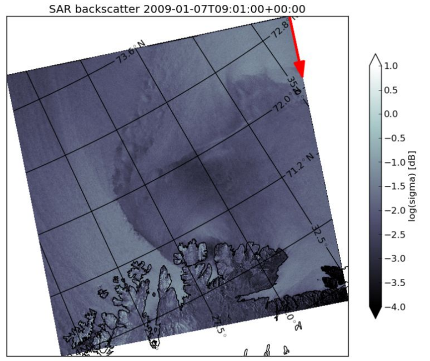


Figure 3.2: A SAR picture showng the sharp contrasts of a polar low.

The data is presented using a high resolution limited area model(HIRLAM) 8km model, which is an international model for presenting metrological data. This means the resolution set for the zone is 8 kilometers. The whole zone is divided into a 8km grid with wind measurements at each gridpoint. The wind is measured 10 meters above ground level, which is the international standard for climate measurements.

The most interesting part of the wind data is navigation inside and around a polar low. The polar low of this wind data is pictured in figure 3.3 and the eye of the storm is situated at approximately (72N, 30E) at 0 hours. Navigation around this point will be a focus of the experiments conducted in chapter 6.

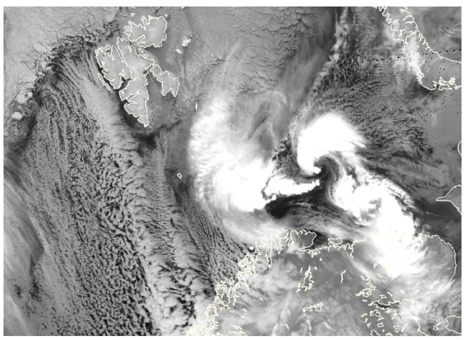


Figure 3.3: A satellite picture that clearly shows the swirling formations of skies around the polar low.

Inside the netCDF file the wind is represented as values in longitudinal and latitudinal directions, a plot of how some of the wind vectors look in a confined area is illustrated in figure 3.4.

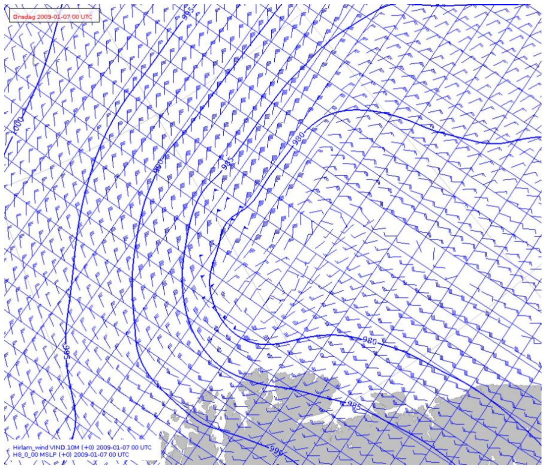


Figure 3.4: A plot of the wind barbs around the polar low, using weather data.

3.2 Ship Model

Ship model is the representation of the maritime vessel that will be used in the experiments. When choosing the ship model the author wanted to present a common ship model that users could relate to. Fishing vessels is a group of vessels which is used for commercial fishing. The ship model was chosen by using the most common specifications for Norwegian fishing vessels in 2013. Using data from table 3.1, taken from Fiskeridirektoratet [2014], it was decided to use a ship model shorter than 15meters.

A ship model under 15 meters with common fishing vessel attributes was chosen. The final ship model can be found in Appendix A.

3.2.1 Motor

The specifications for the motor used with the ship model was chosen as a QSL 9[1]. This motor was selected because it fits the ship model. The motor works

[1] QSL9 motor: `http://cumminsengines.com/qsl9-tier-4-final`

Fishing vessel length grouped on year	Year 2012	Year 2013
Less than 10 meters	3418	3347
10-10.99 meters	1483	1503
11-14.99 meters	729	719
15-20.99 meters	178	167
21-27.99 meters	147	137
28 meters and more	256	255

Table 3.1: Fishing vessels in Norway, grouped by length and year. The table is translated from Fiskeridirektoratet [2014].

at different intensity levels for different intentions. The continuous duty ratio was chosen because it covers the most common usages of the motor. The choice of motor together with the work intensity level sets a standard for the possible speed of the ship model and the fuel consumption rate.

3.3 Stability And Physics Overview

Simulating ship maneuvering and navigation are problems with great depth. Much effort has been spent on trying to develop dynamic ship models covering groups of maritime vessels, that can predict the effects of maneuvering characteristics from ship hull and propulsion models.

The elements that make up a maneuvering simulation can be defined as the environment, the vessel and the operator.

Environment is the world modeled as feasible and infeasible geographical locations. A feasible location is defined as a traversible location by the maritime vessel. An infeasible location contains something that makes the location intraversible, such as landmasses and other static challenges. The location also has to be contained within the wind data, since the solution space is limited by this area.

Vessel is represented by a ship model using common characteristics of fishing vessels in Norway at present time.

Operator is the captain of the ship and is included as the user of the algorithm, providing weighting for the preferences he wants from the algorithm run.

Hydrodynamic forces that act on a ship separate the problem from rigid body dynamics. Early attempt to map these forces were based on experiments in

Norrbin [1971]. A more recent method is captured by performing wind tunnel tests on a scale model and determine the wind loads by reference to known values of similar ships as in Blendermann [1994]. These models still rely on empirical formulations for the effects which cannot be captured by the underlying theory for the computations.

3.3.1 Simplified Observer Design Model

To simplify the complex nature of forces acting on a ship, an observer design model from Fossen [2011] was used. The design is used to observe how ships behave in their natural environment. The model includes a disturbance model, where the disturbance is the wind and the goal is to estimate the effect of wind on a maritime vessel. Other effects on the vessel, such as currents and waves, is not taken into consideration. Wind loads, and how they will affect the ship, is calculated from the Blendermann method. Seakeeping theory is used as a basis for the calculations. Seakeeping theory refers to the study of ship motion and ship stability.

3.3.2 Ship Stability

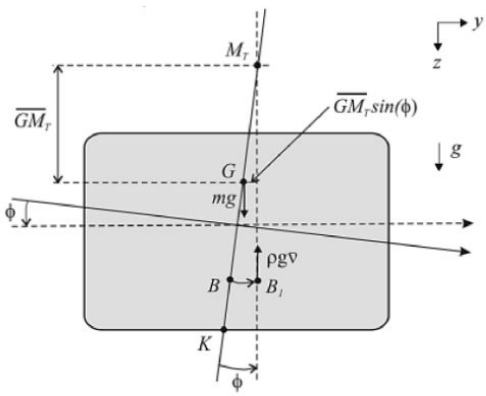


Figure 3.5: The different stability considerations taken into account, M_t is the metacentre, G is the center of gravity, B is the centre of buoyancy, K is the keel and ρg$\nabla$is the displacement of the model. Figure reprinted from Fossen [2011]

Stability consideration in ship literature is usually refered to as metacentric stability. The metacentric height(GM_t) of a ship is a measurement of the initial stability of a floating body. The metacentric height is calculated as the distance between the centre of gravity(G) of the ship and its' metacentre (M_t). The metacentre is defined as the point where the vertical of the new centre of buoyancy(B) meets the original vertical(B_1 through G) through the center of gravity(G). These values are used for computing the rolling angle (θ) of the ship, which is the amount that the ship will roll from side to side. Increasing rolling angles cause increased discomfort and can ultimately cause the ship to capsize. In figure 3.5 the different stability considerations of a maritime vessel are illustrated.

When a vessel is effected by an external force the centre of buoyancy B_1 is shifted to B and the ship will be tilted to an angle θ. The righting couple is the force acting to restore the ship to its' initial equilibrium position. The righting couple is defined as:

$$RightingCouple = \rho g \nabla * GM_t * sin(\theta) \tag{3.1}$$

where ρg∇ is the displacement of the ship, comprised of water density, gravity and weight, GM_t is the transverse metacentric height and θ is the rolling angle of the ship.

The righting couple, is equal to the external force acting on the ship. Since the external forces effecting the ship in this model is the rolling force created by the wind moment, the righting couple is the same as the rolling moment.

A boat can handle rolling angles up to a certain point where the boat will capsize. The angle a boat can roll to is strongly decided by the shape, size and cargo of the boat. To calculate the angles with precision a GZ-curve needs to be calculated for every individual boat. The GZ-curve is determined from extensive experiments in port. Because this is a rigorous process, not all boats have a pre-calculated GZ-curve. Additionally, it would be infeasible to use a GZ-curve as input. An alternative method is therefore used to calculate the stability of a ship in this thesis. Using the righting couple to find the rolling angle from the rolling moment. Paths that have increasingly high rolling angle are penalized.

Two attributes of a route are of particular interest in regard to stability:

1. Maximum Rolling Angle: This determines how safe the route is to traverse. Less rolling angle means less chance for capsizing and less chance of injury to crew or equipment.

2. Average Rolling Angle: This determines how comfortable the route is for passengers and is a particularly important aspect of travel for commercial passenger ships and leisure trips.

3.3.3 Blendermann's Method

To find the rolling angle for a maritime vessel, a method that can calculate the rolling moment is needed, as shown in equation 3.1. In Krata [2008] it is suggested that the first improvement of the present stability situation, is to get rid of using the same stability criterias for vessels of any size.

The method presented in Blendermann [1994] is developed through extensive testing in wind tunnels for different vessel models, ship hulls and ship characteristics, and provides different stability calculations for vessels of different size.

To calcule the rolling moment of a maritime vessel equation 3.2 is used.

$$CK = \frac{K}{\rho_A * A_L * H_M} \tag{3.2}$$

where K is the rolling moment, CK is the rolling moment coefficient, is the air density, A_L is the lateral area exposed to wind, and H_M is the mean height of the vessel.

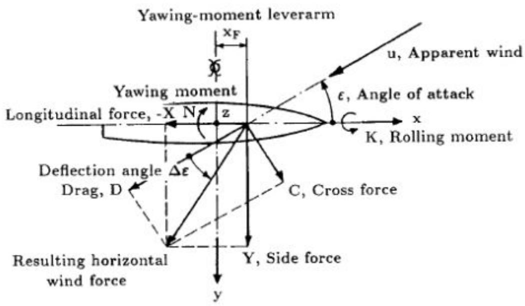


Figure 3.6: The different forces acting on a ship is illustrated. The figure is reprinted from Blendermann [1994]

For equation 3.2 we see that the rolling moment coefficient is needed to calculate the rolling moment. Every coefficient is calculated using ship model characteristics and the attack angle of the wind represented as epsilon in figure 3.6.

Using the simple loading concept of:

- Constant deflection parameter
- Linear yawing-moment lever arm

- Constant rolling-moment lever arm

we get the parametrical loading conditions for calculating the rolling moment coefficient, as shown in equation 3.3.

$$CK = \kappa * \frac{S_H}{H_M} * CY \tag{3.3}$$

where $\kappa$is reference data presented in table 3.2, and CY is the side-force coefficient of the maritime vessel.

Equation 3.3 shows that CK is a function of CY, so CY has to be evaluated first, using equation 3.4.

$$CY = CD_t * \frac{sin^2(\epsilon)}{1 - \frac{\delta}{2}} * (1 - \frac{CD_L}{CD_t}) * sin^2(2\epsilon) \tag{3.4}$$

The coefficients are calculated using the angle of attack from the wind(ϵ), ship model characteristics S_H and H_M, and reference data from table 3.2.

S_H is the position of the lateral-plane centroid above the waterline.H_M is mean height, defined by $H_M = \frac{A_L}{L_{OA}}$, where A_L is the lateral-plane area and L_{OA} is the length of the areal.

3.3.4 Ship Safety

It is assumed that wind causes the prime safety concern of the maritime vessel. The wind is one cause for icing on a boat and rolling of the boat. It can cause discomfort, safety hazard for the crew and equipment and in some cases capsizing.

Analysis of historical data has revealed that most accidents is caused by several elements coinciding at once. The relation between the elements is as a Venn diagram in figure 3.7. There is an increasing risk for accidents, the more elements that coincide.

In this thesis, the ship and environment elements are addressed. The captain still has a big responsibility of overseeing that proper standards are followed since an increase in cargo loading can induce a list, strong heel and even capsizing.

3.4 Distance Calculations

A path along the surface of the Earth connecting two points is a track. Two types of track lines are of interest geographically, great circles and rhumb lines. Great circles represent the shortest possible path between two points. Rhumb lines are paths with constant angular headings.

Type of Vessel	CD_t	CD_{LA_F} $\epsilon = 0$	CD_{LA_F} $\epsilon = \pi$	δ	κ
Car carrier	0.95	0.55	0.60	0.80	1.2
Cargo vessel, loaded/container on deck, bridge abaft	0.85	0.65/0.55	0.55/0.5	0.40	1.7/1.4
Container ship, loaded	0.90	0.55	0.55	0.40	1.4
Destroyer	0.85	0.60	0.65	0.65	1.1
Diving support vessel	0.90	0.60	0.80	0.55	1.7
Drilling vessel	1.00	0.70 ÷1.00	0.75 ÷1.10	0.10	1.7
Ferry	0.90	0.45	0.50	0.80	1.1
Fishing vessel	0.95	0.70	0.70	0.40	1.1
Liquified-natural-gas-tanker	0.70	0.60	0.65	0.50	1.1
Offshore supply vessel	0.90	0.55	0.80	0.55	1.2
Passenger liner	0.90	0.40	0.40	0.80	1.2
Research vessel	0.85	0.55	0.65	0.60	1.4
Speed boat	0.90	0.55	0.60	0.60	1.1
Tanker, loaded/in ballast, bridge aft	0.70	0.90/0.75	0.55/0.55	0.40	3.1/2.2
Tender	0.85	0.55	0.55	0.65	
Mean of data set	0.86	0.62	0.59		
Standard deviation	0.09	0.13	0.12		

Table 3.2: The wind loads coefficients are calculated using the reference data in this table, reconstructed from Blendermann [1994]

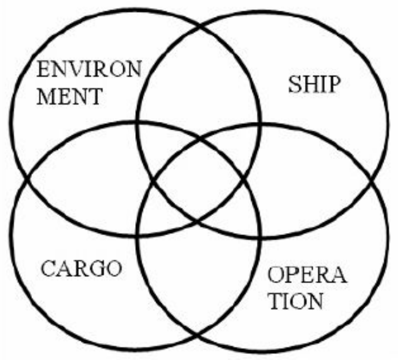


Figure 3.7: The relations between elements concerning ship safety are illustrated as a venn diagram. The figure is reprinted from Kobyliński [2007]

3.4.1 Great Circles

A great circle is the shortest path between two points on a sphere. It makes an intersection through the sphere that cuts through both points. The equator and all meridians are great circles. It is not always apparent that the great circle is the shortest path between two points because very few map projections (including the mercator projection used in chapter 6) represent great circles as straight lines.

Plotting great circles for navigation is a traditional method of path-planning and is also known as orthodrome navigation method.

The haversine formula is used for finding the great circle distance between two points. It calculates the distance, given the longitudinal and latitudinal coordinates. The formula for the haversine distance, taken from Sinnott [1984] is shown in equation 3.5.

Haversine distance:

$$\nabla\lambda = \lambda_2 - \lambda_1 \tag{3.5a}$$

$$\nabla\phi = \phi_2 - \phi_1 \tag{3.5b}$$

$$a_1 = \sin^2\left(\frac{\nabla\phi}{2}\right) + \cos(\phi_1) * \cos(\phi_2) * \sin^2\left(\frac{\nabla\lambda}{2}\right) \tag{3.5c}$$

$$a_2 = 2 * \arcsin\left(\min_{(1,\sqrt{a_1})}\right) \tag{3.5d}$$

$$d = R * a_2 \tag{3.5e}$$

where $\lambda$is longitude, $\phi$is latitude and R is the radius of the earth, and d is the distance between point 1 and 2, defined as (ϕ_1, λ_1) and (ϕ_2, λ_2) . A spherical Earth with 6371 kilometer radius is presumed in this thesis.

The formula for finding the mid-point of a great circle is defined in equation 3.6.

Great Circle Midpoint:

$$b_1 = \cos(\phi_2) * \cos(\nabla\lambda) \tag{3.6a}$$

$$b_2 = \cos(\phi_2) * \sin(\nabla\lambda) \tag{3.6b}$$

$$\phi_m = atan2\left(\sin(\phi_1) + \sin(\phi_2), \sqrt{(\cos\phi_1 + b_1)^2 + (b_2)^2}\right) \tag{3.6c}$$

$$\lambda_m = \lambda_1 + atan2(b_2, cos(\phi_1) + b_1) \tag{3.6d}$$

$$\tag{3.6e}$$

where $\lambda$is longitude, $\phi$is latitude R is earth's radius and atan2 is the function described in equation 3.7. (ϕ_m, λ_m) is the midpoint between (ϕ_1, λ_1) and (ϕ_2, λ_2).

$$atan2(y, x) = \begin{cases} \arctan(\frac{y}{x}) & x > 0 \\ \arctan(\frac{y}{x} + \pi) & y \geq 0, x < 0 \\ \arctan(\frac{y}{x} - \pi) & y < 0, x < 0 \\ +\frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ undefined & y = 0, x = 0 \end{cases} \tag{3.7}$$

Way-points in a great circle tracking is found by recursively finding the midpoint of the great circle.

3.4.2 Rhumb Line

A rhumb line, which is also called a loxodrome method of navigation, is a traditional method of navigation. The loxodrome method constructs a curve that crosses each meridian at the same angle. All parallels, including the equator, are rhumb lines, since they cross all meridians at 90 degrees. Additionally, all meridians are rhumb lines, in addition to being great circles.

Plotting a loxodromic path on a mercator map is drawing a straight line between two points. Navigating is also easy since you keep the same course throughout the whole journey.

The formula for finding the mid-point of a rhumb line is defined in equation 3.8.

Rhumb line midpoint:

$$\phi_m = \frac{(\phi_1 + \phi_2)}{2} \tag{3.8a}$$

$$c_1 = \tan\left(\frac{pi}{4} + \frac{\phi_1}{2}\right) \tag{3.8b}$$

$$c_2 = \tan\left(\frac{\pi}{4} + \frac{\phi_2}{2}\right) \tag{3.8c}$$

$$c_m = \tan\left(\frac{\pi}{4} + \frac{\phi_m}{2}\right) \tag{3.8d}$$

$$\lambda_m = \frac{(\lambda_2 \lambda_1) * \ln(c_m) + \lambda_1 * \ln(c_2)\lambda_2 \ln(c_1)}{\ln(\frac{c_2}{c_1})} \tag{3.8e}$$

where $\lambda$is longitude, $\phi$is latitude and ln is natural log. (ϕ_m, λ_m) is the midpoint between (ϕ_1, λ_1) and (ϕ_2, λ_2) .

Way-points in a rhumb line tracking is found by recursively applying the equation 3.8.

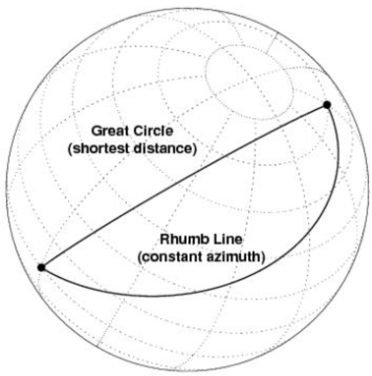


Figure 3.8: The figure shows the differences between a great circle and a rhumb line plotted on a sphere. Great circles and rhumb lines are called orthodrome and loxodrome navigation methods, respectively. The figure is reprinted from mathworks [2]

3.5 Genetic Algorithm Theory

3.5.1 Multi-Objective Optimization

When a problem has more than one objective for its' solution, and these objectives can be in conflict with each other, the problem becomes increasingly complex. It is necessary to weight or prioritize the different features of a path in some way. There are three main groups of different multi-objective evolutionary algorithms, separated by how input is processed:

A priori the importance of the parameters are chosen in advance of the running algorithm.

A posteriori a pareto-optimal front is found using the algorithm and the choice of selecting one of these paths is given to the user.

Alternating the weighting of parameters are done in alternating fashion while running the algorithm, usually in the form of questions and computing from these.

A priori is the choice for this algorithm, since it will search through the complete search space with the intention of the user from beginning to end.

All optimization problems have inputs, an objective function and output. The input is parameters that describe how the genetic algorithm will run. The objective function, also called fitness function, is used to measure the results that the algorithm produces and compare the results against eachother. The output is a solution or set of solutions that are optimal or close to optimal. The goal of optimization is to find an optimal solution within the variable's bounds, i.e. within the solution space, as in Weise [2009]

3.5.2 Evolutionary Computing

Evolutionary Computing is a group of algorithms which are loosely based on the Darwinian principles of "survival of the fittest". A genetic algorithm is a search heuristic that is inspired by natural selection. Imagine a species and their population. Some individuals have good traits, which is passed down to their children. The individuals with the best traits carry their genes on, while individuals with worse traits eventually die.

The genetic algorithms uses its own solutions and several rules to select the best solutions and improve upon them through iteration. These kinds of algorithms are prefered over mathematical optimization when the solution space is sufficiently big and complete knowledge of the domain is not necessarily known.

Genetic algorithms are especially useful in certain situations:

[2]Great circles, Rhumb lines, and Small circles: `http://www.mathworks.se/`

- The search space is big, complex and poorly understood
- Domain knowledge is scarce and expert knowledge is hard to use for ordinary searches
- No adequate mathematical analysis
- Traditional search methods fail

3.5.3 Life-Cycle

The cycle of life in an evolutionary algorithm is depicted in figure 3.9.

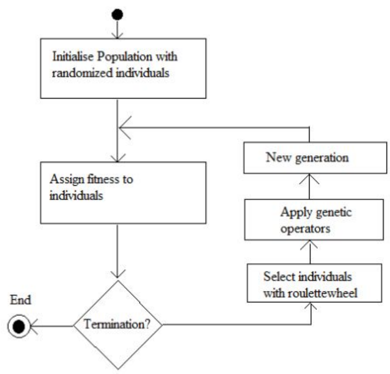


Figure 3.9: The main paradigms from the life cycle of a genetic algorithm are illustrated in this flow-chart.

The main paradigms of the cycle of life of solutions are the following ones:

1. An individual is referred to as phenotypes and contain genes, or genotypes, which is a representation of a solution to the problem.
2. To create a new child one or two parents is needed, depending on the evolutionary operator. Two parents can create a child using the crossover operation, which combines genes from both parents to create a child (for instance the first half of the gene from the first parent and the second half of the second parent). For instance, consider two parents:

$$P1 = 10101010 \text{ and } P2 = 11110000$$

if the crossover-point is set after point 5 in the genome, then the two children are spawned:

$$C1 = 10100000 \text{ and } C2 = 11111010$$

One parent can also create a child alone, this is done by changing parts of the parents gene through a mutation operation.

3. There exists a fitness value for every individual which determines the quality of the individual. Individuals are chosen to be parents through a selection mechanism, individuals with a high fitness value has higher probability of being chosen for reproduction.

4. The individuals can be removed from a population, their survival is strongly connected to their fitness values.

To understand the connection of the genetic algorithm to the biological process just described, one has to think of the following correspondences on the level of terminology described in Hromkovic [2010]:

An individual: A feasible solution

A gene: An item of the solution representation

A fitness value: A cost function

A population: A subset of the set of feasible solutions

A mutation: a random local transformation

3.5.4 Free Parameters

Following the concept of the life cycle of a genetic algorithm there are some parameters that has to be set for the concrete implementation of an algorithm. Most of these parameters are connected to a strategy for finding good solutions at a fast rate:

- Population size
- Selection of initial population
- Fitness function and parent selection mechanism

- Representation of individuals and the evolutionary operators
- Stop criterion

Population Size

Selection of population has a certain impact on the run of an algorithm:

1. A small population reduce the diversity of the population. This can result in that the population converge to a local optimum, which is a solution that may be significantly weaker than the fitness of a global optimum.
2. Large populations increase the chance of finding a global optimum.

Having large populations increase the computations for each generation, so there is important to find a good compromise between the time used for the algorithm and the quality of the solutions. In general, small population sizes are prefered in order to be competitive with other approaches with respect to algorithm time complexity.

Selection Of Initial Population

The traditional way to select an initial population is through completely randomizing the individuals in a population. Experience has shown that including pre-computed solutions into the initial population can increase the speed of convergence towards good solutions. However, one risk of using only pre-computed solutions is getting stuck at a local optimum.

Fitness Function And Parent Selection Mechanism

The simplest way to choose fitness for an optimization problem is to give each individual an exact fitness value, fitness(individual), which will give each individual in the population a chance of getting chosen as parents according to the following probability:

$$p(individual) = \frac{fitness(individual)}{\sum_{P_i \in Population} fitness(P_i)} \tag{3.9}$$

If the probability for most of the individuals in the population is similar, their probability of getting selected as parents is similar, resulting in what is called low selection pressure. This means even though some individuals are better than others, there is almost no higher chance of them getting selected as parents. The previous shown way of finding parents is called roulette-wheel selection. Another way is to rank the individuals according to their fitness and then give them probability of getting selected for parents from this ranking. This avoids the problem of similar probability in a population, and is called ranking selection.

Representation Of Individuals And Genetic Operators

The challenge of representating an individual is the same as representing an encoded solution. The individual has to be constructed so it is possible to change it easily using evolutionary operators. Some problems, like travelling salesmans problem(TSP), consider a permutation of 1,2,...,n as the representation of feasible solutions. This representation is not suitable with the crossover operation. Consider these two parents:

$$P1 = 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8 \text{ and } P2 = 2\ 4\ 6\ 8\ 1\ 3\ 5\ 7$$

and a crossover operation after point 4, giving the following children:

$$C1 = 1\ 2\ 3\ 4\ 1\ 3\ 5\ 7 \text{ and } C2 = 2\ 4\ 6\ 8\ 5\ 6\ 7\ 8$$

where neither of the children represents a permutation of 1,2,3,4,5,6,7,8. There are several ways to deal with this, for example allowing solutions that do not represent feasible solutions, such as C1 and C2. This can however result in that the number of individuals that do not represent any feasible solution grows fast. Another solution is to perform the crossover not on the exact solution of the problem, but on a specifically modified representation of the problem. Whichever way is chosen to represent the individuals and genetic operators they have to be chosen with great care.

Stop Criterion

The stop criterion determines when the genetic algorithm will terminate with a sufficiently good solution. This can be determined in several ways. For instance if an answer is needed within a specified time, a time criterion can be set, or if there is a desire to run the algorithm for a specific number of generations, one can set a fixed number of generations at the beginning. Another possibility is to measure the average fitness of the population at the end of each generation. If the fitness of the population did not change for the last few generations, then the algorithm can stop. However, if this method is too impatient the run can result in a local optimum.

Chapter 4

Genetic Algorithm Implementation

Implementation of the genetic algorithm, which was developed as a main part of the master thesis is presented in the following chapter. The genetic algorithm is developed from scratch to solve multi-objective weather routeing between two locations, with focus on safety related to the rolling angle of a ship model.

4.1 Search Space

The first challenge when implementing an algorithm that works with real world values is the representation of the search space. A real world problem has infinite amount of coordinates, so the amount of possible coordinates has to be limited in some way. Most traditional path planning algorithms use a search space that is either graphic-based (for example a roadmap such as google maps) or grid-based (for example cell decomposition) as in Lacki [2008].

Building a search map and searching for a path in it are the main computational tasks in the algorithm. Lacki [2008] arguments that discretized grid model is the simplest and most efficient way to represent real world problems. For this project the search space will consist of a grid constructed from the wind data. This has a predefined resolution for the grid, and the additional benefit that all the different wind data nodes are reachable and has the same chance of being chosen.

This also means that the path can only take on the geographical locations there are defined wind data for, which is the grid of 8km. The navigation path can not take on any geographical locations between the 8km nodes, which may result in a less smooth path than the global optimum. Using this method saves computational power and simplifies the algorithm search space. The search space is shown in figure 3.1.

4.2 Chromosome Representation

For any genetic algorithm the chromosome is the data that will represent a solution. The structure of the chromosome lays the foundation for how genetic operators can be used, and thus how the chromosome can improve. In early GAs, the binary digit alphabet was used. Janikow and Michalewicz [1991] has performed extensive experiments comparing floating and binary chromosomes and shown that the float-valued chromosomes is faster, provides more precision and is more concise.

The chromosome is represented as an array of all coordinates that should be traversed by the path, in chronological order. Each index consists of float-valued coordinates in the solution space grid, and a state variable. The first and last node of the array, which is the origin and destination of the navigation problem, is static and immutable.

The state variable determines whether the specific coordination is feasible. A feasible grid is traversable and unfeasible grids are not traversable because they contain land-masses, structures or other hindrances, which makes it impossible for a boat to travel through. A path is only feasible if all the locations in the chromosome grid is feasible and constructs a path between the origin and destination of the problem.

Initially it was decided that each coordinate in the chromosome representation had to be connected to the previous coordinate. During development of the algorithm it quickly proved that trying to have the algorithm find a path where every coordinate had to be connected was infeasible with the computing power and time constraints at hand. A try with 50 population and 300 generations gave no feasible paths, and no improvement in feasible nodes, only start node.This led to a change in the design, allowing "jumps" from one node to another that wasn't connected in the immediate vicinity.

In practice this means that an initially designed path, without every intermediary node plotted, would be infeasible:

$$\text{Initial Chromosome (invalid)} : [[1,0],[3,0]]$$

It had to contain the intermediary node [2,0] resulting in this feasible path:

$$\text{Initial Chromosome (valid)} : [[1,0],[2,0],[3,0]]$$

Because the initial representations caused crossover and mutation operators to create invalid paths most of the time, the final representation of paths was chosen so "jumping" between nodes are valid paths:

Final Chromosome (valid) : [[1,0],[3,0]]

Because of this representation, there had to be implemented a method to check every intermediary node between two coordinates in the path representation for their fitness values and feasibility. The revised design allows more paths to be feasible, which means less computation time wasted trying to find feasible paths and more computation time spent improving the feasible paths.

This representation of the chromosome allow for flexible length, there can be any number of nodes between the origin and destination node. Allowing for flexible length has the benefit of creating smoother paths and the potential for complex path-patterns reliant on wind data.

4.3 User Input

Before starting the algorithm a set of limitations must be set, to specify the problem for the run. The set of user inputs that needs to be set for each run are:

- Origin
- Destination
- Generations number
- Population size
- Fitness weights
- Ship model
- Wind data

Wind data is the meteorological data that limits the area there exists wind data for, and as a result the search space for the problem. Generations number is the termination criteria of the algorithm, which determines when the algorithm will end.

Ship model is the model that will be used in the simulation to traverse the set of paths that are found, and origin and destination defines the problem where the ship will travel.

Fitness weights are the weights that are assigned to the different fitness features of the algorithm, such as distance, maximum rolling angle and average rolling angle.The different weights are chosen from the user's preferations and intentions. For example a high weight on distance is prefered from an economical point of view, for big boats that can handle big rolling angles. Focus on maximum rolling angle on the other hand is important for small fishing vessels and passenger boats that can only handle a certain degree of rolling during a voyage. Average rolling can be of importance for passenger and transport vessels of personnel, where comfort is important. Whichever parameters the user values is input for the algorithm at the start, so the algorithm knows which way to evolve and what the desired outcome to the run is.

Population size is the amount of individuals a population can contain. A large population size will help diversity of the population since there is bigger chance of a suboptimal solution within the generation to be carried on. These solutions can mutate into different optimum than the optimal solution at the present time.

4.4 Solution Space

We already mentioned the search space, which is every possible values that the algorithm can take on, but we want to limit the search even further. The search space is limited to a rectangle around the origin and destination with an additional 10% padding on all sides of the rectangle, as shown in figure 4.1. This

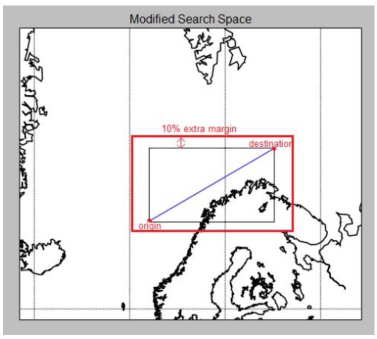


Figure 4.1: The modified search space, the solution space, is created by making a rectangle around origin and destination with a padding of 10%.

limitation is done to speed up the performance of the algorithm by restricting the search space of the algorithm. The likelihood that a path has to go outside the solution space is considered to be sufficiently low, and reduce the amount of unusable nodes included in paths.

4.5 Initial Population

The chromosomes are not completely randomized since they static origin and destination points. The length of a chromosome is determined as a random number between 1 and length specified in equation 4.1.

$$Chromosome_{MaxLength} = \sqrt{(\lambda_{destination}\lambda_{origin})^2 + (\phi_{destination}\phi_{origin})^2} \quad (4.1)$$

Between the origin and destination there are an amount of intermediary nodes corresponding to the chromosome length number. These nodes are randomized within the solution space. The initial population is created by filling a population with randomly created individuals up to the population size.

4.6 Selection Mechanism

In each generation some of the individuals of the population are chosen to get the privilege of reproduction, and passing their genes on. The fitness value of each individual in a population is normalized to the total fitness value of the whole population. The individuals are given a chance of getting chosen equal to their share of fitness compared to the rest of the population using equation 3.9.

This selection is called fitness-proportionate selection, also known as roulette-wheel selection, and is illustrated in figure 4.2.Each individual is given a percentage probability on a roulette wheel. The wheel is then spun by choosing a random value between 0 and 100. The individual that is assigned to the choosen value gets to reproduce and carry on its genes. The wheel is spun until a new population, with the user inputted population size of individuals, is chosen.

4.7 Reproduction and Evolutionary Operators

After the selection of individuals that gets to carry on their genes there is the reproduction phase, which consists of crossover operation and mutation operators. Crossover operation produces two new individuals with genetic traits from two chosen parents. Mutation will have genetic traits from one parent together with a mutation that is not inherited, but produced from a randomly chosen operator. The reproduction phase will continue until a new generation has been produced. The new population is then merged with the parents to create the new generation.

The evolutionary operators that are used in this thesis are shown in figure 4.3, listed and then explained in detail.

- Crossover operation

[1]Newcastle Engineering Design Centre: `http://www.edc.ncl.ac.uk/`

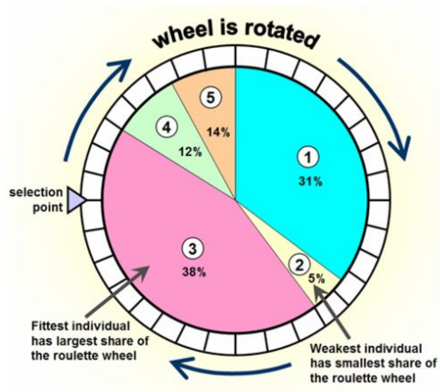


Figure 4.2: Roulette wheel selection mechanism applied to 5 individuals with different fitness values. Figure is reprinted from [1]

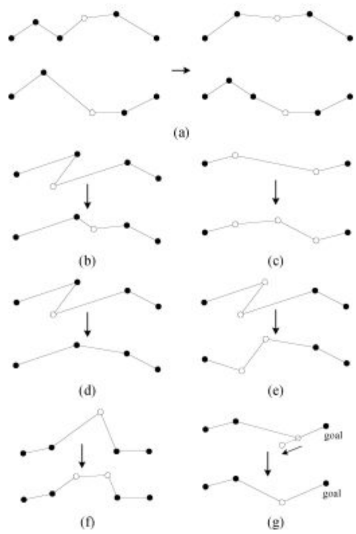


Figure 4.3: The different evolutionary operators are illustrated: a) crossover, b) perturb, c) insert, d) delete, e) swap, f) smooth, g) fixed-vector. The figure is reprinted from Zheng et al. [2005]

- Perturb mutation
- Insert mutation
- Delete mutation
- Smooth mutation
- Swap mutation
- Fixed-vector mutation

An evolutionary operation will always be applied during reproduction to a child. The chance of an operator being chosen is the same for all operators, which is 1/7, since there are 7 normal operators. The last operator, fixed vector mutator is only applied in special situations. This means that this algorithm has higher chance for mutation than traditional genetic algorithms, which usually has a higher chance for the crossover operation. The reasoning for this is that the chromosomes often are short, and in these cases mutation operators provide better diversity for the population.

4.7.1 Crossover Operation

The type of crossover operation used in this thesis is a one-point random crossover. This means that two parents are chosen together with a random point in their chromosomes. The first part of the genotype, divided by the crossover-point is taken from the first parent, and the second part from the other parent. These two parts are recombined to make a new child, this child has a chromosome which is a recombination of the two chromosome parts from both parents, as shown in figure 4.4.

4.7.2 Perturb Mutator

The perturb mutator is used to change one node in the chromosome representation to a random value, which can be either feasible or infeasible. Given a route the operator selects a random node between the origin and destination and changes the coordinates of that node to something random within the defined solution space. This operator is introduced to increase diversity in a population.

4.7.3 Insert Mutator

The insert mutator selects a random node in the chromosome representation and inserts a new geographical location at this point. The inserted point is a random

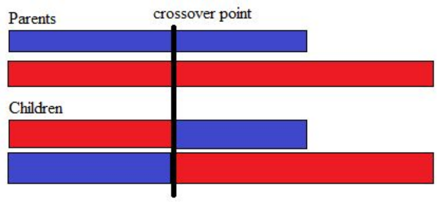


Figure 4.4: The crossover operation is illustrated in detail. The crossover point is chosen as a random point of the shortest parent. The chromosome parts are then extracted from both parents and recomposed into children.

location within the defined solution space. This operator is introduced to increase diversity in a population.

4.7.4 Delete Mutator

Delete selects a random node between the origin and destination nodes and removes it from the chromosome representation. This operator is introduced to remove unnecessary nodes from a chromosome, to make a path more effective.

4.7.5 Swap Mutator

The swap mutator selects a random node between index of the origin and second to last node. The node that is selected is swapped with an adjacent node in the chromosome representation. This operator is used to make small scale adjustments to paths.

4.7.6 Smooth Mutator

The smooth mutator selects a random node between the second index and second to last index of the chromosome. For the selected node, the mutator inserts two new nodes before and after the selected node, which take a random value from the path that the ship would traverse before reaching the selected node, and after reaching the selected node. The selected node is then deleted. This operator is used for cutting corners of sharp turns and smoothing paths.

4.7.7 Fixed Vector Mutator

This is a specialization mutator and only used when specifically requested by the destination point. The mutator selects the closest node to the destination if the harbour at this point has any specification on how the ship must approach the goal. For example in a harbour that must be approached from south, the mutator will mutate the node next to the goal so that the path adheres to this constraint.

4.8 Termination

The algorithm terminates when it reaches a set number of generations, specified by the user. The number of generation has to be big enough to allow sufficient evolution of solutions. 200 generations is found to be sufficient to get good results through extensive experimentation.

4.9 Fitness

The path-planning algorithm is a search to maximize the cost function, and the task of making a good fitness function is left to the designer. There is a trade-off between different constraints as discussed earlier, and these has to be fine tuned to present the correct path to the user. Trade-offs are regulated by making the user weight which fitness features he wishes to focus on.

Fitness is the measurement used to find the quality of a solution and should be an indication of how close the individual is to an optimal solution.

There are many ways to measure a maritime path., such as fuel usage, time used, distance traveled, optimal cargo placement and so on. In this system we have focused on wind moments on the maritime vessel and distance traveled.

The fitness score of an individual is determined by applying calculations to the path that the individual represents. The total length of a path, the maximum rolling angle and the average rolling anlge is calculated. These factors are weighted by using the user-specified preferences from the start of the algorithm. These weights help tailor each specific problem for the users intentions, whether its a leisure trip, fishing trip or tight time schedule.

The following features are included in the fitness function:

- Distance
- Maximum rolling angle
- Average rolling angle

4.9.1 Distance

The sum of all the distances connecting every node in the chromosome. The distance for a complete chromosome is calculate equation 4.2

$$Distance = \sum_{i=0}^{n-1} haversine(i, i+1) \tag{4.2}$$

where $i \in Chromosomenodes$, and haversine is described in equation 3.5. Chromosomenodes consist of every position that the maritime vessel has to traverse on a path, and will be refered to as Nodes in equations.

4.9.2 Maximum Rolling Angle

This fitness feature measures the rolling angle at every point in a path that a chromosome represents. The maximum rolling angle is highest recorded angle which a ship will roll from side to side out of all the points that a ship passes through. The fitness is calculated using equation 4.3.

$$MaxRollAngle = \sum\nolimits_{i \in Nodes} \left(\max_{angle(i)} \right) \tag{4.3}$$

where angle(i) is the wind angle at a specific node, found using the Blendermann method.

4.9.3 Average Rolling Angle

Similar to the maximum rolling angle fitness feature, this feature measures every rolling angle that the ship experiences on a path and calculates the average of these rolling angles. This fitness feature is calculated using equation 4.4.

$$AvgRollAngle = \frac{\sum_{i \in Nodes} angle(i)}{n} \tag{4.4}$$

4.9.4 Time and Fuel Usage

Time used during a path is quite interesting, however in this implementation the average speed of a maritime vessel is static, which results in that time used for a path is always equal to:

$$\text{Time used} = \tfrac{\text{distance}}{\text{average speed}}$$

Fuel usage measures the fuel used on the specified path. Since our motor has a fuel consumption rate of 54L/hr, the fuel is calculated as follows:

$$\text{fuel consumption} = \text{fuel consumption rate} * \text{time used}$$

Both of these fitness features are functions of the distance of the path, and since the distance is already included in the fitness function, fuel usage and time usage are left out of the final fitness function.

4.9.5 Maximizing Fitness Features

When the maximum rolling angle for a given chromosome is calculated, it is in the range between 0 and 180 rolling degrees, although a ship will usually capsize at around 45 degrees, the rest of the values are fictional and a result of the blendermann method of computing the degrees. The maximum rolling angle for a ship is found, by applying the methods in chapter 3 at every location of the chromosome representation. The desired maximum rolling angle is the smallest possible, however we want to maximize the fitness function. Because of this the maximum rolling angle is normalized against the maximum value it can reach, and then subtracted from 1, so the resulting fitness value of the maximum rolling angle is a value which we want to maximize. The process is shown in equation 4.5

$$Fitness_{maxrollangle} = 1 - \left(\frac{MaxRollAngle}{180} \right) \tag{4.5}$$

Similarly, average rolling angle is a value which should be as small as possible, and is made into a maximizing function using equation 4.6.

$$Fitness_{avgrollangle} = 1 - \left(\frac{AvgRollAngle}{180} \right) \tag{4.6}$$

Distance is also a value which should be as small as possible, and is turned into a maximizing function by first finding the shortest possible path between the origin and the destination using the haversine, in equation 3.5. The shortest path is divided on the distance found for this chromosome to find the fitness, as shown in equation 4.7.

$$Fitness_{distance} = \frac{haversine(origin, destination)}{distance} \tag{4.7}$$

4.9.6 Total Fitness

Now that we have the functions for each fitness feature it is time to put them together into a final fitness sum. The total fitness is a sum comprised by the functions of fitness features, and should give a good pointer to the quality of the solution, and how close it is to an optimal solution. This leaves the complete fitness function as a set of fitness features and corresponding weights:

$$\begin{aligned} Fitness = & (fitness_{maxrollangle} * weight_{maxrollangle}) \\ + & \quad (fitness_{avgrollangle} * weight_{avgrollangle}) \\ + & \quad (fitness_{distance} * weight_{distance}) \end{aligned} \tag{4.8}$$

where weights are user-specified focus on the different fitness features. These weights range from 0 to 1, where 0 is irrelevant weight and 1 is the most important weight.

4.10 Locations Between Nodes

Since the chromosome only represents the nodes that the algorithm has to visit, it does not contain every point that the vessel has to pass through. To get a complete picture of all the wind forces acting on the maritime vessel, wind force has to be calculated for every point between nodes represented by the chromosome.

To deal with this every actual step that the vessel has to traverse is calculated by defining stepping values for longitudinal and latitudinal directions between two nodes:

$$\nabla\phi = \phi_2 - \phi_1$$
$$\nabla\lambda = \lambda_2 - \lambda_1$$
$$LonStep = \frac{\nabla\phi}{\sqrt{\nabla\phi^2 + \nabla\lambda^2}} \tag{4.9}$$
$$LatStep = \frac{\nabla\lambda}{\sqrt{\nabla\phi^2 + \nabla\lambda^2}} \tag{4.10}$$

where ϕ is longitude, and λ is latitude.

The lonStep and latStep is then applied to the first node. If this increase in coordinates results in a new grid, the wind forces on the maritime model is calculated for this grid. These step values are applied iteratively to the new coordinates, and the wind forces are measured for every new coordinate that is found. This continues until the next node in the chromosome representation is reached. The same method is then applied on the two sequential nodes, and so on, until the destination in the chromosome representation is reached and every wind force on the maritime vessel model has been recorded.

It takes the previously calculated ship direction and puts it into the method for calculating winddata in every node that is passed, between two nodes.

4.11 Wind and Ship Angle Between Two Nodes

As we have seen there is a need for finding the wind at every position of a path. To get every position that the ship has to traverse the method proposed in the last section is used. However, to find the rolling moment of the ship you need the wind attack angle relative to the ship at every point. The wind attack angle relative to the ship is pictured as γ_w in figure 4.5

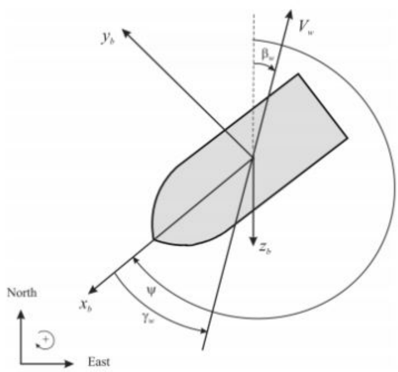


Figure 4.5: The figure illustrates different forces acting on a ship, where X_b is the ship course, V_w is the wind vector, and γ_w is the wind attack angle in relation to the ship vector. The figure is reprinted from Fossen [2011]

To find the wind attack angle, the ship course vector, X_b, and the wind vector, V_w is needed. The ship vector is calculated between two nodes in the chromosome using the equations for lonLength and latLength in the previous chapter. This ship vector is kept the same for every step between two nodes. The wind vector is looked up for every grid location that the ship traverse in the wind data file. The wind attack angle γ_w is then calculated using the formula for finding a dot product between two euclidean vectors shown in equation 4.11.

$$ShipVector * WindVector = ||ShipVector|| * ||WindVector|| * \cos(\gamma_w) \quad (4.11)$$

The resulting wind attack angle is then used used to check against the coefficient table produced from using equations 3.4 and 3.3. The coefficient is then used to find the rolling moment and ultimately the rolling angle of a ship at a given location using equations 3.2 and 3.1.

A coefficient table is first constructed from the ship model data and a given precision for angles, the precision the coefficient table is calculated for every whole rolling angle. This table is then stored in the ship model class of the algorithm for easy access, since it is accessed for every point that every solution traverses.

4.12 Time-Domain

For the wind data there exists data for different time steps. These are for every whole hour for 72 hours. One timestep is defined as one hour. If a path exceeds the maximum time that exists in the wind data, the last record of time data is used to find wind data.

Finding the time for each different point in a chromosome requires to find every point between the nodes in the chromosome. The length from origin is then calculated for these points using the haversine formula. The time used at that point is then found using the:

$$\text{time used} = \frac{\sum_{i \in Nodes} Length(i)}{AverageShipSpeed}$$

Paths that exceeds the time limitation should usually be purged, because the wind data at these points is not existing. However, most of these results will automatically be purged by the algorithm because of their long distances, and time is only dependant on distance. As long as there is some weight on the distance fitness feature of the algorithm these results will not make it to the final solution set.

4.13 Result Representation

The results from the genetic algorithm is the final child and parent generation. These are combined together and then the individual with the highest fitness is chosen as the final result. All of the results gathered through the different generations might be interesting to analyse to determine the speed of the algorithm, and how much each generation improves. The following features of a generation is stored in a results file for each run of the algorithm, together with their plots:

The average rolling angle average, max and min values for every generation.

The maximum rolling angle average, max and min values for every generation.

The distance average, max and min values for every generation.

Time used by the system measures the performance of the algorithm

The final path solution with corresponding information

The results file also contains a header with all the input parameters from the user, including the fitness features weights, generation number, population size, winddata, ship model, origin and destination. After this the rest of the information listed is stored in labeled arrays, so they can be extracted at any time for further analysis.

4.14 Data Flow of The System

Figure 4.6 shows the structure of the genetic algorithm and the main relations between the different classes discussed in this chapter.

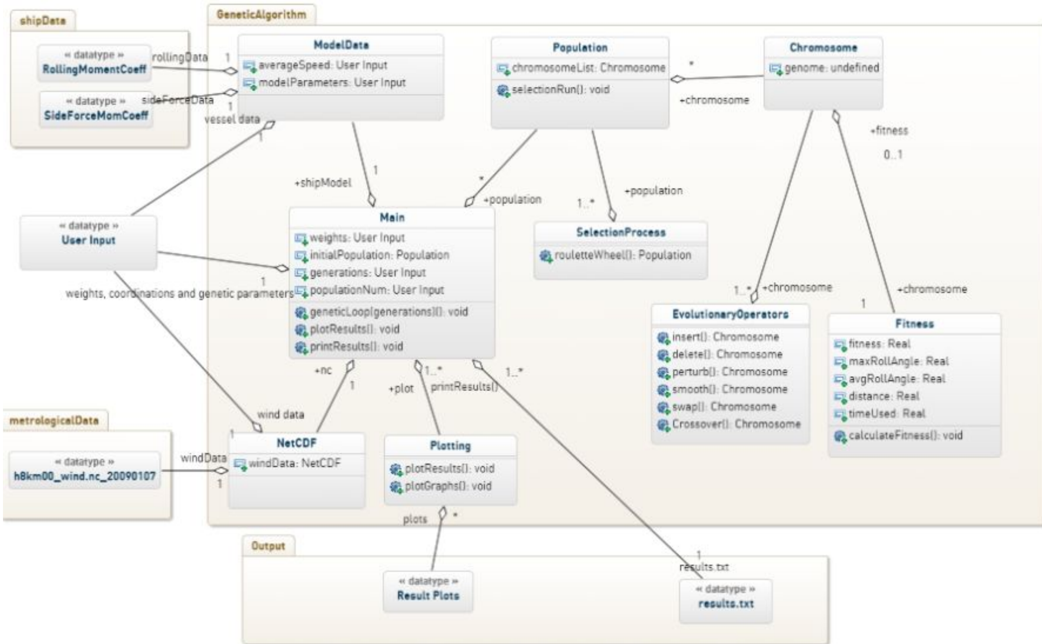


Figure 4.6: The structure of the genetic algorithm is provided in a UML Class Diagram. It shows a simplified overview of the main relations between classes, and their main tasks.

Chapter 5

Technical Details

The system is built from scratch using the Python programming language. This chapter lists the experimental setup, the extended frameworks and libraries that were used.

5.1 Experimental Setup

The input parameters vary between the experiments and is described at the start of each result. Two different systems were used for different parts of the experimenting in chapter ẽxperimentsandresults.

System 1 Phases 1 through 4 of the experiments were conducted on a fairly low-performance laptop, Samsung ultrabook with 1.8GHz Intel Core i3-3217U processor, 6GB memory running 64-bit Windows 7.

System 2 Phases 5 and 6 were performed on a dedicated Acer laptop with 2.3Ghz Intel Pentium T4500 Dual-Core processor, 2GB memory running 64-bit Ubuntu 13.04.

5.2 Programming Language

The source code for the system is written in Python 3.3[1], an interpreted, interactive, object-oriented programming language. The code is implemented using

[1]Python programming language: `https://www.python.org/`

the Eclipse ID[2] with PyDev[3] extension.

5.3 Wind Data Format

The file format that wind data was provided in was NetCDF4.0. NetCDF is an abbrevation for Network Common Data Form, and is a self-describing array-oriented scientific data structure, and widely used for meteorology. Since it is a standard file format for most wind data, the system is designed to have a NetCDF file as standard input if wind data is going to be used.

5.4 Libraries and Frameworks

The genetic algorithm is implemented from scratch to allow total control of it, and the necessary extensions needed for the maritime environment. It was assumed that it could be possible to find suitable and working framework without too much effort wasted. Time and effort was redirected to learn these modules, and extending them to some degree, to meet the program requirements. This dedication of time was considered a less time-consuming approach than to write similar frameworks from scratch.

Matplotlib[4] Matplotlib is a python 2D plotting library which produces publication quality figures.Requires the following dependencies:

- PyParsing
- NumPy
- Python-Dateutil
- Six
- Pytz

PyParsing[5] PyParsing creates and executes simple python grammars.

NumPy[6] NumPy is a fundamental package needed for scientific computing with Python.

[2]Eclipse IDE: `http://www.eclipse.org/`
[3]Python Developer: `http://pydev.org/`
[4]Matplotlib: `http://matplotlib.org/`
[5]PyParsing: `http://pyparsing.wikispaces.com/`
[6]NumPy: `http://www.numpy.org/`

Python-Dateutil[7] Python-Dateutil provides powerful extensions to the standard datetime module.

Six[8] Six is a Python 2 and 3 compatibility library.

Pytz[9] Pytz provides world timezones definitions, modern and historical.

NetCDF4-Python[10] NetCDF4-Python is a library for reading NetCDF files. Requires:

- HDF5

HDF5 [11] HDF5 is a pythonic interface to the HDF5 binary data format.

[7]Python-Dateutil: `http://labix.org/python-dateutil`
[8]Six: `https://pypi.python.org/pypi/six`
[9]Pytz: `https://pypi.python.org/pypi/pytz`
[10]NetCDF4-Python: `https://code.google.com/p/netcdf4-python/`
[11]HDF5: `http://www.h5py.org/`

Chapter 6

Experiments and Results

Experiments were performed using the implementation of the genetic algorithm described in chapter 4. The experiments consist of extensive runs for interesting applications of the algorithm. The chapters starts off with a plan for conducting the experiments, then provides a description of the scenarios the algorithm is tested in. Finally the configurations for the algorithm, and the findings from the results are presented.

6.1 Experimental Plan

The most interesting application of the algorithm was around and through a polar low. The polar low was close to or on top of land masses. To fully test the algorithm's capability in the interesting applications, and due to the lack of sufficiently detailed terrain data, the landmasses are disregarded in the results. This mean that even if a path is on top of terrain, it is still regarded as a valid path, and is working as intended. The results of the experiments plan are divided into phases to cover all research questions. The phases are listed in the following section, and the research questions they cover.

6.1.1 Pre-Algorithm Configuration

This phase is conducted to configure ship model, and setting up algorithm to be to handle the wind data.

6.1.2 Phase 1

RQ1 Can a genetic algorithm find an optimal path in the maritime environment?

To see if the implementation of the algorithm can find an optimal path in maritime environment a simple run is executed. The results shows the set of solutions at different generations to show how the algorithm improves upon the paths, and ultimately finds a set of viable solutions.

6.1.3 Phase 2

RQ2 How will the genetic algorithm compare to more traditional methods of path-planning?

In the second phase, a path vertically through a polar low is considered. The algorithm evolves a path with reduced risk to the maritime vessel, caused by wind forces. It then compares the different fitness features of a path to the more traditional methods, loxodrome and orthodrome path-finding.

6.1.4 Phase 3

Similar to phase 2, the experiment performed in this phase shows a horizontal path through a polar low. This experiment reveals the differences between loxodrome and orthodrome paths more clearly. The genetic algorithm findings is again compared to the traditional methods, according to RQ2.

6.1.5 Phase 4

RQ3 How will the genetic algorithm handle particularly interesting applications, such as navigating through a polar low, and navigating out of a polar low?

Navigating out of a polar low is a particularly interesting application of the algorithm. Not much literature exists on how to navigate out of storms, and finding the safest path could help save lives. This phase attempts to find the safest path out of a polar low.

6.1.6 Phase 5

In phase 5 a longer run is executed with higher genetic parameters. The path starts inside a polar low and escapes to Tromsø.

6.1.7 Phase 6

Phase 6 concludes the final experiment in this thesis and is an extended run using considerably larger genetic parameters. It escapes the polar low towards the west. It uses considerably larger genetic parameters to try to find intricate patterns through the polar low, evolved over more generations.

6.2 Scenarios

The different scenarios that are of interest for the application of this algorithm, given the wind data available, are listed in the following section.

6.2.1 A Normal Route

A seemingly safe path over a long distance. This path seems safe at the start of the journey, but if we investigate the time-domain it could intersect with a storm at a later point. This scenario will investigate the time domain of the algorithm to see how this will be handled.

6.2.2 Polar Low Between Two Nodes

A path between two geographical locations can have a polar low between them. When having to cross a polar low there arises the question of whether to try to avoid the polar low completely by going all the way around it, or just take the chance of going straight through. Gains by going straight through is a shorter path distance, but in real environments this does not necessarily translate to less time and fuel spent. The gains of taking the longer route around a polar low are primarily in safety of the ship and crew.

6.2.3 Path Out of Polar Low

This scenario investigates how a path is constructed when a ship is already stuck inside the middle of a storm. In this scenario distance of a path is of little importance, and is assigned only a small value to prevent the ship from going in circles or staying at the same location. This scenario is the most dangerous and paths out of a polar low has to be carefully planned.

6.3 Algorithm Parameters

Before running the algorithm some considerations has to be made, they are addressed in the following section.

6.3.1 Weights

The weights has a big impact on the run of the algorithm. They are specified in advance of running the algorithm, so the user can decide which fitness features to focus during each run. The weights are all normalized between 0 and 1 to allow for desired configuration. For instance a user could specify that he wants to focus

50% on distance of the path and 50% on maximum rolling angle for one run. The algorithm would then treat the two fitness features with equal focus, and totally disregard average rolling angle, which would have 0%. In this case distance would have weight 0.5 and maxAngle 0.5. It is also allowed to put 70% on maximum rolling angle, 70% on average rolling angle and 10% on distance, and thereby exceed 100 %. In this case the weighting will automatically be normalized back to 100 %.

6.3.2 Search Space

The search space is limited by the origin and destination as explained in chapter 4. This process eases the load on the algorithm, and disregards locations that are highly unlikely to be included in an optimal solution to the problem. This lets the algorithm find better solution in less time, which is an important step towards making the tool faster and more usable for a potential end-user. The search space is easily observed in figure 6.3a.

6.3.3 Plotting of Solutions and Graphs

All experiment results were stored in results files. The results were then handled by existing python frameworks and configured manually for each case to get the specified visual figures presented in the results.

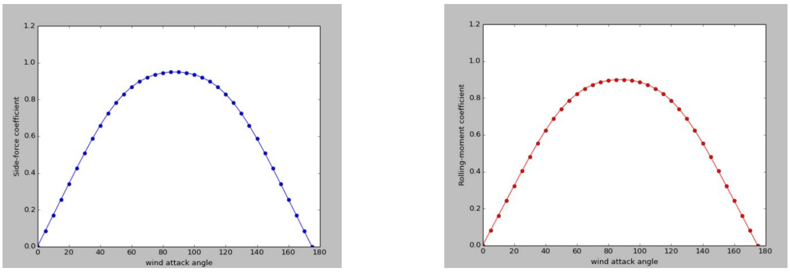
6.4 Pre-Algorithm Configurations

To be able to determine the rolling moment that is affecting the ship at a given time, we first need the rolling moment coefficient. The rolling moment coefficient is a dimensionless value determined through extensive empirical testing in Blendermann [1994].

Equation 3.4 and ship values from appendix a is used to determine the side-force coefficients displayed for every angle of wind attack in figure 6.1a.

The wind attack angle is the γ_w angle displayed in figure 4.5, on either side of the ship. The rolling-moment coefficient is the value used for calculating the rolling angle given the provided wind attack angle. These graphs are calculated before the algorithm run and saved as a table used for easy lookups during the run of the algorithm. The coefficients are stored for every whole wind attack angle.

Since the rolling-moment coefficient is a function of the side-force coefficient, the side-force coefficient is calculated first, then the rolling-moment coefficients is calculated as shown in figure 6.1b.



(a) Side-force coefficient (b) Rolling moment coefficient

Figure 6.1: The wind load coefficient values are plotted for every wind attack angle.

Even though we have most attributes from the fishing vessel in appendix a, some additional calculations had to be done to get the full list of parameters that the method used in Blendermann [1994] requires. For equation 3.2 S_H, the lateral-plane centroid above the water line and H_M, the mean height of the ship model has to be calculated. A slicing program was used to capture all outer parts of the ship, resulting in figure 6.2.

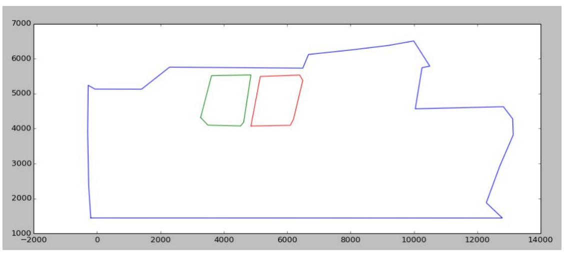


Figure 6.2: The ship lateral plane areal is found from creating a polygon of the ship specifications sheet in Appendix A. The axes are given in millimeters.

The lateral areal of the plane is then calculated from the polygon with corner coordinates (x_i, y_i), i=1,..,n using polygon equation 6.1.

$$Areal_{lateral} = \frac{1}{2} * \sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i) \quad (6.1)$$

To find the lateral-plane centroid above the waterline, the centroid in a lateral plane of a polygon was used, and is shown in equation 6.2.

$$Centroid_{lat} = \frac{1}{6A_L} \sum_{i=0}^{n-1}((y_i + y_{i+1}) * (x_i y_{i+1} - x_{i+1} y_i)) \quad (6.2)$$

where A_L is the lateral plane areal.

To find H_M, lateral areal is divided by length of areal. The final values for the ship, calculated using values from appendix A, and the ship-specific data, are presented in table 6.1

Type	Symbol	Value
Displacement	(tonnes)	62.42
Transverse metacentric height	GM_t (m)	0.646
Lateral area	A_L (m^2)	63.226
Mean height	H_M (m)	4.159
Lateral Plane centroid	S_H (m)	3.58

Table 6.1: The ship specific data are calculated using the equations listed in this chapter, ship specific data and lateral plane areal.

the table concludes the necessary ship parameters for the algorithm.

6.5 Phase 1

For this experiment we want to see if it is possible to use a genetic algorithm to find an optimal path in a maritime environment. As we know from chapter 2, genetic algorithms has been successfully applied to path-finding problems in several other domains. Only weighting distance would reduce the problem to finding the shortest path between two points, which would be identical to problems in other domains. By introducing weighting of the different fitness features values, the path-finding problem becomes multi-objective and significantly more complex.

The following experiment is designed to show how the algorithm will deal with equal focus on both distance and maximum rolling angle.

The following list of parameters were used in this experiment:

Population size 20

Genenerations number 100

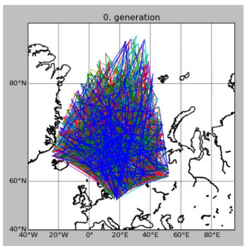
Weight: Maximum rolling angle 0.5

Weight: Distance 0.5

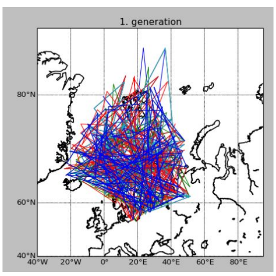
Origin (8E, 70N)

Destination (48E, 74N)

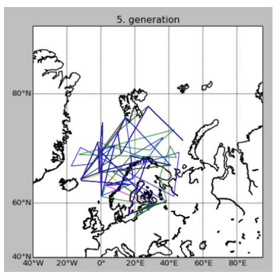
For the maximum rolling angle feature to have an impact on the solution, the wind in the chosen area has to be significant enough to effect the ship to some degree. The chosen area for experimenting is a path with a polar low located between the origin and destination.



(a) Generation 0



(b) Generation 1



(c) Generation 5

Figure 6.3: The set of solutions produced in phase 1 by the genetic algorithm at 0, 1 and 5 generations.

Figure 6.3a shows the first generation, which is the initial set of solutions. This set of solutions are a set of random paths created within the modified search space, solution space, described in chapter 4. We can clearly see the outlines of the solution space in this generation as the randomized solutions does not exceed these borders. This figure does not give any indication of potential solutions.

Already for generation 1 in figure 6.3b, we can see a significant improvement in the density of the paths, which means that the algorithm is significantly decreasing distance of the solutions.

At generation 5 we see a significant improvement in that different paths can be separated from each other, displayed in figure 6.3c.

This incremental improvement of the solutions continue steadily. The improvement to the solution set is easiest to quantify by looking at graphs for fitness for each generation, as depicted in figure 6.4.

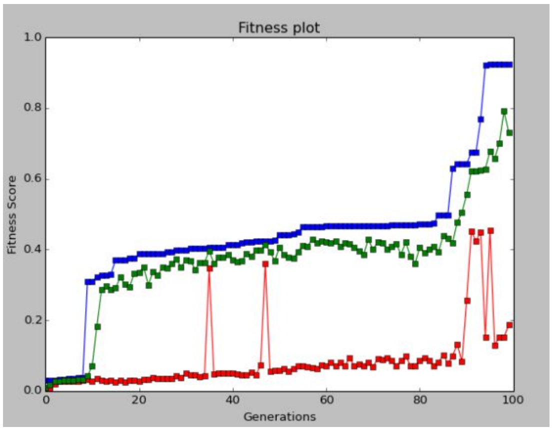
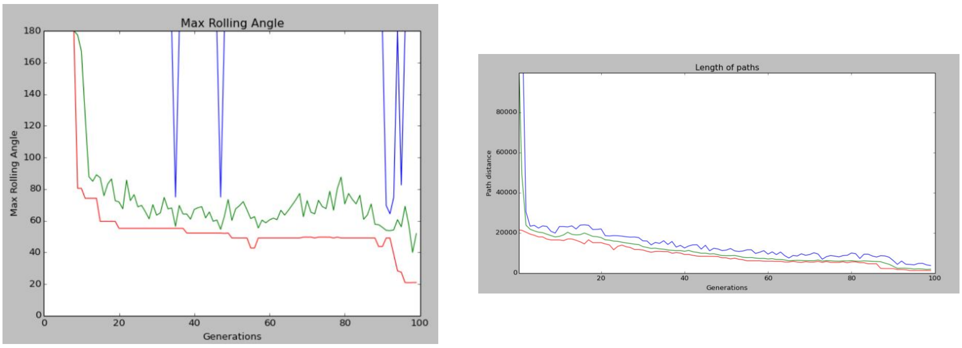


Figure 6.4: The fitness score of phase 1 at each generation. Blue marks the best fitness, green marks the average fitness and red marks the worst fitness of each generation.

To see how the fitness is calculated, we can look at the fitness features that make up the fitness graph. Maximum rolling angle and distance functions are displayed in figure 6.5a and 6.5b.

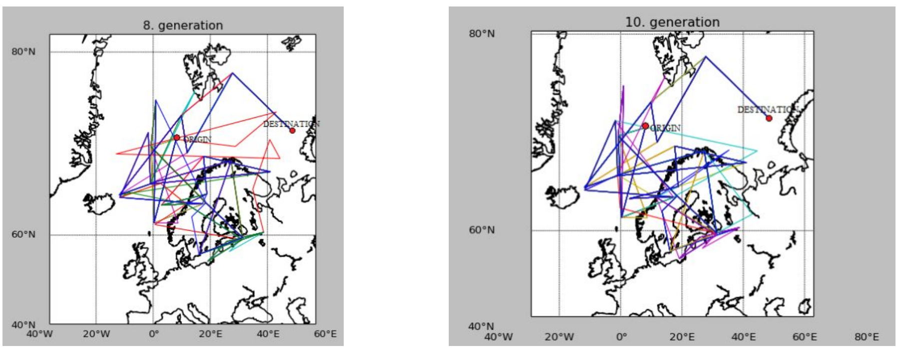
Looking closer at the differences in key generations we see that there is a significant increase in the fitness from generation 8 to 9 and 86 to 93. By studying the changes between these key generations we can look for indications of how paths evolved and why the fitness change was so significant.

From generation 8 to 10 there is no obvious visible pointers to why the fitness of the paths has increased so significantly. This is mainly because there is only a



(a) Maximum rolling angle (b) Distance

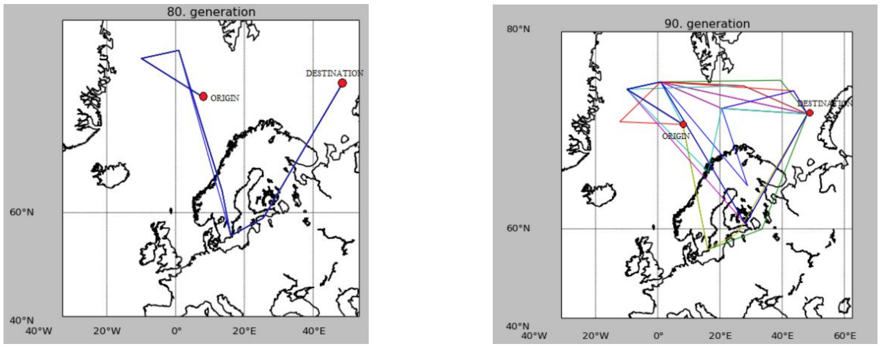
Figure 6.5: The maximum rolling angle and distance graph at phase 1 for each generation is measured.Blue marks the highest, green the average and red the lowest values for each generation.



(a) 8. generation (b) 10. generation

Figure 6.6: The set of solutions at 8 and 10 generations.

big increase in the fitness score of the best paths, as illustrated in the fitness graph. The worst solutions are still as bad as they were prior to the 10. generation and is cluttering up the picture. Another reason can be that the increase in fitness is mainly due to the big increase in rolling angle fitness, as suggested by the rolling angle graph. Observing a path representation is not going to give a good idea of how the rolling angle has improved, without seeing how the wind is affecting the paths.



(a) 80. generation, local optimum. (b) 90. generation, escaped local optimum.

Figure 6.7: The set of solutions at 80 and 90 generations. At 80 generations the generation has reached a local optimum, this is escaped at 90 generations, and the set of solutions keep evolving towards better solutions.

At generation 80 we see that all the paths of the population are bundled together in a thick line. This indicates that the algorithm has become stuck in a local maximum. We can easily see that this solution is not an optimal solution.

As the algorithm progresses towards generation 90 this local minima is escaped and more reasonable solutions emerge. When looking at the final generation, generation 99, we see that the set of solutions have improved significantly and are converging towards what appears to be a good solution for the global optimum.

As seen earlier an increase in rolling angle fitness is not as easy to see on a path plot, however a significant increase in the distance fitness feature resulting in shorter paths is easy to see, and is obvious from generation 90 to 99.

The final solution is then extracted from the final generation, generation 99, by choosing the best solution ranked by fitness score. This solution will be the final solution presented to the user. The chromosome representation of the final solution is:

Chromosome [[326, 97, True], [426, 103, True], [470, 158, True]]

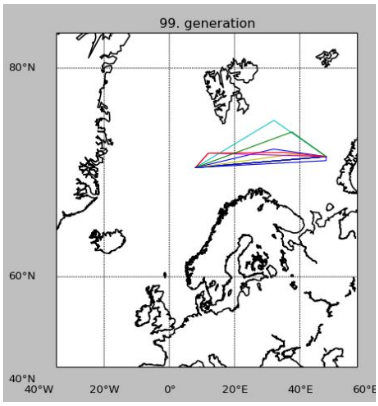
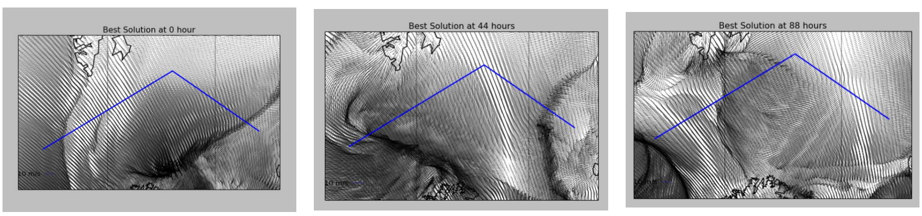


Figure 6.8: The set of solutions produced by the genetic algorithm 99 generations into the run. The solutions seem to be converging towards a global optimum.

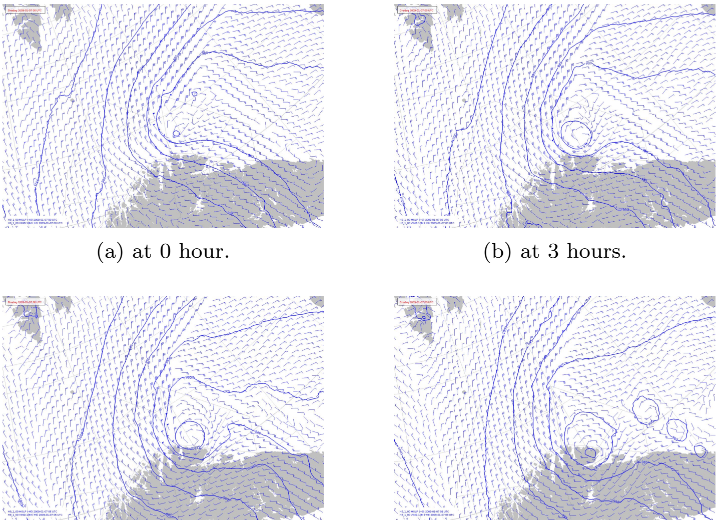
The best solution is shown with wind field at the start, halfway and at the end of the route in figure 6.9a



(a) at 0 hour. (b) at 40 hours. (c) at 88 hours.

Figure 6.9: The final solution from phase 1, with wind data 0, 40 and 88 hours from origin. The algorithm is trying to align the path with the wind quivers, so the least amount of rolling angle is experienced.

From the wind fields on the final solution we can see a tendency of wind vectors hitting the ship from aft or bow. This minimizes the amount of wind



(a) at 0 hour. (b) at 3 hours.

(c) at 6 hours. (d) at 9 hours.

Figure 6.10: The area around the polar low is shown together with wind barbs at 0, 3, 6 and 9 hours.

hitting the side of the ship, and thus reducing the rolling moment of the ship model, which is the goal of weighting the maximum rolling angle of the ship.

From the result in generation 99 we can see that even with 100 generations the algorithm is not completely converged at a singular result, this is highly contributed to the algorithm getting stuck for several generations at a local maximum which we saw at generation 80. This slows the algorithm down significantly for several generations. After extensive testing of the algorithm and comparing to other implementations it was decided that a population of 20 and 150 generations was sufficiently high genetic parameters for reaching good solutions for the different phases. These parameters will be used as a standard for further experiments.

6.6 Phase 2

Phase 2 will investigate path-finding through a polar low. Looking at the polar low with 3 hour time steps we see from figures 6.10a, 6.10b, 6.10c, and 6.10d that the polar low is moving south.

It is desired to investigate path suggestions by moving through the polar low vertically from south to north. Disregarding landmasses we take the origin south of the polar low at time 0 and see how the solution will develop when trying to go straight through the polar low.

In this scenario we are interested in avoiding any unnecessary risks and having a comfortable route. To deal with these criterias the weighting of all fitness features is moderate, yielding the following parameters:

Generations 150

Population 20

Weight: Max rolling angle 0.4

Weight: Average rolling angle 0.3

Weight: Distance 0.3

Origin (26E, 67N)

Destination (34E, 75N)

The solution found after 150 generations is:

Chromosome [[342, 217, True], [423, 166, True], [423, 130, True]]

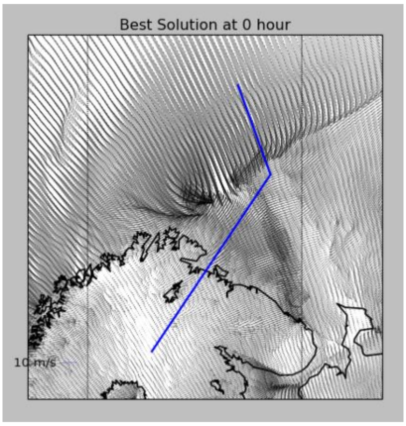
The solution is illustrated in figure 6.11a, with windforce at time 0. The final solution found uses 70.22 hours at the distance, so the following three figures of the solution are taken at timesteps 0 hour, 35hours and 70 hours:

In figure 6.11d we can see the best solution plotted together with the more traditional methods of navigation, loxodrome and orthodrome:

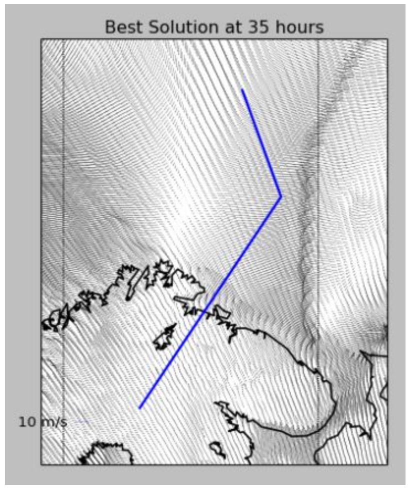
Loxodrome and the orthodrome fitness features are measured against the genetic algorithm's fitness features in table 6.2 and 6.3.

Method	Max Rolling Angle	Avg Rolling Angle	Length	Fitness Function
Loxodrome	31.386 deg	8.929 deg	934.21 km	0.915
GA	12.103 deg	3.704 deg	1034.88 km	0.938
GA Improvement	38.56 %	41.48%	110.78 %	102.51 %

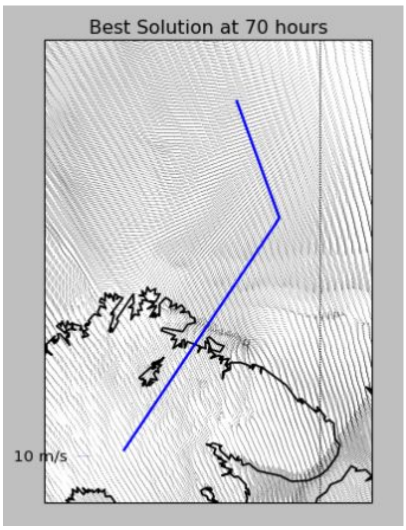
Table 6.2: A quantification of the differences in fitness features, measured between the loxodrome method and genetic algorithm in figure 6.11d.



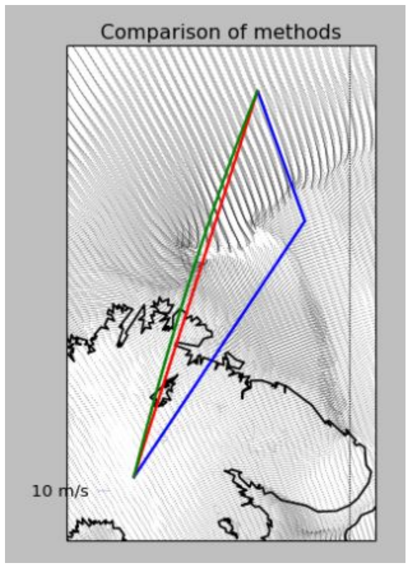
(a) at 0 hour.



(b) at 35 hours.



(c) at 70 hours.



(d) method comparison.

Figure 6.11: Phase 2 solution with wind data at 0, 35 and 70 hours, and finally a comparison between different navigational methods. Blue is the GA, red is the loxodrome method and green is the orthodrome method.

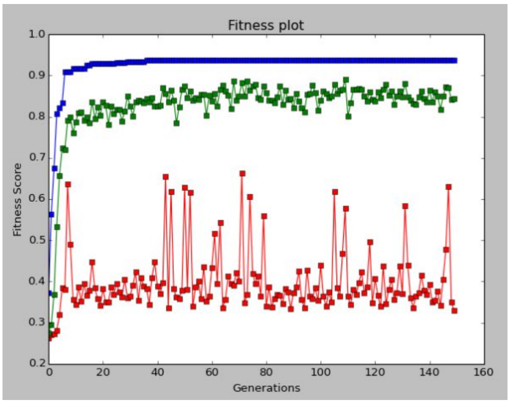


Figure 6.12: The fitness graph for every generation of the run is illustrated. Blue is the highest fitness, green is the average and red is the lowest for each generation.

We can see that the loxodrome path has over 31 degrees rolling angle at some point during its' voyage, which in some cases can be enough to capsize a small fishing vessel with low stability. Despite this high difference in maximum rolling angle the loxodrome has almost as good fitness as the genetic algorithm. This is an indication of an imperfect fitness function, and for further experiments this is compensated by weighting the maximum rolling angle fitness feature higher.

Method	Max Rolling Angle	Avg Rolling Angle	Length	Fitness Function
Orthodrome	37.928 deg	8.502 deg	933.533 km	0.902
GA	12.103 deg	3.704 deg	1034.88 km	0.938
GA Improvement	31.91 %	43.57 %	110.86 %	102.88 %

Table 6.3: A quantification of the differences in fitness features, measured between the orthodrome method and genetic algorithm in figure 6.11d.

We see similar tendencies in the orthodrome's statistics, but this path has even worse maximum rolling angle, and a slightly shorter length.

6.7 Phase 3

Loxodrome and orthodrome methods for plotting a path is quite similar when traversing the earth vertically, however by going horizontally their differences becomes more apparent. Phase 3 is conducted by moving horizontally through the polar low, changing the origin and destination from the previous phase. The weighting is also changed in order to see how the fitness function will react. The changed parameters are as follows:

Origin (14E, 71N)

Destination (44E, 72N)

Weighting: Maximum rolling angle 0.8

Weighting: Distance 0.2

The solution after 150 generations is:

Chromosome [[325, 136, True], [358, 183, True], [449, 183, True]]

A look at the comparison of the different methods is presented in figure 6.13. There is a noticeable change in how the loxodrome and orthodrome methods diverge from each other, compared to the previous phase.

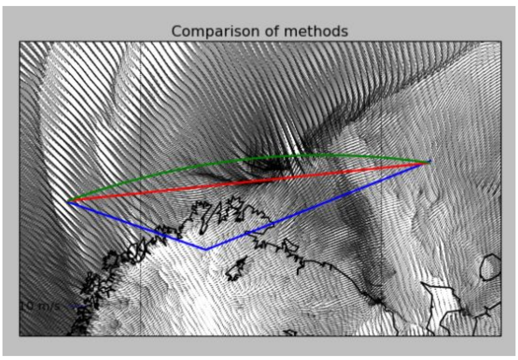


Figure 6.13: A comparison between different navigational methods are shown for phase 3, at 0 hours. Blue is the genetic algorithm, red is the loxodrome method and green is the orthodrome method.

In figure 6.13 the genetic algorithm seems significantly longer than the other paths. A closer look at the statistics of each route is provided in table 6.4 and 6.5.

Method	Max Rolling Angle	Avg Rolling Angle	Length	Fitness Function
Loxodrome	28.128 deg	10.058 deg	1064.05 km	0.873
GA	15.353 deg	8.652 deg	1182.31 km	0.911
GA Improvement	54.58 %	86.02 %	111.11 %	104.35 %

Table 6.4: A quantification of the differences in fitness features, measured between the loxodrome method and genetic algorithm in figure 6.13.

Method	Max Rolling Angle	Avg Rolling Angle	Length	Fitness Function
Orthodrome	32.505 deg	10.170 deg	1053.12 km	0.856
GA	15.353 deg	8.652 deg	1182.31 km	0.911
GA Improvement	47.23 %	85.07 %	112.27 %	106.43 %

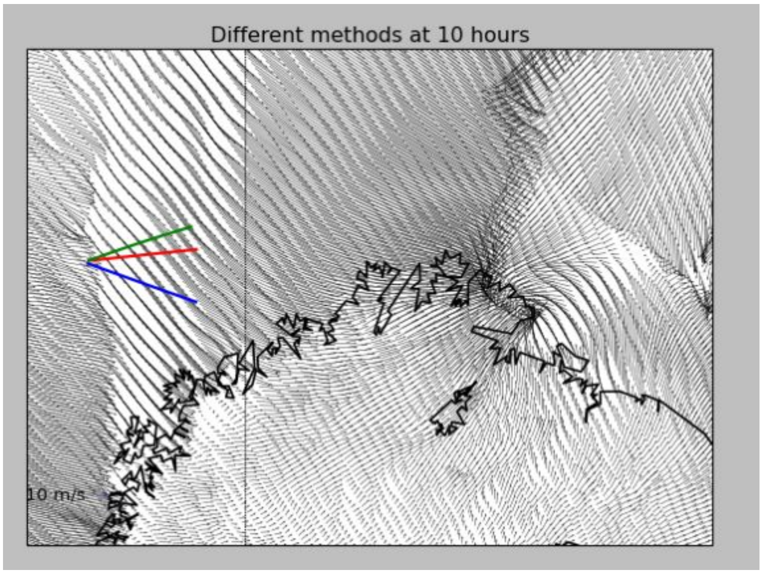
Table 6.5: A quantification of the differences in fitness features, measured between the orthodrome method and genetic algorithm in figure 6.13.

We can see a reduction to 54.58% of the rolling angle in the loxodrome case and a reduction to 47.23% of the rolling angle in the orthodrome case.

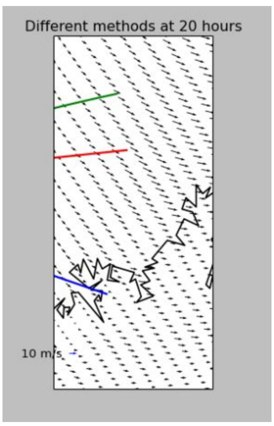
Since it is a bit hard to picture why the genetic algorithm is so much better in regards to rolling angle, from the simple illustration, we can study wind field pictures where all the paths are plotted. The total time used on the track for loxodrome method is 65.7 hours, orthodrome 65 hours and GA 76hours.

We take a look at the wind field at 10, 20, 30 and 50 hours. The maritime vessel model is at the end of the illustrated path at the specified time from origin.

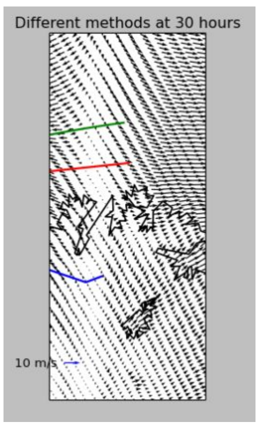
From these results we can see that the wind that hits the ship model in the genetic algorithm, usually hits from behind, at better angles or the with lower speed than the other methods. This is especially easy to see in figure 6.14b, where the wind arrows align perfectly with the ship course, and at figure 6.14d where the wind is especially weak at the exact point of the ship.



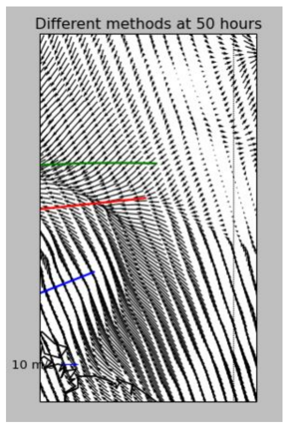
(a) at 10 hour.



(b) at 20 hour.



(c) at 30 hour.



(d) at 50 hour.

Figure 6.14: The solution from phase 3 is compared to different navigational methods at 10, 20, 30 and 50 hours. Blue is the genetic algorithm, red is the loxodrome method and green is the orthodrome method.

6.8 Phase 4

Previous results show that the algorithm will usually avoid the polar low, when given the chance. A particularly interesting application of the algorithm is how it will navigate inside of a polar low, or rather navigate out of it.

The following scenario is constructed so that the ship model starts in the polar low at time 0 and has to navigate out of the polar low towards Tromsø at destination (69,40N 18,56E).

This scenario presents a particularly dangerous environments for small maritime vessels. The weighting of maximum rolling angle will be set to very high and the weighting of distance to very low, yielding the following parameters:

Generations 150

Population 20

Weight: Max rolling angle 0.95

Weight: Distance 0.05

origin (38E, 72N)

destination (18.6E, 69.4N)

The resulting path was found:

Chromosome [[393, 166, True], [309, 166, True], [315, 211, True], [338, 199, True], [356, 213, True], [355, 185, True], [359, 184, True], [351, 154, True], [329, 166, True]]

At first glance this solution does not seem optimal. It passes quite close to the destination and then proceeds to make a big loop before finishing at the destination. This can be due to several factors, for instance insufficient evolution for the solution, the extreme weighting of the max rolling angle feature or a combination of both.

If we take a closer look at the point where the path is close to the destination it might shed some light on the choice of taking a loop. Figure 6.16 shows the path at 31 hours from origin. The final destination is highlighted.

Stopping the algorithm path at 31 hours and going straight to the destination marker seems like a reasonable choice for the algorithm. The coordinate at 31 hours is (19.2476 E, 69.4274 N) which translates to (331, 169) in grid coordinates. A modified path is constructed by stopping the solution path at 31 hours and then drawing a straight line to the destination. The real world coordinate at 31 hours is calculated to grid coordinates and substitutes the second node in the original solution, resulting in the following chromosome path:

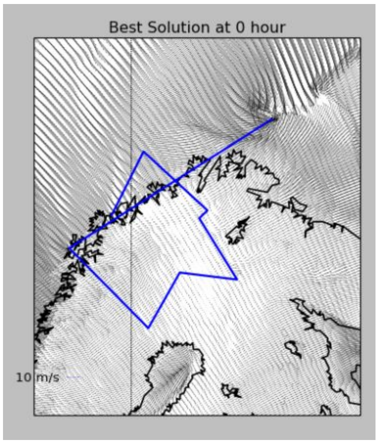


Figure 6.15: The solution of phase 4 is plotted. The weight of the max rolling angle fitness features is 0.95. The path origin is the eastern point.

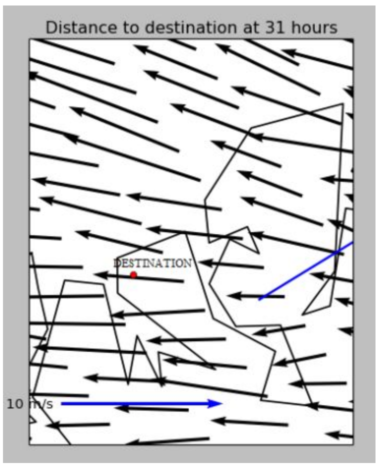


Figure 6.16: A closer look at the solution of phase 4, when it approaches the destination.

Modified Chromosome [[393, 166, True], [331, 169, True], [329, 166, True]]

This modified path is analyzed and compared to the genetic algorithm in the following table:

Method	Max Rolling Angle	Distance	Total Fitness
GA Solution	23.58 deg	2114.57 km	0.744
Modified Solution	180 deg	525.46 km	0.049

Table 6.6: A comparison of the GA solution proposed in phase 4, and a modified version of the the final solution.

The modified solution scores 180 degrees, which is the maximum rolling angle the ship can have, and thus scores a very low fitness value as well.

6.9 Phase 5

In the previous phases paths chosen by the genetic algorithm has been simple and straight lines. This may be because of polar lows being enormous and ship speed too slow in comparison to it. Because of this, maneuvers in correlation with the wind changes has small benefits. It is desired to investigate whether it is possible to find more intricate patterns for navigating a vessel. It is hypothised that a significantly higher speed for the maritime vessel will enable more complex path design. Phase 5 is conducted out of the polar low, the maritime vessel average speed is more than doubled and increased genetic parameters are applied. Weights are modified slightly to avoid loops while still retaining focus on maximum rolling angle. The following parameters were used:

Average ship speed 10 m/s = 19.43knot

Generations 500

Population 50

Weight: Max roll angle 0.9

Weight: Distance 0.1

Origin (36E, 72.6N)

Destination (18.6E, 69.5N)

The following solution was found after 500 generations:

Chromosome [[419, 164, True], [418, 165, True], [385, 180, True], [374, 182, True], [329, 166, True]]

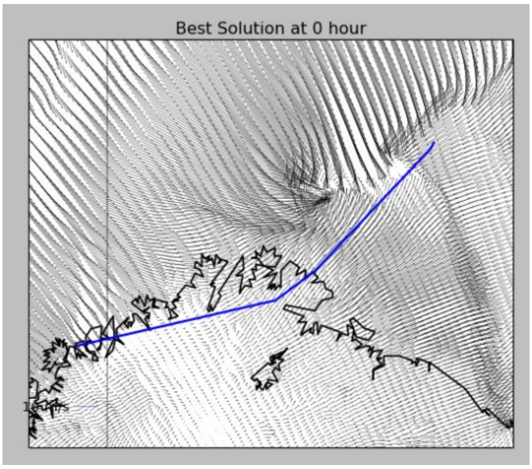


Figure 6.17: The final solution path for phase 5 is shown at 0 hour from origin. The origin is the eastern side of the path.

This genome contains more nodes than the other results. Running the algorithm for more generations and with a higher population lets the algorithm improve more and thus provides a smoother path, with more nodes than the previous results.

6.10 Phase 6

A longer run of escaping east from the eye of a polar low is included as the final result, with only origin and destination changed to:

Origin (30E, 72.5N)

Destination (45E, 70N)

Which yields this final solution:

Chromosome [[395, 156, True], [385, 173, True], [406, 190, True], [403, 196, True], [439, 209, True], [447, 213, True]]

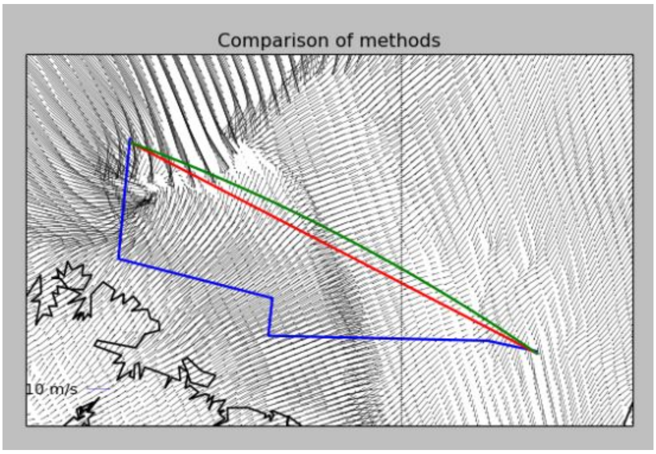


Figure 6.18: A comparison between the different methods are performed at the same route.Blue is the GA, red is the loxodrome method and green is the orthodrome method.

with the following statistics for the GA algorithm, loxodrome and orthodrome method:

Method	Max Rolling Angle	Avg Rolling Angle	Distance	Time Used	Fitness Function
Loxodrome	180 deg	14.099 deg	603.23 km	16.76h	0.099
Orthodrome	180 deg	18.59 deg	601.68 km	16.71 h	0.099
GA	20.89 deg	5.67 deg	795.19 km	24.31 h	0.871

Table 6.7: A quantification of the differences in fitness features, measured between the loxodrome, orthodrome and genetic algorithm in figure 6.18.

Both the loxodrome and orthodrome methods reach the highest possible rolling angle, which is 180 degrees, a fictional maxvalue since most boats will capsize before 45 degrees.

To see how the wind vectors affect the ship, a set of different timesteps is investigated in closer detail in figure 6.19

In figure 6.19a the genetic algorithm instantly aligns perfectly with the wind vector to minimize rolling angle. In figure 6.19b the algorithm proceeds to stay in the middle of the storm, where the wind is least, while aligning with the wind. It continues this behaviour in the next three figures, before it finally escapes the polar low in figure 6.19f.

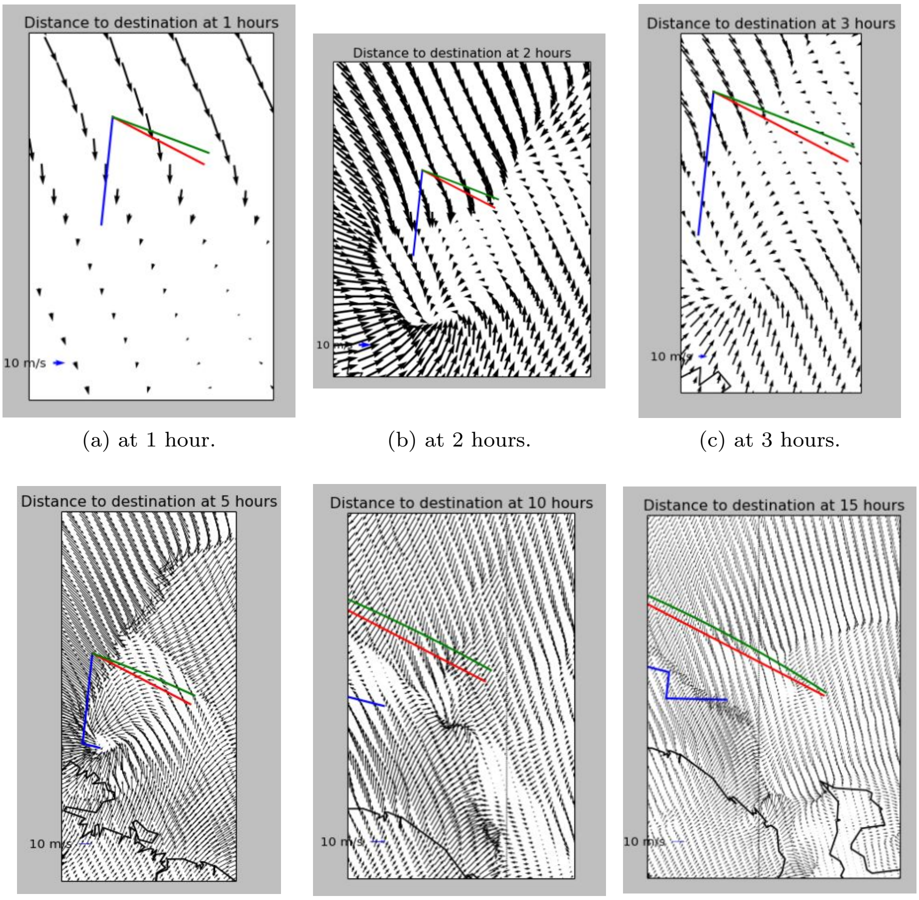
In all results both loxodrome and orthodrome methods resulted in higher maximum rolling angle than the genetic algorithm when measured.

Finally we present a comparison of the time used to run each of the phases of the algorithm, to see if it is a feasible tool. The run-time has to be sufficiently fast to allow for high interactivity with a users that want to try out several different plots. The phases with 150 generations and 20 population is measured against the longer run with 500 generations and 50 population, to see how much an increase in genetic parameters affect the performance of the algorithm:

	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6
Time used	920.61 s	564.33 s	394.77 s	2431.64 s	950.27 s	1052.71 s

Table 6.8: The performance of the different phases are compared.

Phase 5 and phase 6 was run on computer 2 as specified in the technical details. Phase 4 and 6 was navigating out of the eye of the polar low, and they had the longest run times, this indicates that intricate paths requires more computation time.



(a) at 1 hour. (b) at 2 hours. (c) at 3 hours.

(d) at 5 hours. (e) at 10 hours. (f) at 15 hours.

Figure 6.19: A comparison between the different methods are performed at the same route, at different timesteps.Blue is the GA, red is the loxodrome method and green is the orthodrome method.

Chapter 7

Discussion and Evaluation

To justify the results of a project there is a need to evaluate the findings and methods used. How a project is evaluated is dependent on the nature of the project. The author attempts to answer the research questions and discuss the results found in the previous chapters. Finally thorough evaluation is conducted in accordance to the criterias listed in Cohen and Howe [1988].

7.1 Research Questions

Each of the research questions defined is listed with a discussion on how they were addressed.

Research question 1 Can a genetic algorithm find an optimal path in a maritime environment ?

This research questions was constructed to figure out if it is possible to use the genetic algorithm method a maritime environment. Phase 1 carefully investigates the genetic algorithm process in the maritime environment and shows how it is possible for a genetic algorithm to approach an optimal path.

Research Question 2 How will the genetic algorithm compare to more traditional methods of path-planning?

Research question 2 was designed to compare the genetic algorithm to other methods that exists for navigating in maritime environment. The loxodrome and orthodrome methods were selected and defined in chapter 3. In phase 2,3 and

6 the genetic algorithm solution is compared to the loxodrome and orthodrome methods. The fitness features of the different methods are carefully analyzed, revealing that genetic algorithm can improve greatly upon the maximum rolling angle for a path, and as a result the safety of the maritime vessel.

Research Question 3 How will the genetic algorithm handle particularly interesting applications, such as navigating through a polar low, and navigating out of a polar low?

The main focus of the genetic algorithm is how wind forces act on a maritime vessel and how this affects safety of the vessel. Safety concerning the vessel is discussed in chapter 3. The theory section highlights that high speed and rapid changing winds is one of the greatest threats to safety of a vessel. Polar lows contain these both these hazards. Navigation around, through and out of a polar low is investigate in phase 1 through 6, in chapter 6.

7.2 Discussion

The findings are commented to a certain extent in chapter 6. In this section the research questions are answered and then interesting aspects of the algorithm are discussed.

7.2.1 Rolling Angles

The genetic algorithm can effectively reduce maximum rolling angle and average rolling angle of a maritime vessel for a given route. Table 6.2 and 6.3 in phase 2 shows that the genetic algorithm can reduce maximum rolling angle to 38.56% of loxodrome and 31.91% of orthodrome method. Table 6.4 and 6.5 in phase 3 shows that the genetic algorithm can reduce maximum rolling angle of loxodrome and orthodrome methods to 54.58% and 47.23% respectively, in high wind areas. Overall it was shown that the genetic algorithm can reduce the maximum rolling angle in comparison to loxodrome and orthodrome methods, which is an important factor for increasing the safety of maritime vessels.

7.2.2 Phase 4

In phase 4, figure 6.15, the final solution appears suboptimal. The final path passes the destination and proceeds to take a loop, before going towards the destination. A possible reason for this unexpected choice of path is the extreme weighting of maximum rolling angle used in the experiment. Another possible reason is limitations in precision. The algorithm can only take on geographical values described in the wind data file, which is an unwanted limitation. The choice of this type of limitation has been discussed, and was chosen because of a need for discretizing the search space. However, increasing the precision and possible values the algorithm can take on is an important point to improve in potential further work.

7.2.3 Selection Mechanisms

In late generations the fitness values are very similar(see figure 6.12) with small changes between the best solutions. Similar fitness values in combination with roulette-wheel selection causes the selection probability of all individuals to be similar. This means that the best solution has approximately the same chance of being chosen for reproduction as the close to best solution. This is low selection pressure and is undesirable, since we want the very best individuals to reproduce. Introducing tournament selection later stages of the generations could help solve the low selection pressure. Tournament selection chooses a group of individuals from the population and rank them within the group. The individuals are then given selection probability in correlation with their ranking within the group. Using a combination of these selection mechanisms could increase the selection pressure at all stages throughout the algorithm.

7.2.4 Reproducibility of Experiments

The random nature of the operations done in the algorithm on path solutions, makes it highly unlikely to reproduce the exact same result on two separate algorithm runs. However, running the algorithm with similar parameters, should result in somewhat similar results when they approach an optimal solution. The nature of genetic algorithm states that it has no guarantee of getting an optimal solution, so two different runs does not guarantee the same results. It should still be possible to analyse the results and compare them more traditional methods to determine the benefits of this system.

7.2.5 Performance

It is difficult to conclude anything from the time used for each run of the algorithm, shown in table 6.8. They were run on 2 different systems, where the system setup used for phase 5 and 6 were on a dedicated machine and used for the higher genetic parameter runs. The longer runs in phase 5 and 6 performed at similar speed as the rest of the experiments even though they had significantly higher genetic parameters. The time for each run of the algorithm is a bit high for enabling high interaction with a user. Possible ways to improve performance is through using dedicated machines, faster systems or improve the algorithm efficiency through algorithm analysis.

7.2.6 Utility of the algorithm

As we can see from the performance of the different phases, the algorithm requires quite some time and computational power to produce good results.Since many boats are old, with old hardware and minimal computing power, it is not feasible to run the algorithm on-board the ship. To let these users be able to benefit from the algorithm, it should be accessible through an online service, e.g. the Barentswatch portal. The user would input their preferences, and the all the computations related to the algorithm would be performed on the server side. The user would then be presented with their results when a criteria is met, e.g. a given time has elapsed, or a sufficiently good solution has been found. This allows users to benefit from the algorithm without the need for any computation on their side, they would only need a browser and a connection to the internet.

7.3 Limitations

In this section limitations to the algorithm is highlighted and discussed.

7.3.1 Wind Data

The wind data, and uncertainties of it, is one of the main limitations to the thesis. The algorithm needs as precise wind data, and high resolution, as possible. However, if the data has too high resolution and time steps are increased the algorithm performance will suffer greatly. This will result in slow performance and inability to provide fast navigational guidance, as intended. If the wind data can be simplified without losing precision and resolution the performance and quality of the algorithm will increase.

7.3.2 Terrain

Terrain was implemented as feasible and unfeasible grids in the genetic algorithm implementation, but all terrain was disregarded in results. The terrain was disregarded because the polar low was on top of terrain in the wind data, and the most interesting results were connected to navigation around and through a polar low. Another reason was lack of sufficiently good quality terrain data. This means that shallow water is disregarded as well.

7.3.3 Blendermann Method Limitations

The tests are performed in a wind tunnel scenario, and the method relies heavily on empirical data. When maritime vessels are observed in their natural environment, additional factors arise. For instance the effects of wind gradient, turbulence, efficiency of wind shelter and how wind affects a heavy tilted ship. Blendermann [1994] notes that in these cases the wind loading functions does not change, but wind load parameter values must adjusted.

7.3.4 Average Speed

To calculate the exact speed of a maritime vessel is not a trivial task. Speed is affected by wind, waves and currents, and these are subject to rapid changes. For simplification the average speed used is a static value, provided by the motor capabilities. In reality the speed of a maritime vessel will vary depending on the weather conditions, load on the vessel and navigation decisions.

Another limitation of having a constant average speed is that it makes it impossible to find some paths. The most intelligent solution to a problem could for instance consist of staying at one location until the wind blows over, and then move. Constant average speed also makes it impossible to navigational patterns that move at different speeds through areas, which could decrease wind forces on the vessel.

7.3.5 Traffic

That there are no crossing ship trajectories. If the path cross another vessel's trajectory the captain is expected to handle the necessary avoidance maneuver. The difference in path caused by such a maneuver is assumed to be negligible.

7.3.6 Weights

The algorithm could also be designed to find the pareto optimal front for an origin and destination, and then let the user select the most appropriate route from this set of solutions. Using the pareto-optimal front could be a useful alternative approach to the algorithm. This would require the algorithm to know the origin and destination beforehand.

7.3.7 Time Domain

The wind is measured at every whole hour. Any changes to the weather happening between the full hours is not registered.Finding the wind data between every hour is not easily dealt with, since it is very resource demanding to accumulate and work with wind data that is delivered at a smaller intervals. Weather can change quite drastically within an hour, so this is a weakness in the wind data. One way to deal with this is to approximate the intermediary wind values as a function of the wind data at every whole hour, however this representation of wind data would not sufficiently precise.

7.3.8 Format

The wind data has to be specified as a netCDF file, this is a standard used for most scientific data, especially weather forecasting data. The maritime vessel model has no standard file input. When making a new ship model for the algorithm every value that is used has to be inputted into the ship model class, which can be tedious. This can be solved by developing a standard for specifying maritime vessel models and an easy and intuitive way of input them.

7.3.9 On Board Sensors

A way to improve predictions of weather, advanced sensors can measure real time data more precise and compare them to the weather predictions to help the algorithm make better decisions. The genetic algorithm can work as a tool to complement the captains observations and experience, to make better decisions for navigation. Together, the genetic algorithm and the captains expert knowledge can make an even better path.

7.3.10 Scalability

At present when doing simulations the system performs at a decent rate. When increasing the resolution of the wind data and grid resolution, the algorithm will have to be run for more generations to develop a sufficiently good path. Computations will take longer because the chromosomes will be longer to deal with the increased resolution. This will cause a scalability issue where the algorithm will perform slower than it is doing at the moment. This could pose a problem for further work on the system. Improving upon the algorithm and running on a dedicated and faster setup is a necessary step towards addressing this problem.

7.3.11 Safety Hazard of Icing on Ships

An aspect that was investigated in regards to the design of the algorithm is icing on ships, and the safety hazards that this presents. IMO A749(18) reveals that to be able to measure the risk of icing and determining the dangerous areas along a route requires knowledge of temperature along this path as well, which was not available for this study. However, icing is a very important aspect of safety for maritime vessels and should be looked into for further work.

Ice formation adversely affects the seaworthiness of the vessel as ice formation leads to:

1. An increase in the weight of the vessel due to accumulation of ice on the vessel's surface. This causes a reduction in freeboard and buoyancy.

2. A rise of the vessel's centre of gravity due to high location of ice on the vessel's structures. This causes corresponding reduction in the level of stability.

3. An increase of windage area due to ice formation on the upper parts of the vessel and hence an increase in heeling moment due to the action of the wind.

4. A change of trim due to uneven distribution of ice along the vessel's length. Trim is when a ship is tilting forward or backwards.

5. The development of a constant list due to uneven distribution of ice across the breadth of the vessel. List is similar to trim, it acts on the sides of the ship, so list is when a ship is tilting to one of the sides due to added weight.

6. Impairment of the maneuverability and reduction of the speed of the vessel.

The main points gathered from IMO A749(18) were meteorological conditions causing the most comomon type of ice formation due to spraying of the vessel, where spraying is water transported to the vessel by wind. Slow accumulation of ice takes place when:

1. at ambient temperature from -1 celsius to -3 celsius and any wind force

2. at ambient temperature from -4 celsius and lower and wind force from 0 to 9 m/s

Very fast accumulation of ice takes place:

1. at ambient temperature of -4 celsius and lower and wind forces of 16m/s and over

2. at ambient temperature of -9 celsius and lower and wind force 10 to 15 m/s

7.4 Further Work

As this system is a first prototype, a part of the thesis objective has been to give an extensive overview over the intricacies and challenges presented in this project and suggestions for further work.

7.4.1 Higher Resolution Grid

Using wind data grid as the only viable positions that the maritime vessel can move between is restricting. Allowing the algorithm to take on more locations will allow for smoother movement and more complex movement patterns, but would also require significantly more computation power.

7.4.2 Terrain

To complement the increased resolution grid for movement and wind, a sufficiently high resolution terrain data should be included. This will allow the algorithm to find paths that can maneuver in low waters depending on the depth of the ship, instead of just out at the deep sea.

7.4.3 Plotting

In chapter 6 the figures are plotted using matplotlib. These figures had to be manually configured for every result to allow for decent illustrations of path and wind plotting.

When the plot is fully zoomed out the windquivers should be sufficiently small, so they don't clutter and block the contours of terrain that lies under them. At the same time they have to be sufficiently large so it is possible to see where there are strong winds and where the wind creates specific patterns, such as polar lows. On the other hand when zooming in the quivers should expand to get thicker so it is possible to see each individual wind quiver and study the data in greater detail.

This plotting function is a very delicate tuning process of visualization and zoom parameters, and automatization of this fine calibration was not a focus of the thesis. For further work, a clear improvement for ease of use and intuitiveness of the system would be to automate the visualization process. This would include always providing good plots that give a clear view of all relevant aspects of the solution.

7.4.4 Redundancy

Towards the end of the algorithm, a lot of identical paths are evaluated, which causes unnecessary work. For these cases a lookup table could be created during the algorithm, which stored the values for the most popular solutions, so they wouldn't have to be recalculated every time an already visited path was mutated into. Finding identical solutions in a lookup table, or similar solutions could save a lot of computation time, since most of the computation of the algorithm goes into computing the fitness and wind moments on a ship.

7.4.5 Evolve Time for Each Node

Earlier alternating speed for a ship was discussed. A solution to include alternating speed in the algorithm is to evolve time used for each segment of the path. Some limitations to how fast a passage could be traversed would have to be computed from the ship model.

This extra implementation would include the additional element of alternating speed, which could be an interesting aspect for the system, instead of the constant average speed of the maritime vessel, which is used in this implementation. Including alternating speed could introduce more complex calculations of fuel usage, time usage and speed of the path, and could give suggestions of when it is safe to give full speed and when the vessel should move slower or even stop.

7.4.6 Risk Curve

Punishing the rolling angle of a maritime vessel linearly from 0 to 90 degrees is unlikely the best solution. A vessel can have quite high rolling angles depending on ship characteristics, up to a certain point, without any threat to safety. Most ships can roll towards 30 degrees without danger, depending on their hull and vessel characteristics. Instead of using linear punishment of the rolling angles, a fitted sinus curve could be a better approximation. The sinus curve would threat low angles ranging from 0 to 10 with very low threat to safety, then punish angles higher than 10 degrees increasingly more.

7.4.7 Performance

A common nominator in the further work area is to increase performance, make the algorithm faster, get higher resolution and smoother movement for small scale navigation. A faster system can be achieved by changing to a faster programming language to control the system in more detail. The algorithm can be specialized for the system platform and more careful control of resources can optimize the performance of the algorithm. A faster system would allow for more and faster computations, which would allow for better user interaction as well.

7.5 Threats to Validity

Threats to validity are aspects with the results that can make them hard to reproduce or cause the results to not be valid.

7.5.1 Time

If time exceeds the time that there is recorded wind data for they will simply use the latest wind data that exists. These wind data will not be correct, and is a threat to validity. However, since this only occours at exceptionally long solutions these solutions will usually not be included in any final solution set, because they have too long distance.

7.5.2 Algorithm Performance

Time measurements done for the algorithms were performed on laptops with several other running applications and different work-loads at different times. Looking at the run-time of the algorithm at different phases they are very spread. For more precise performance calculations the algorithm should have been run in a dedicated environment.

7.5.3 Diversity

The algorithm was only tested on one type of wind data and one type of ship model, which may introduce bias. Further work should include extensive testing with different ship models and wind data, to verify the value of the program.

7.6 Evaluation

The model presented in Cohen and Howe [1988] is used to justify the choice of project and has served as a guidance throughout structuring and conducting the thesis. The model consists of five steps:

1. refine the topic to a task
2. design the method
3. build a program

4. design experiments
5. analyze the experiments and results

The five steps will be summarized in the following sections:

7.6.1 Refine the Topic to a Task

To refine the problem, a set of research questions was defined. Previous definitions of the problem and how they were handled is discussed in chapter 2, the most relevant one being MEWRA. The differences between these methods is highlighted and a justification for creating a new algorithm consisting of arctic wind data and using a representative ship model for this region is argumented for. The task is simplified by abstracting away some aspects listed in the scope.

7.6.2 Design the Method

The second step naturally relies on the first step, and addresses the research questions with the necessary theory and methods found in chapter 2 . Argumentations for methods are provided.

7.6.3 Build a Program

A detailed implementation of the genetic algorithm is provided in chapter 4. This describes how the solutions are produced and how they are represented. This section also discusses any unforeseen problems encountered and how these were handled.

7.6.4 Design Experiments

A detailed experiment setup and plan is provided in the beginning of chapter 6. These experiments are performed to demonstrate how well the program works. The experiments are designed to be compared against traditional navigation methods, and validity of these experiments rely on identical conditions for all methods, which is provided.

7.6.5 Analyze the Experiments and Results

The experiment results are shown in chapter 6. The results includes all the different phases and scenarios from the experiments plan. The results are shown together with comments meant to guide the reader through the experiments. Further in this chapter, the results are discussed, comparisons are made to other methods and limitations of the program is highlighted. Finally, the performance of the program is considered.

Chapter 8

Conclusion

The main goal for this thesis has been to implement a path planning system for maritime navigation in arctic regions. The author has implemented a fully functional genetic algorithm that processes wind data in the arctic region, and creates an optimal path given user input.

The system is highly modular and highly configurable, which allows the user to specify which parameters he wishes to focus on, in relation to wind effects on maritime vessels.

Execution of extensive simulation runs in the arctic region shows that the system can find paths that are hard to discover by using traditional methods of pathplanning, due to the added dimension of time-domain simulation and wind forces on the maritime vessel.

Besides the delivered system, a list of potential improvements and limitations of the algorithm has been discussed, which may be implemented in future projects.

8.1 Generalization

By substituting the ship model with a model of any vessel, the problem can be generalized as reducing side-force moment on any type of vessel, and could be used in any vessel problem that seeks to minimize length of a path and side-force moment on their vessel.

8.2 Contributions To The Research Field

The author's previous specialization project uncovered that using artificial intelligence methods in combination with maritime path-planning is a relatively untested area of path-planning. Most of the literature is focused on autonomous collision-avoidance, cargo optimization and traffic optimization.

In light of the results found in this thesis, the genetic algorithm has been successfully applied as a multi-objective evolutionary algorithm for weather routeing in extreme weather for the arctic regions.

Bibliography

Alvarez, A., Caiti, A., and Onken, R. (2004). Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *Oceanic Engineering, IEEE Journal of*, 29(2):418–429.

Barraquand, J., Langlois, B., and Latombe, J.-C. (1991). Numerical potential field techniques for robot path planning. In *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 1012–1017 vol.2.

Blendermann, W. (1994). Parameter identification of wind loads on ships. *Journal of Wind Engineering and Industrial Aerodynamics*, 51(3):339–351.

Chen, D., Szczerba, R., and Uhran, J. J. (1995). Planning conditional shortest paths through an unknown environment: a framed-quadtree approach. In *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 3, pages 33–38 vol.3.

Cohen, P. R. and Howe, A. E. (1988). How evaluation guides ai research: The message still counts more than the medium. *AI Magazine*, 9(4):35.

Dubrovsky, O., Levitin, G., and Penn, M. (2002). A genetic algorithm with a compact solution encoding for the container ship stowage problem. *Journal of Heuristics*, 8(6):585–599.

Fiskeridirektoratet (2014). Norwegian fishing vessels fishermen and licenses 2013. Technical report, Fiskeridirektoratet.

Floreano, D. and Mattiussi, C. (2008). *Bio-inspired artificial intelligence: theories, methods, and technologies.* MIT press.

Fossen, T. I. (2011). *Handbook of marine craft hydrodynamics and motion control.* John Wiley & Sons.

Garau, B., Alvarez, A., and Oliver, G. (2005). Path planning of autonomous underwater vehicles in current fields with complex spatial variability: an a* approach. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 194–198.

Holland, J. H. (1975). Adaption in natural and artificial systems.

Hollnagel, E. (1996). Reliability analysis and operator modelling. *Reliability Engineering and System Safety*, 52(3):327–337.

Hornby, G. S., Globus, A., Linden, D. S., and Lohn, J. D. (2006). Automated antenna design with evolutionary algorithms. In *AIAA Space*, pages 19–21.

Hromkovic, J. (2010). *Algorithmics for hard problems: introduction to combinatorial optimization, randomization, approximation, and heuristics*. Springer-Verlag.

Ito, M., Zhnng, F., and Yoshida, N. (1999). Collision avoidance control of ship with genetic algorithm. In *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*, volume 2, pages 1791–1796 vol. 2.

James, R. W. (1957). Application of wave forecasts to marine navigation.

Janikow, C. Z. and Michalewicz, Z. (1991). An experimental comparison of binary and floating point representations in genetic algorithms. In *ICGA*, pages 31–36.

Jarvis, R. (2006). Robot path planning: complexity, flexibility and application scope. In *Proceedings of the 2006 international symposium on Practical cognitive agents and robots*, pages 3–14. ACM.

Kamon, I. and Rivlin, E. (1995). Sensory based motion planning with global proofs. In *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 2, pages 435–440 vol.2.

Kobayashi, E., Asajima, T., and Sueyoshi, N. (2011). Advanced navigation route optimization for an oceangoing vessel. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 5(3):377–383.

Kobyliński, L. (2007). System and risk approach to ship safety, with special emphasis of stability. *Archives of Civil and Mechanical Engineering*, 7(4):97–106.

Krata, P. (2008). Total losses of fishing vessels due to the insufficient stability. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 2(3):311–315.

Kwon, K.-Y., Cho, J., Yoo, B.-S., and Joh, J. (2005). Autonomous navigation of an underwater robot using fuzzy logic. In *Fuzzy Information Processing Society, 2005. NAFIPS 2005. Annual Meeting of the North American*, pages 540–545. IEEE.

Lacki, M. (2008). Reinforcement learning in ship handling. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 2(2):157–160.

Manley, J. (2008). Unmanned surface vehicles, 15 years of development. In *OCEANS 2008*, pages 1–4.

Norrbin, N. H. (1971). Theory and observations on the use of a mathematical model for ship manoeuvring in deep and confined waters. Technical report, DTIC Document.

Petkovi, D. (2011). Dynamic programming algorithm vs. genetic algorithm: Which is faster? In Bramer, M., Petridis, M., and Hopgood, A., editors, *Research and Development in Intelligent Systems XXVII*, pages 483–488. Springer London.

Rasmussen, E. (1979). The polar low as an extratropical cisk disturbance. *Quarterly Journal of the Royal Meteorological Society*, 105(445):531–549.

Rechenberg, I. (1965). Cybernetic solution path of an experimental problem.

Roberge, V., Tarbouchi, M., and Labonté, G. (2013). Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. *Industrial Informatics, IEEE Transactions on*, 9(1):132–141.

Sinnott, R. W. (1984). Virtues of the haversine. *Sky and telescope*, 68:158.

Smierzchalski, R. (1999). Evolutionary trajectory planning of ships in navigation traffic areas. *Journal of Marine Science and Technology*, 4(1):1–6.

Szlapczynska, J. and Smierzchalski, R. (2009). Multicriteria optimisation in weather routing. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 3(4):393–400.

Vasudevan, C. and Ganesan, K. (1996). Case-based path planning for autonomous underwater vehicles. *Autonomous Robots*, 3(2-3):79–89.

Weise, T. (2009). Global optimization algorithms–theory and application. *Self-Published,*.

Zheng, C., Li, L., Xu, F., Sun, F., and Ding, M. (2005). Evolutionary route planner for unmanned air vehicles. *Robotics, IEEE Transactions on*, 21(4):609–620.

Appendix A: Ship model