Summary

There are multiple Electronic Navigational Charts (ENC) with detailed information about the maritime environment available today. These maps are made by Hydrographic Offices, and they follow a standardized format that makes it possible to unify the transfer of maritime map data. In this thesis I explore options for using this information for navigational purposes for an Autonomous Surface Vessel (ASV).

The thesis is a contribution to a Collision Avoidance System (CAS) that was formally carried out by Johansen [16] and later implemented by Hagen [11]. An application extracting hazards from an ENC is implemented. Additionally, a real-time anti-grounding system that uses the extracted information is implemented as a subsystem of previous implementations. Different situations of the ASV are simulated, with and without the anti-grounding system and the resulting system behaviour is compared and analyzed. The anti-grounding system is implemented using ROS as a framework and utilizes GDAL/OGR to access the data of the ENC.

The simulations demonstrated both predicted behaviour, as well as unexpected behaviour from the anti-grounding system. The anti-grounding system made the ASV avoid hazards based on the extracted hazard map. A method of grounding avoidance and the hazard representation showed promise with respect to real-time constraints and ability to detect hazards. However, the implementations are not flawless and improvements are necessary for future applications.

Oppsummering

Det finnes mange Elektroniske Navigasjonskart med detaljert beskrivelse om det maritime miljøet tilgjengelig idag. Disse kartene er laget av hydrografiske kontorer, hvor de følger et standardisert format som gjør det mulig ådele maritime kart. I denne oppgaven utforsker jeg mulighetene for åbruke elektroniske kart for navigatoriske formål for et autonomt overflatefartøy.

Denne oppgaven er et bidrag til et anti-kollisjonssystem som er spesifisert formelt sett av Johansen [16] og senere implementert av Hagen [11]. En applikasjon som ekstraherer farer fra et elektronisk kart er implementert. I tillegg er et sanntidssystem som bruker de ekstraherte farene ogsåimplementert som et undersystem av tidligere implementeringer. Et autonomt overflatefartøy er simulert i forkjellige situasjoner, med og uten anti-grunnstøtningssystem, og resulterende oppførsel er sammenlignet og analysert. Anti-grunnstøtningssystemet er implementert med bruk av ROS som rammeverk og bruker GDAL/OGR for åaksessere den elektroniske kartdataen.

Simuleringene demonstrerte både forventet og uforventet oppførsel av anti - grunnstøtningssystemet. Anti-grunnstøtningssystemet påvirket det autonome overflatefartøyet slik at det unngikk de ekstraherte farene. En metode for åunngågrunnstøtning og representasjonen av farer viste lovende resultater med hensyn til sanntidsbegrensninger og evne til åoppdage farer. Derimot, såer ikke implementasjonen feilfri og forbedringer er nødvendig for fremtidige applikasjoner.

Preface

This thesis is my final work for the Master of Science degree in Engineering Cybernetics at NTNU. The topic of this thesis was kindly provided by Prof. Tor Arne Johansen, and I would like to thank him for giving me the opportunity to contribute to such an interesting project. I would also like to thank Dr. Giorgio D. Kufoalor for his guidance throughout this period. His thoughts have been invaluable and immensely helpful. I am also grateful for being a part of an inspiring community at NTNU Gloeshaugen and thankful for my fellow students for five great years. Last, but not least, I want to thank my family for their love and support.

Table of Contents

List of Tables

List of Figures

Abbreviations

Abbreviations

API Application Programming Interface

ASV Autonomous Surface Vessel

CAS Collision Avoidance System

COLREGS Convention on the International Regulations for Preventing Collisions at Sea

DE-9IM Dimensionally Extended nine-Intersection Model

ECDIS Electronic Chart Display and Information System

ENC Electronic Navigational Chart

ESRI Environmental Systems Research Institute

GIS Geographical Information System

GVD Generalized Voronoi Diagram

IHO International Hydrographic Organization

IMO International Maritime Organization

MPC Model Predictive Control

NIMA National Imagery and Mapping Agency

ROS Robot Operating System

WGS84 World Geodetic System 1984

Chapter 1

Introduction

The basic nature of every living organism is to survive in its environment. Through our senses we are able to observe and make decisions that minimize chances of hazardous situations. This sounds as a simple task, but is in fact very complex. There are countless factors and details of the environment that must be sensed and processed, and where time and information are in short supply. This complexity becomes prominent in the development of mobile autonomous vessels that are able to handle its environment.

Collision Avoidance System (CAS) is a necessity of developing an autonomous maritime vessel. An appropriate CAS must evaluate the situation correctly, and make maneuvering decisions that avoid and reduce the risk of collision. A CAS is a computer assisted navigation system and will add consistency to the vessels maneuvering behaviour. This would plausibly reduce accidents caused by human errors, which are the biggest contributor to maritime accidents, and improve the safety of both the vehicle and human. In addition to reducing accidents, a reduction of operational cost also motivates the development of CAS.

In addition to avoiding moving obstacles, a CAS should also consider all environmental obstructions to avoid collision with static obstructions. Traditionally this has been done by relying on navigation paper charts and the mariners ability to act correctly. Electronic Navigation Charts (ENC) have widely replaced the traditional paper charts today. ENC contain detailed information about the maritime environment and are mainly displayed to, and used by, navigators. But the usage of ENC could also be extended for autonomous navigation purposes.

The avoidance of static obstructions can be separated into two categories. The first category is by mission planning. Consider an Autonomous Surface Vessel (ASV) that is destined to get from A to B. Before this mission is executed, the ASV must know a path in the environment that does not lead to a grounding. A mission planner consider the environment described by the ENC, and provide a grounding free route from A to B. In this way is the ASV able avoid groundings based on the information of the ENC.

The second category is a real-time anti-grounding system. This is necessary in case of the ASV is forced to leaves its planned, grounding free, path. Such situations may be provoked by an unknown, moving obstacle that interferes with the ASVs motion. In such case, the ASV may have to take a collision avoidance maneuver and leave its planned path. Once the ASV leave its path, that is known to be collision free, it is exposed of environmental hazards. An appropriate anti-grounding system must prevent the ASV to enter hazardous areas with the risk of grounding in such situations.

1.1 Problem Formulation

This master thesis will primarily address the second category of problems by building on the MPC-based CAS for marine vessels proposed by Johansen et al. [16] and implemented by Hagen et al. [11][25]. The thesis will study different ways mapped hazards can be used in the CAS.

The CAS assumes that mapped hazards from an electronic map are provided. Therefore, the goal of this master thesis is to develop an application that extracts information about mapped hazards from ENC provided by Kartverket (`https://www.kartverket.no/data`). The information extracted from ENC will be translated into a suitable input for the CAS, and the outcome will include:

- A representation of mapped static obstacles indicating location, size, shape, uncertainty, etc.
- No-go zones for generating anti-grounding constraints based on information about land, depth data, and navigational aids.

The master thesis will consider the automatic generation of the above hazards within both 1) an absolute coordinate framework, suitable for automatic route planning and presentation to an operator, and 2) a coordinate framework relative to the position of the ASV, suitable for automatic real-time decision-making in the ASV's local, and dynamic, environment. The result will be a C++ and/or Python software module, which can be tested using a ROS simulation environment and the existing implementation of the MPC-based collision avoidance system.

Depending on the availability of the necessary resources, full-scale experiments can be carried out to verify the software module developed in the thesis.

1.2 Relevant previous work

Article [16] and master thesis [11] form the basis of this master thesis. The work of these two background papers proposes a formal solution to the collision avoidance problem for an ASV, as well as the implementation of this solution. Main aspects and principles that are relevant for an expansion of the CAS with an anti-grounding system, will be presented in Chapter 2.

In an article by Mąka [19] the problem of using data from an ENC for applications is discussed. Selection and transformation of data are subjected, and two different algorithms for creating representations of the environment are presented. The article also presents a suggestion of choosing problem space for solving path planning problems. The suggestion is to choose a rectangular problem space surrounding the ship. The argue for this choice is that optimization problems often require a rectangular problem space.

Methods of data access and translation of ENC data are presented by Carignan et al. [4]. ENC data of S-57 format is first translated to ESRI shapefile format. ArcGIS (`https://www.arcgis.com`) is then used to display and access the information of the ENC. The data set is manually edited from this Geographical Information System (GIS) interface. The article provides a summary of the data sources and methodology for developing a digital elevation model of Central Oregon Coast, which is used for tsunami inundation modeling, as a part of offline tsunami forecast system.

A brief introduction to creation of S-57 formatted ENC is presented by Austin [2]. The article covers basic conventions to follow, based on the S-57 standard (see Section 2.1.1) that is published by the International Hydrographic Organization (IHO). This provides an understanding of the construction of an ENC.

Few studies emphasize the opportunity of using ENC in autonomous operations. However the book *Electronic Chart: potential and limitations of a new marine navigation system* [13] covers ENC and their potential. It discusses ENC in the perspective of usage in an Electronic Chart Display and Information System (ECDIS). The highest level of autonomization mentioned in the book is ECDIS with the potential of alerting an operator of an incoming hazardous situation.

Path planning and collision avoidance methods are widely researched subjects. There are several different methods presented, e.g. A* [12], Velocity Obstacle [8], Artificial Potential Field [21], and applied to a range of different applications. The book [18] concerns the field of path planning and presents decision-theoretic ideas. A collection of different path planning and collision avoidance methods are presented later in Chapter 2.

1.3 Outline

Chapter 2 presents topics and theoretical principles of relevance for an implementation of the anti-grounding system. Chapter 3 concerns the software choices and the implementations of this thesis. The results of a simulation study are presented in Chapter 4 and a more extensive discussion of these results are discussed in Chapter 5. Finally, a conclusion is given in Chapter 6, along with some future prospects.

The Appendix consists of a selection of rules from Convention on the International Regulations for Preventing Collisions at Sea (COLREGS) that are relevant to CAS assisted maneuvering of ships.

Chapter 2

Theoretical Background

This chapter will introduce topics and theoretical principles that are of relevance to the implementation of an anti-grounding system. These constitute the foundation of the implementation, and support the analysis and discussion of the anti-grounding system. Some of the content in this chapter constitute implicit background theories for the work done throughout the thesis and inform various aspects of implementing the anti-grounding system. On other occasions, the theoretical material is made explicit in the analysis and discussion.

2.1 Electronic Navigational Charts

ENC is a descriptive database created and published by a national hydrographic office. The ENC contains all necessary data for safe navigation. ENC require a certification by complying with the S-57 Standard specified by the IHO. This is published and intended to be used on an ECDIS. ECDIS is an installation on a ship to display charts to the pilot and is compliant with International Maritime Organization (IMO) regulations. This is a replacement of more traditional paper charts used for navigational purposes.

2.1.1 IHO S-57

IHO is an intergovernmental consultative and technical organization that was established in 1921. The purpose of the organization is to support safety of navigation and protection of the marine environment. IHO investigates and proposes solutions to several topics and matters that are of interest to navigation at sea. One goal is to ensure the greatest possible uniformity in nautical charts and documents. To accomplish this, IHO has published the documentation "*Transfer Standard for Digital Hydrographic Data*" [15]. The proposed standard was published in 1992 and is maintained and revised by IHO. The standard has been given the code name S-57, and specifies how hydrographic offices should construct ENC. The specifications of the standard will ensure that the ENC contains all the chart information that is necessary for safe navigation. Kartverket is the Norwegian

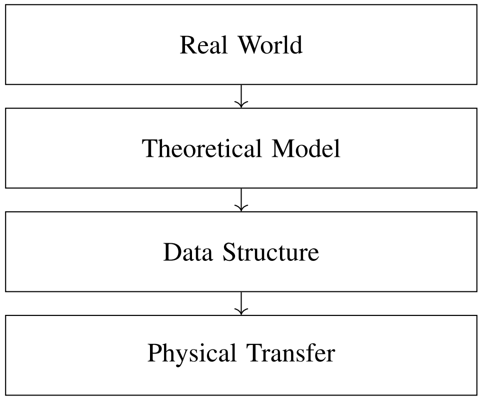

Figure 2.1: Transfer layers of ENC data according S-57.

institution responsible of maintaining the international ENC for the Norwegian maritime environment. The ENC produced by Kartverket is in accordance with S-57 standard.

2.1.2 S-57 Structure

S-57 separates the data transfer into four different layers, as shown in Figure 2.1. The real world layer represent the physical space we want to quantify, measure and model. It is not practical, nor beneficial, to describe the real world precisely due to its great complexity. There is a need to investigate what information is necessary for navigational purposes. The next layer is therefore what is called a model layer. The purpose of this layer is to model the real world to create a simplified view of the real world. The model is further structured into a data structure specified by S-57. The outcome of the data structure layer is a single file that is ready to be transferred and distributed. The transfer is represented by the physical transfer layer.

Model layer

The model specified in S-57 concerns entities in the real world that are relevant for hydrographic purposes. The standard defines an entity of interest as an object owning a spatial object, and one or more feature objects. The feature objects contain descriptive attributes, and do not contain geometry. The features are defined as four different types of feature objects; Meta, Cartographic, Geo and Collection. All these features are non-locational objects, and are meant to provide additional information about a real world entity.

The spatial object defines the geometry and its geographic position. The standard limits the representations of spatial characteristics to be of either vector-, raster- or matrix type. However, the current standard has yet to define raster and matrix representation, and thus covers only vector representation. A vector model is defined in a two dimensional planar view and is implemented as nodes, edges and faces. The third dimension, like elevation of depth data, is represented as a related non-spatial descriptive feature object.

Data Structure

S-57 describe the rules of translating the simplified theoretical model into named constructs, ie. records and fields. Definitions of rules and constraints for the constructs and their content is what makes up the data structure. The S-57 describes these rules comprehensively to create unified data structures for representing electronic map data. This section is of interest for agencies creating ENC, and software developers who interprets the data sets. Since this thesis primarily makes use of software API (Application Programming Interface) that is developed to interpret such data sets, this is not covered in detail.

2.1.3 Naming Convention

The standard specifies the naming of a data set file. From IHO's S-57 publication [15], following name convention is specified:

```
CCPXXXXX.EEE
| | |     |
| | |     |---- EEE   = Update Number
| | |--------- XXXXX = Individual Cell Code
| |----------- P     = Navigational Purpose
|------------- CC    = Producer Code
```

- The first two characters identify the producer. This list is given in Annex A to Appendix A (IHO Object Catalogue).
- The third character indicates the navigational purpose (see clause 2.1).
- The fourth to eight characters are used for the cell code. This code can be used in any way by the producer to provide the unique file name. If characters other than numbers are used only uppercase letters are allowed.

Navigational purpose code:

Subfield Content	Navigational Purpose
1	Overview
2	General
3	Coastal
4	Approach
5	Harbour
6	Berthing

Kartverket has a unofficial document that specifies the naming convention they use. This information is retrieved by a mail exchange with Kartverket. The document is approximately twenty years old and may contain obsolete information. However, it indicates to an extent the naming convention Kartverket uses on their ENC. The following information is a brief overview of the naming convention and is retrieved from the unofficial document.

Norwegian ENC will always be identified with *NO* as producer code. The five cell codes are divided into three segments. The first code number is a letter and specifies the size of the cell. The following two code numbers specifies the latitude position of the ENC. The last two code numbers specifies longitude of the ENC. The cell representation has its origin at $(55°N, 0°E)$.

Information about naming convention is not directly important regarding methods of extracting information from ENC, but rather a nice-to-know information for the developer using ENC.

2.1.4 Mission Specific Assessment of S-57 information

The S-57 standard is comprehensive and very descriptive. The standard describes every detail of the real world that can be of navigational interest. This makes the list of entities quite large. A complete list of objects that may appear in a ENC can be found in Appendix A of the S-57 publication.

The mission of a ship is often the factor that decides which entities that are important. For example, when the mission of a ship is to safely dock at a harbour you should take into account harbour specific information. Harbour specific information is not important when the ship convoys at open waters. This implies that an assessment and selection of relevant features are necessary when developing mission specific systems. Table 2.1 is a simple example of features to be considered for a vessel to safely navigate on open sea water relatively close to coast line. The list accounts for most physical objects that may occur in such a scenario. More specifically the list contains coast line, water depth, beacons and buoys.

2.2 Collision Avoidance System

The success of autonomous ship navigation depends on the development of efficient real-time intelligent algorithms of a CAS [24]. A CAS assesses the situation of a ship through its sensor system and take maneuvering decisions to avoid collisions. Most of the CAS imitates human piloting, with the computer controlled system mostly add consistency to the navigation. The development is mainly motivated by an increase in vehicle safety and reduction of operational expenditure. Typically is a CAS based on two different approaches:

- *Mathematical models and algorithms.* Algorithms are utilized to decide the maneuvering of the vessel based on mathematical description of the vessels dynamic and environment.

- *Soft computing - Evolutionary algorithms, neural networks and fuzzy logic.* Collision problem is solved by artificial intelligence; neural networks with learning capabilities.

Johansen et al. [16] proposes a solution to the collision avoidance problem, and this is the subject of this thesis. It is a mathematical model and algorithm based solution. The system output two different control variables; guidance course angle (χ_{CA}), and propulsion command (P_{CA}). Originally the vessel follows a predefined planned path. The control variables CAS outputs are considered as offsets to the planned path. Figure 2.2 shows a schematic system overview of the CAS implementation. The calculation of the offsets

Acronym	Object	Description
BCNISD	Isolated Danger Beacon	A beacon erected on an isolated danger of limited extent, which has navigable water all around it.
BCNLAT	Lateral Beacon	Used to indicate the port or starboard hand side of the route to be followed.
BOYCAR	Cardinal Buoy	Used in conjunction with the compass to indicate where the mariner may find the best navigable water.
BOYINB	Installation Buoy	Used for loading tankers with gas or oil.
BOYISD	Isolated Danger Buoy	A buoy moored on or above an isolated danger of limited extent, which has navigable water all around it.
BOYLAT	Lateral Buoy	Used to indicate the port or starboard hand side of the route to be followed.
BOYSAW	Safe Water Buoy	Used to indicate that there is navigable water around the mark.
BOYSPP	Special Purpose Buoy	Primarily used to indicate an area or feature, the nature of which is apparent from reference to a chart, Sailing Directions or Notices to Mariners.
COALNE	Coastline	The line where shore and water meet
DEPARE	Depth Area	Area of water within a defined range of values
DEPCNT	Depth Contour	A line connecting points of equal water depth

Table 2.1: A selection of objects from the S-57 object catalogue. *Acronym* is how the name of the object in the data set, *object* is the full text name of the object and *description* specify further details about the object.

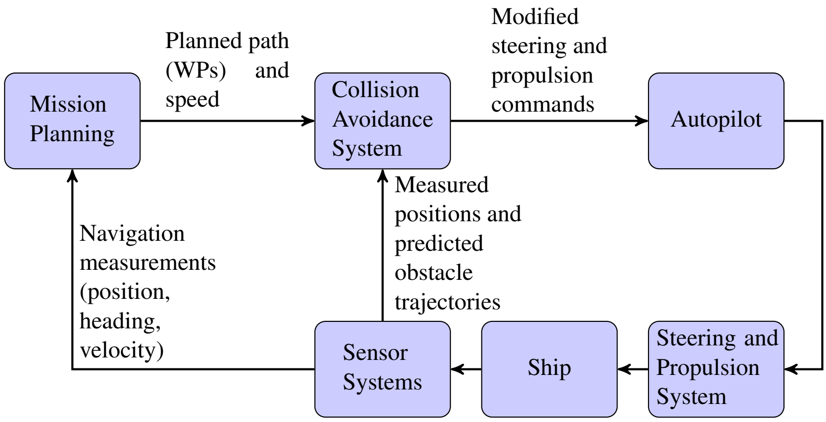

Figure 2.2: System overview of a navigation assisted ship with a collision avoidance system.

is simulation-based and utilize Model Predictive Control (MPC). The system predicts the state over a finite horizon of control inputs and choose the offsets by minimizing a cost function.

The changes are applied to the mission path keeping command and are considered as offsets to the planned path. Originally the vessel follow a predefined, planned path, that is optimal and do not intersect with hazardous areas. In case of an obstacle interfering with the planned path of the vessel, the vessel has to maneuver to avoid collision with the obstacle, and the ship might deviate from the planned path. At open sea with sufficiently deep water, this is not a problem. But when the ship navigates in areas with hazardous static obstacles, it is necessary make sure that the offset does not lead to collision with static hazardous obstacles.

Any practical implementation of a CAS must also comply in accordance with marine traffic regulations. Johansen emphasizes this and the method is in accordance with COLREGS [14]. These are navigation rules for marine vessels to prevent collision. Appendix A shows a selection of COLREGS that is of relevance to a CAS implementation and to this thesis.

Furthermore, an ASV simulator was developed by Thomas Stenersen in 2015 [25] and a simulation based MPC algorithm was developed by Inger Berge Hagen 2016 [11].

2.2.1 Model Predictive Control

MPC is an optimization control method that minimizes a cost function under constraint given by a dynamical model of the controlled system. The method predicts the behaviour of a system for a set of control inputs with a given prediction horizon, and chooses the most suitable control input. The most suitable control input is chosen by minimizing a cost function, and is applied to the system. Figure 2.3 shows a prediction of one control sequence, where the notation T and h are the horizon and sampling time respectively.

The method is iterative, and might require a lot of computational power. This is dependent on the number of control sequences to iterate through, and the complexity of the

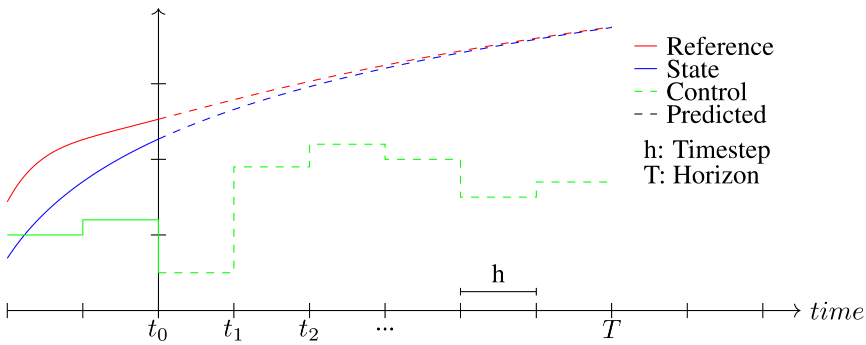

Figure 2.3: Illustration of a MPC prediction for one control input sequence. The illustration is inspired by sketch made in [23].

cost calculation. The computational power consumption of the MPC might be an issue for a real-time application. From the perspective of any real-time application, it is required that the MPC has solved the optimization problem and found the best control input within the time sampling time h [23].

A reduction of possible control inputs reduce the computational power consumption of the MPC. Such reduction may be beneficial for a real-time application. This is suggested by Johansen et al. and is implemented by Hagen. A simplified overview of the iteration loops of the MPC implemented is shown in Figure 3.3, where P_ca and χ_ca are the set of control variables to examine.

2.2.2 Hazard Assessment

The formalized cost function to be minimized is shown in Equation 2.1. This function penalizes collision related risk, change in nominal speed, change in course and risk of grounding.

$$H^k(t_0) = \max_i \max_{t \in D(t_0)} (C_i^k(t) R_i^k(t) + k_i \mu_i^k(t)) + f(P^k, \chi_{ca}^k) + g(P^k, \chi_{ca}^k) \tag{2.1}$$

$$C_i^k(t) = K_i^{coll} |\vec{v}_0^k(t) - \vec{v}_0^k(t)|^2 \tag{2.2}$$

$$R_i^k(t) = \begin{cases} \frac{1}{|t-t_0|^p} \left(\frac{d_i^{safe}}{d_{0,i}^k(t)}\right)^q, & \text{if } d_{0,i}^k(t) \le d_i^{safe} \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

$$f(P, \delta) = k_P(1-P) + k_\chi \chi_{ca}^2 + \Delta_P(P - P_{last}) + \Delta_\chi(\chi_{ca} - \chi_{ca,last}) \tag{2.4}$$

$$D(t_0) = t_0, t_0 + T_s, ..., t_0 + T \tag{2.5}$$

where:

$H^k(t_0)$	= Cost function
$C_i^k(t)$	= Collision associated cost
$R_i^k(t)$	= Collision risk factor
$f(P, \delta)$	= Nominimal speed and course keeping cost
$g(P^k, \chi_{ca}^k)$	= Grounding associated cost
T	= Prediction horizon
T_s	= Discretization interval
k_i	= Tuning parameters
$\mu_i^k(t)$	= Binary indicator for violation of COLREGS

The first term evaluates all situations along a predicted trajectory, and returns the cost of the situation that gives the highest cost. The situation giving highest cost corresponds to the most hazardous situation along the predicted trajectory. Maximization problem is constructed by the three following factors. $C_i^k(t)$ is a collision associated cost. The expression of this is shown in Equation 2.2. This is multiplied with a collision risk factor, $R_i^k(t)$. Equation 2.3 shows the expression of the collision risk factor. Including to this product, a the term $k_i\mu_i^k(t)$ is added to punish violation of COLREGS.

The second term punishes changes in nominal speed and course offset. This is to keep a predictable behaviour of the vessel. A reduction of risk must be significant large before the CAS choose to change offset controls. The vessel tries to follow a straight line, with constant speed, if possible.

The last term is a grounding penalty and is said to be defined based on electronic map data and ship sensor data. This term is missing from the implementation of Hagen [11], and is suggested as future work. The problem formulation of this thesis is directly addressing the unspecified grounding term.

2.3 Geodesy

Geodesy is the science of accurately measuring properties of the Earth, including its geometric shape, orientation in space and gravitational field. A global position on Earth is measured in a coordinate system associated with a geodetic datum. A geodetic datum is reference points that a coordinate system relies on and refers to. Datums are used to translate map positions to the real position on Earth. There are several different datum existing today, each one tries to represent the Earth as accurately as possible. WGS84, EGM2008 are two examples on global datum, and OSGB36 is an example of a local datum.

2.3.1 World Geodetic System 1984

World Geodetic System 1984 (WGS84) is a global geodetic datum. It is released and revised by National Imagery and Mapping Agency (NIMA). The datum represents the Earth as a geocentric oblate spheroid. The main parameters regarding the spatial geometry of WGS84 are shown in Table 2.2. WGS84 is commonly used as a standard for georeference. Kartverket uses this reference frame in their ENC productions.

Description	Notation	Value	unit
Semimajor Axis	a	6378137	meter
Flattening	f	1/298.26	—
polar semi-minor axis	b	6356752.3142	meter

Table 2.2: The spatial parameters of WGS84 [7].

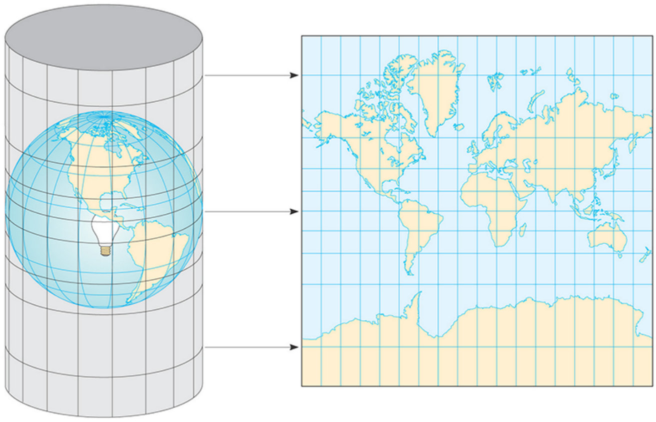

Figure 2.4: The Mercator projection is a cylindrical projection. Courtesy of `http://geospex.github.io/NSPE_Lectures_sp15/Class_4_Lecture/Class_4_index.html`

2.3.2 Projection problem

The geometry of the Earth is defined in three-dimensional space and so are the geodetic datums. A proper projection of the datums is necessary to represent the three dimensional reference frame of the datum on a two dimensional plane. This transformation will distort the relative positions in the map. Take a cylindrical projection for an example (see Figure 2.4). A canvas is wrapped around the world as a cylinder, and the surface of the world is projected onto this canvas with a straight line from the center of the Earth. The relative position will be distorted. This can be seen in Figure 2.4 where Antarctica is immensely large relative to the rest of the world.

A linear approximation is sufficiently accurate for small local map, where the curvature of the earth is negligibly small. The approximation is made about a chosen point where the gradient of the coordinates in the Cartesian space is used. The result is a two dimensional plane projection that tangents the geodetic reference system at the chosen point.

2.4 Path Planning and Collision Avoidance

Path planning is a term used in robotics, and it is a necessity to gain automatic motion control of a mobile robot. The subject is about generating collision free paths from starting positions to a final destination in a known or unknown environment. The environment might contain obstacles that the robot must consider and avoid.

Path generation can be differentiated between local and global path generation. The difference is the scope of the environment that is evaluated. Local and global path generation are suitable for different applications. A global path planner is often applicable for planning and generating a route for a mission, while a local path planner is often suitable for avoiding obstacles where the environment is unknown.

This section will present both global and local path planning methods that can be applied to navigation of a vessel.

2.4.1 Global Methods

A global path planning method considers the whole environment, and generates a collision free path from a starting point to a final destination. In case of moving obstacles in the environment, a global method requires information about the motion of obstacles in order to generate a collision free path. Global path planning is suitable for offline mission planning and route generation.

Generalized Voronoi Diagram

The Generalized Voronoi Diagram (GVD) can be used for path planning and is a road-map method. The advantage of using this method is the short computing time, which makes it suitable for real-time applications. The disadvantage is that the road-map generated might be non-optimal. Redundant waypoints may also be an issue [20].

Consider a workspace containing a starting point, an end point and one or more obstacles. The set defined by where the distance between the two closest obstacle is equal, constructs a network of lines. This is illustrated in Figure 2.5. These lines are the GVD and are a road-map that must be searched in order to find the optimized path from start to the end point.

A*

A* algorithm is used to find the shortest path in a graph and is a best-first-search. This means that the algorithm explores a graph by expanding the most promising node. The algorithm is similar to the well known Dijkstra's algorithm. The difference is that A* is guided by a heuristic function that is an approximation that represent the distance from a node to the destination. The method has the same worst case complexity as Dijkstra's algorithm, but is empirically faster [1].

A* will both find a path to the destination, if it exists, and an optimal solution if the heuristic function is admissible. A heuristic function is admissible if it never overestimates the cost. More specifically, the cost should be greater than, or equal to, the true cost from a node to the destination. A simple example of a heuristic function is the length of a straight line from the respective node to the destination. This heuristic function, however, will not be admissible since it might overestimate the cost as a straight line might pass through an obstruction.

The evaluation function of A* is as follows:

$$f(n) = g(n) + h(n) \tag{2.6}$$

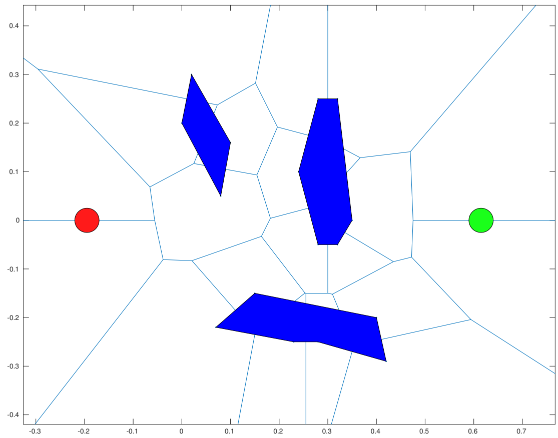

Figure 2.5: Simple Voronoi Diagram example. Red circle is the start, Green circle is end point, the blue boxes are obstacles and the blue lines are the Voronoi Diagram generated.

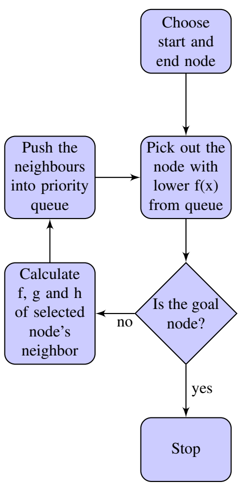

Figure 2.6: Flow chart of the A* algorithm.

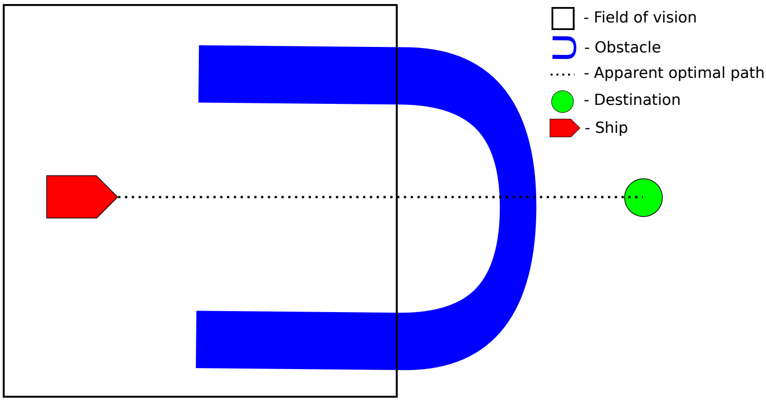

Figure 2.7: A situation where a vessel get stuck in a local minima.

where:

$$\begin{aligned} n \quad &= \text{Respective node} \\ f(n) &= \text{Evaluation function} \\ g(n) &= \text{Current cost} \\ h(n) &= \text{Cost to destination} \end{aligned}$$

With this formulation, the algorithm of A* is illustrated in Figure 2.6.

2.4.2 Local Methods

Local path generation tackles the area closest to the mobile robot. These methods are often purely reactive methods. This means that the mobile robot reacts on the changes through its sensor input.

A known issue of local methods is that the robot might end up stuck in a local minima. A U-shaped obstacle prone such situations as the robot might not sense the continuation of the U-shaped obstacle ahead (see Figure 2.7). The robot has to do sub optimal maneuvers over a period of time to avoid this situation. However, local action planning methods have been proposed to avoid this problem [3].

Local methods are typically used in real-time applications and require the processes to be time efficient. The methods are suitable for traversing in unknown, and dynamically, environments and the environment is constantly measured by sensors. The methods presented in this section have these properties.

Dynamic Window

The Dynamic Window method is presented by Fox et al. [9]. The method is derived directly by the dynamics of the robot, and take into account for the kinematic and dynamic constraints of the robot. It searches the velocity space for optimal trajectory, where the search space is a set of translational- and rotational velocities.

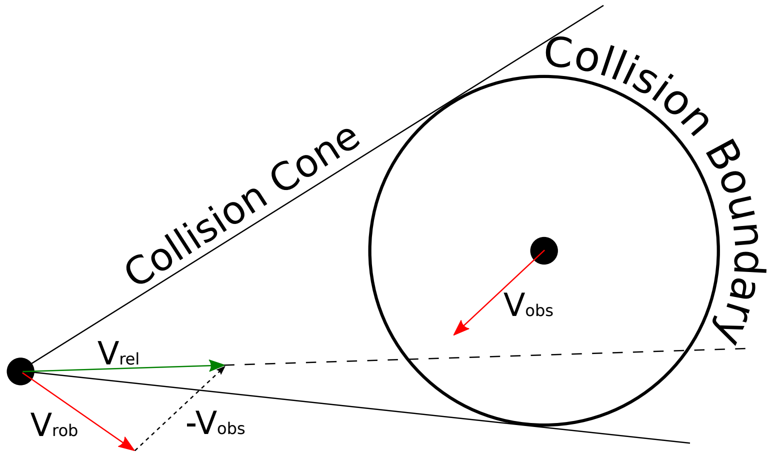

Figure 2.8: Illustration showing some of the principles of the Velocity Obstacle method.

The search space is reduced in two stages. The first stage reduces the set of velocities to what is called admissible velocities. An admissible velocity is a velocity that allows the robot to stop before a collision with an obstacle occurs. This stage considers the kinematic constraints of the robot. Second stage reduces the search space to the dynamic window. The dynamic window is the velocities the robot can achieve within a given time interval.

The resulting reduced search space is then representing control velocities that are both achievable and avoids collisions. This is then used to find an optimal control by applying a optimization scheme. The original suggested optimization scheme in [9] was to maximize the objective function in Equation 2.7, where $heading(\nu, \omega)$ is a measure of progress towards the goal location, $dist(\nu, \omega)$ is the distance to the closest obstacle on the trajectory and $vel(\nu, \omega)$ is the forward velocity of the robot. σ, α, β and γ is for smoothing and weighting.

$$G(\nu, \omega) = \sigma(\alpha \cdot heading(\nu, \omega) + \beta \cdot dist(\nu, \omega) + \gamma \cdot vel(\nu, \omega)) \tag{2.7}$$

Velocity Obstacles

Velocity Obstacle is a method that assumes that state information of an obstacle, position and velocity, is measurable. The method uses a collision cone defined by the relative velocities between robot and obstacle that lead to an intersection between the direction of the relative velocity and the collision boundary around the obstacle, (see Figure 2.8). This means that the relative velocities that lies within this cone will lead to a collision. The remaining relative velocities, lying outside of the collision cone, are collision free velocity controls, if the velocities are kept constant.

Like the Dynamic Window method, Velocity Obstacle method must search for velocity controls in a velocity search space to avoid obstacles. The collision free velocities must be reduced to reachable avoidance velocities. An optimization problem is then applied to this set for optimal control.

Artificial Potential Field

Artificial potential field is best explained through an analogy. Think of the potential field as a convex sloped terrain where the final destination is in the bottom of the terrain. A marble represents a robot and experiences gravitational forces. The marble starts moving, due to the slope, and will end up at the bottom of the terrain. In the sloped terrain there exists local elevations that represent obstacles. If the marble is moving towards a local elevation, it will be forced to go around this elevation.

The terrain with local elevations in the analogy is created by potential field functions. These are expressed in the Cartesian space of the robot. Obstacles are expressed as repulsive potential functions, equivalent with the elevation in the terrain, and the final destination point is expressed as an attractive potential function, equivalent with the slope of the terrain [22]. Figure 2.9 shows a simplified illustration of a robot guided by an artificial potential field.

An attractive potential function must be convex and is often chosen to be quadratic. Equation 2.8 is an example of a quadratic potential function, where x is the position of the robot, x_d is the position of final destination and k_a is a design parameter.

$$f_{atttractive} = \frac{1}{2}k_a|x - x_d|^2 \tag{2.8}$$

Repulsive potential functions are often described such that it affect the motion of the robot within a given range. Including this it is important to keep the derivative of the function smooth and continuous. Equation 2.9 is a repulsive function presented by Khablib [17], where ρ is the shortest distance to an obstacle, ρ_o is the limit distance and η is a design parameter.

$$f_{repulsive} = \begin{cases} \frac{1}{2}\eta(\frac{1}{\rho} - \frac{1}{\rho_o})^2 & \text{, if } \rho \leq \rho_o \\ 0 & \text{, if } \rho > \rho_o \end{cases} \tag{2.9}$$

2.5 Map Representations

A suitable map representation of the environment is an underlying problem to a path planning problem. Whether a representation is suitable or not, depends on properties of the environment and application. E.g. a map representation of a dynamic environment for a real-time application, must consider the real-time constraints and be able to update the changes in the environment.

This section presents two ways of representing an environment. Keep in mind that the scope of this thesis is to represent two dimensional, static environments.

2.5.1 Geometric Primitives

Geometric Primitives can be used to map an environment. Let a chosen set of different, or similar, geometries be distributed in space such that the obstacles in this space are completely covered. This distribution will be a representation of the environment by use of geometric primitive. Each geometry has a unique identity, and is related to a position

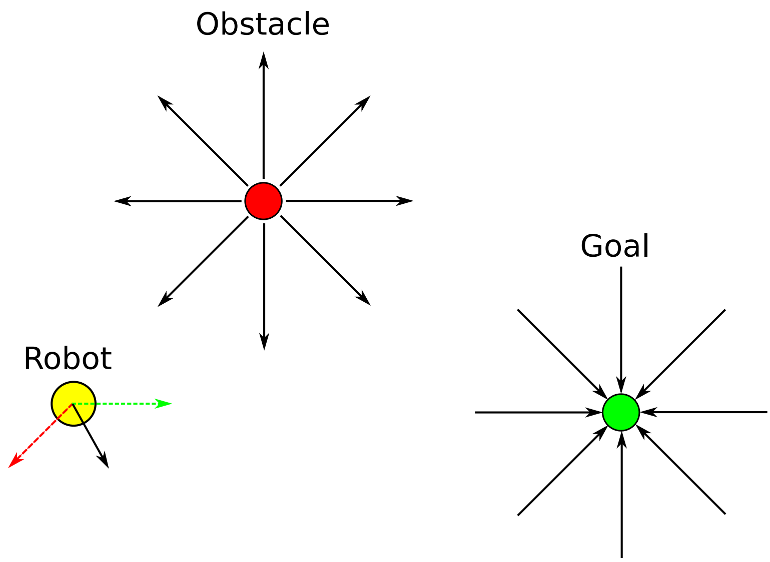

Figure 2.9: Artificial potential field illustration. The obstacle creates a repulsive potential field and the goal creates a attractive potential field. The illustration show three forces acting on the robot. The red one is the forces from the obstacle, the green is the forces from the goal and the black is the sum of forces.

in space. Different geometric primitives are defined differently. Eg. A circle can be defined by its radius, or a line segment can be defined by its length and direction. With this representation and computational geometry it is possible to localize obstructions.

There is a give-and-take compromise between computational time efficiency and accuracy of the representation. Less geometries are needed if they are large. This means that less computations are necessary. However, this will be a less accurate representation of the obstacles than a finer grained representation (see Figure 2.10).

The accuracy of the representation is dependent on the geometry to represent and the size of the primitives used. If large sized primitives are used, the representation will be coarse and the accuracy is more likely to be less accurate. The opposite would be usage of small sized primitives

Computational geometry for geometric primitives does not demand much processing power. However, to add and/or subtract new geometries to a space may be computational heavy. These properties indicates that the representation is suitable for describing static environments, and can be used in real-time applications.

2.5.2 Occupancy Grid

In its most basic form, an occupancy grid is a binary grid where each cell indicates if the area is occupied or not. Figure 2.11 illustrates an occupancy grid of a coastal map. A cell is indicated obstructed if it contains an obstacle. The representation is conceptually simple and useful for path planning and obstacle avoidance.

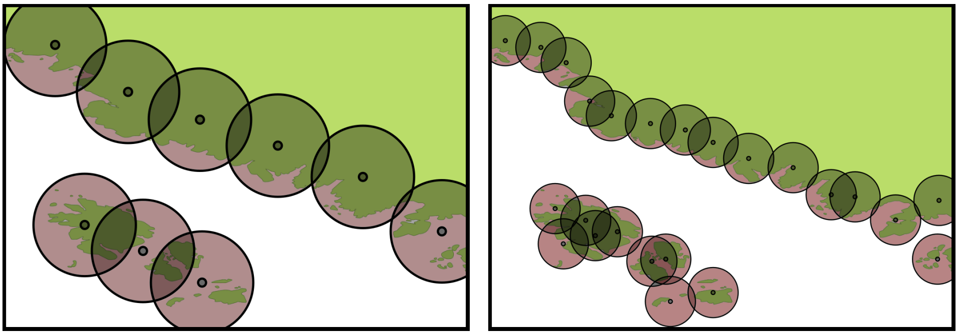

Figure 2.10: Illustration of geometry representations. Circles are used to indicate obstructions and the red areas are the inaccuracy of the representation.; *left*) Coarse representation with less obstacle needed to represent the coast, *right*) Less coarse representation with higher accurancy.

It is common to use an occupancy grid for real-time map building and is often used as a probabilistic map. Each cell contains a confidence value telling the likeliness of occupation. The strength of this is that the representation is able to handle noisy sensor data, which makes it a popular choice for map building applications.

Similar to a geometric representation, there is a trade off between computing time and accuracy of the occupancy grid representation. The resolution of the occupancy grid is determined by the size of the cell. Large cells are more time efficient and less accurate than small cells.

An occupancy grid is able to represent square objects quite well, like walls in a room. This makes the representation suitable for indoor applications. However, a representation of objects with arbitrary shape, like the coast line in Figure 2.11, will result in an inaccurate representation.

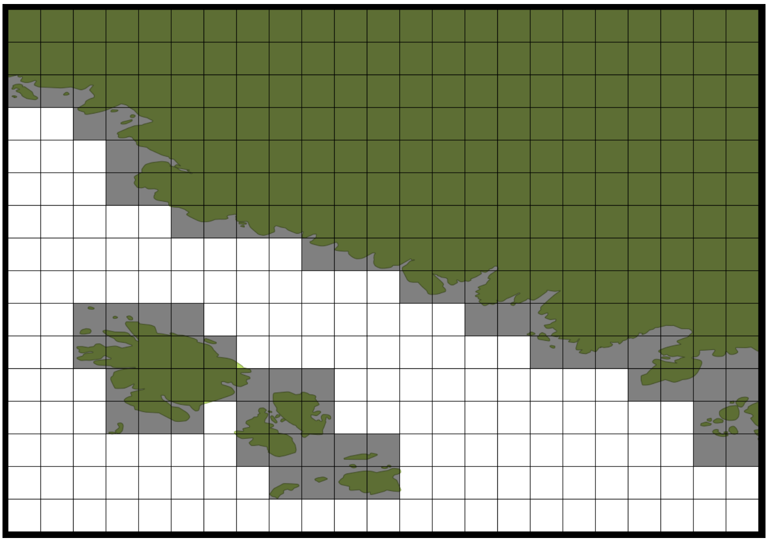

Figure 2.11: Occupancy grid representation of a coastal environment. Cells intersecting with land are indicated occupied (shaded). The remaining empty cells are the space to move freely in.

Chapter 3

Software and Implementation

This chapter starts with presenting third party softwares that are used in the implementations of this thesis. Further, it will take the reader through thought process and main ideas that are used during the implementation. The final implementations of this thesis will be presented as well in this chapter.

The computer used for development and simulations is a Dell OptiPlex 7040 with an Intel i7 core. When iteration time of the CAS is mentioned in the text, it is the iteration time of simulations done on this computer. Have this in mind as the iteration time is dependent of, and vary with, the processing capabilities of the computer.

3.1 Software Requisites

The implementation work done in this thesis uses two third party software libraries; Robot Operating System (ROS) and GDAL/OGR. A brief introduction of the software, why they are chosen and what they are providing is presented in the following sections. Methods that are used and considered important to understanding are also mentioned.

ROS

ROS is an open source framework for development of robot software. It provides drivers, developer tools and libraries. The framework is based on a node structure communicating through a *roscore*. *Roscore* is the master of the node structure and handles the communication between the nodes. This makes it a modular system and makes it easy to add new nodes to a system.

The previous work done and the ASV simulator were implemented with ROS as framework. Since the simulator is implemented in ROS, the work of this thesis will also be implemented with ROS. However, ROS is a qualified choice with respect to functionality and ease of further development.

ROS has a utility called *rosservice*. This is a request-and-reply procedure. The caller of an *rosservice* requests an answer from the service server. The request is processed and replied by the server, and giving the rosservice caller the requested information.

GDAL/OGR

GDAL/OGR [10] is an open source translating library for vector and raster geospatial data formats. It provides efficient vector IO, and allows the user to access geospatial vector data. The software supports many different vector data formats, and presents a single abstract data model. The library provides a C++ API, python API and command line utilities.

This software was chosen since it seems promising with respect to efficiency. It also had drivers supporting S-57 data. Additionally, many powerful, well known programs, like QGIS, MapServer and other spatial data applications, use the functionalities of this library. This indicates a unified quality approval of the library in the GIS community.

Other vector IO software that had drivers supporting S-57 access are ArcPy and Fiona (not by default). Both of these software libraries uses the GDAL/OGR library. To avoid intermediary software, GDAL/OGR was favored over other vector IO libraries.

GDAL/OGR has a class called *OGRGeometry*. This class owns a method called *Intersects*. The method determines whether two geometries intersect by utilizing functionality of GEOS (an opensource geometry engine). The intersection check is based on the Dimensionally Extended nine-Intersection Model (DE-9IM). The model was developed by Clementini et al. [5][6] and is a efficient model for determine spatial relations between geometries, e.g. intersection, disjoint, contain, etc.

3.2 Implementaion

Different ideas have been implemented and tested throughout this project period. Initially, the idea was to implement an anti-grounding cost weighting that worked directly on the ENC data set of S-57 format. This quickly led to problems since the S-57 drivers of GDAL/OGR only support read-access to the data set. Since there is a need for modifying the content of the ENC to generate a suitable representation for the CAS, the access restriction was a limiting factor in the implementation. This led to a decision to split the code base into two parts: one offline-code for preparation of the map, and one real-time code for implementation of an anti-grounding system.

For simplicity and to keep the preprocessing of data to a minimum, it is chosen to experiment with representing hazards as polygons of vector format, which are similar to the original data. This is an unconventional and unusual way to represent obstructions in a real-time application. Methods that are more conventional choice of representing obstruction maps are for example occupancy grid (see Section 2.5.2).

3.3 The Offline-Code

The offline-code prepares information about the ENC for use in CAS. The purpose of this code is to simplify the original data set, and extract the areas that are hazardous for a vessel

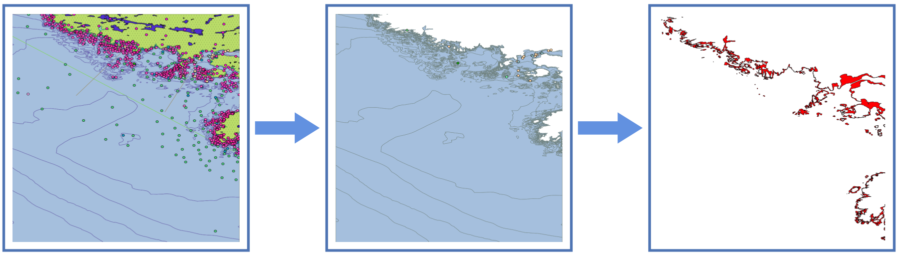

Figure 3.1: The sequence of hazard extraction from an ENC

to traverse. The product of this code is a suitable representation of mapped hazards and is stored in a new file.

The offline-code base in published, and can be found on GitHub (`https://github.com/olesot/map-extraction`).

3.3.1 File Format

The resulting file is in the *ESRI Shapefile* format. The ESRI Shapefile driver of GDAL/OGR support both read- and write-access. Translating S-57 data to Shapefile is a solution to the access problem of the S-57 driver. Just like S-57, Shapefile is a vector file format for geospatial information. Unlike S-57, Shapefile requires one file per layer and it is not possible to mix different geometric shapes (point, line and polygons) in one file. But from the perspective of GDAL/OGR software abstraction, the different format will not behave differently. The differences are handled by the drivers of GDAL/OGR.

3.3.2 Stages of Extraction

The stages of the configuration code are illustrated in Figure 3.1. The figure illustrates two steps. The first step is to extract the necessary information from the original data source. The decision of what to extract is decided by an assessment as described in Section 2.1.4, and this assessment will yield a reduced version of the original data set. The reduced version contains significantly less information, but still is too detailed for practical use, having multiple layers and different geometric shapes. Thus, it is not yet a suitable representation of the hazardous areas for the CAS to interpret. The next step therefore simplifies the reduced version to a single layered data set with one or more polygons. At this stage, interesting entities, previously described as points or lines, are translated to polygons. The translation depends on what the points/lines are describing. The polygons represent the areas the vessel should avoid.

3.3.3 Depth Data

The configuration code is giving special attention to the depth area layer, called *DEPARE*. This layer provides elevation information about the seabed. The layer contains several features where each represents one level of elevation. Seabed with depth less than the beam depth of a ship is considered as hazardous area to traverse. The *OGRLayer* class of GDAL/OGR provides a method call *SetAttributeFilter*. This filter out attributes with a query expressing a condition. With this functionality, it is possible to extract the features from the *DEPARE* layer where the depth is less than a given value. This value should be chosen to be larger than the beam of the respective vessel. This query should be done on the attribute called *DRVAL1*, which is the shoalest value of a depth range, according to S-57 object catalogue.

3.3.4 Geospatial Reference System

Geospatial data is measured by coordinates and is referenced in a geodetic reference system. The measured coordinates are only valid in this reference system. This is an important aspect of geospatial data and an implementation should be carefully aware of these issues. The hazard extraction previously described uses the same projection as the original data set and pass the reference system throughout the whole extraction procedure. The ENC used in this thesis is produced by Kartverket and they use the WGS84 projection. This means that the resulting hazard map is also represented in this geodetic reference system.

3.4 Anti-Grounding System

The realization of an anti-grounding cost weighting is based on the idea of doing a query and retrieve if a point is within a hazardous area or not. This could be used on the trajectory predictions done by the CAS. The predictions generate a list of points that makes up the path to be inspected. A cost scheme would be implemented and applied to decide the cost if the trajectory enters hazardous area. The query would be done up against the hazard map generated by the configuration code.

Hagen's implementation [11] is forked at github and modified by including the real-time code. The forked repo can be found at GitHub (`https://github.com/olesot/ros_asv_system`).

3.4.1 Real-time Constraint of the CAS

An important factor for this part of the code is the limitation of time. The CAS is a real-time system and is limited by time. Heavy offline processes are useless for this purpose. The outer loop iteration of the CAS is set to be five seconds. Any code that trespass this time limitation is considered useless and will cause ultimate failure of the system. Code that uses time close to five seconds but not above, is considered to have insufficient performance. The calculated optimal control is optimal for the state of the ASV at sampling time. During the time of calculating the optimal control, the state of the ship may change. If the time spent is large, the deviation of the state at sampling time, and the state where optimal control is applied, may also be large. This address the motivation of reducing the computation time of the optimization problem. It is a necessity to keep this sufficiently low to gain well performance of the CAS.

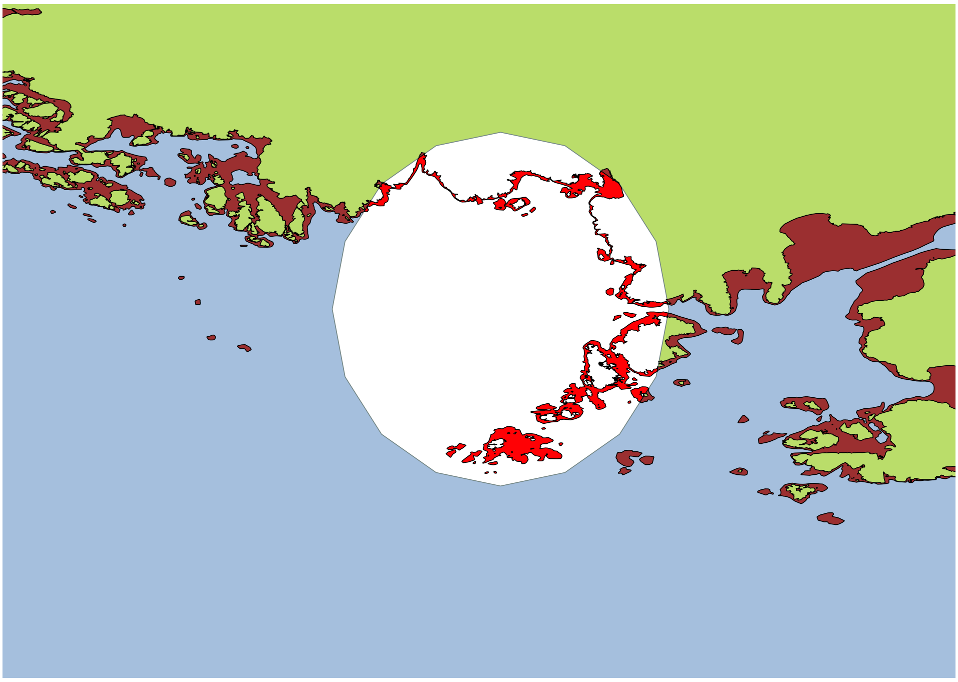

Figure 3.2: Illustration of the anti-grounding system's field of vision. The white circle is the field of vision. All static obstructions within this circle are evaluated by the anti-grounding system.
Hazards in sight of ASV Other hazards Sea Land

As mentioned in Section 2.2, the CAS utilizes MPC, which is an iterative method. The system has a total of 52 combination of the control variables χ_{ca} and P_{ca} to iterate through. The time it takes to solve the optimization problem, is the time used to calculate cost for one combination of controls, multiplied with the number of combinations there are to iterate trough. This address the issue of adding additional computations to an iterative method and motivates to keep the additional computation time low. The methods used for avoiding groundings must therefore be time-efficient.

The time used by the intersection check is directly dependent of the number of vertices, that defines the polygons in the hazard map, to evaluate. A reduction of vertices will lead to a reduction of time to do an intersection check. Therefore, to further improve the time-efficiency of the method, the real-time application is only evaluating parts of the hazard map that are closest to the ASV. At the start of each optimization problem of the MPC, the map to be evaluated is updated to be within a given distance from the ASV. The result of this is illustrated in Figure 3.2. The white circle is the area that is the coordinate frame related to the ASV used for evaluating static obstructions. This reduce the problem space and i.e. reduce the number of vertices to evaluate.

3.4.2 Conversion of Space

The reference frame of the CAS is planar, i.e. the position of the ship is expressed in the Cartesian space with units in meters. This issues the fact that the map is represented in three dimensions by coordinates, and needs to be converted to the Cartesian space. This is the same problem as with the projection problem addressed in Section 2.3.2. The environment considered in this thesis is small enough to be linearly approximated in the same way as described in the projection problem section. The approximation is done about a chosen coordinate of the map, and the gradient is expressing the change of latitude and longitude in Cartesian space at chosen coordinate.

3.4.3 First Version Idea

The first version of the real-time code that was implemented, was based on the idea of making an independent ROS package that provided the cost penalty as a *rosservice* (see Section 3.1 for information about *rosservice*). This abstraction would be beneficial regarding modular code structure and to create a clean overview of the structure. The *rosservice* implemented requested a position and responded with a cost value to penalize paths entering hazardous areas. The *rosservice* call was done in the MPC code of Berges implementation [11] in the *rospackage asv_ctrl_sb_mpc* as it iterated through the position points of the different predicted trajectories, i.e. the innermost iteration loop, shown as sample loop in Figure 3.3. The cost value was included into the cost function of the MPC.

This method showed to be time inefficient and trespassed the time limitation of the CAS. The method used approximately 30 seconds for each iteration. The time inefficiency was located to be the *rosservice* call and how it was called. Several forums address time issues with the *rosservice* method, e.g. inefficient recreation of thread state instance, and suggestion of keeping simple procedures for quick termination. This led to the decision of moving away from a solution of using *rosservice*.

3.4.4 Final Version

It was decided to move all code and integrate it to the *asv_ctrl_sb_mpc* node. It was also decided to alter the method of checking intersection along the predicted trajectories. This was done to prevent executing a method call in the innermost loop of the system by moving it out one level (from sample loop to P_ca loop, ref. Figure 3.3). Additionally, instead of checking each point along a trajectory for intersection, the trajectories were partially checked. A trajectory was segmented, and each segment was checked for intersection with hazardous areas of the hazard map.

This implementation shown to be sufficient time efficient, where the average time for one iteration was approximately 0.5 seconds, and is used in the simulation study in Chapter 4. The method leaves the user to specify the segmentation of the trajectory, and the corresponding penalty for intersecting the different segments. This can be considered as a tuning of the system. Natural choices for which part of the trajectories that should be considered can be based on the look ahead distance of the CAS, i.e. the parameters $d_{safe} = 40[meter]$ $d_{close} = 400[meter]$. The choice of cost weighting values is not trivial. This is dependent of the weighting of the CAS and should be balanced with the system.

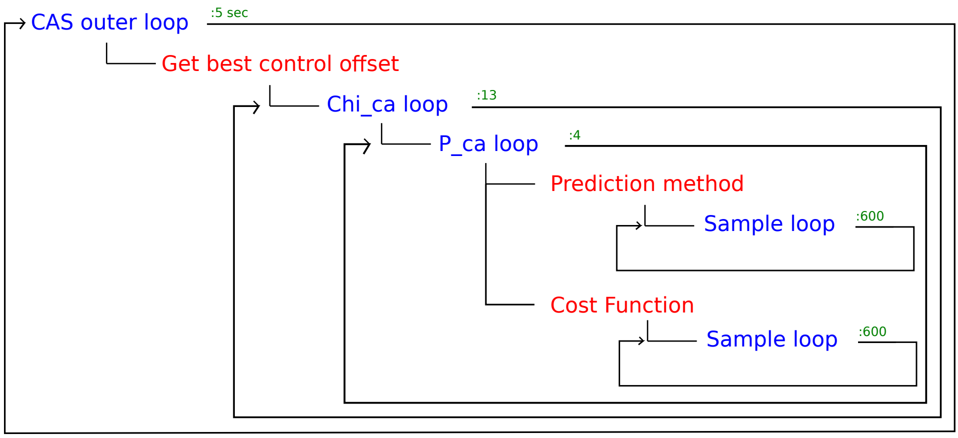

Figure 3.3: Illustration of the nested loop structure of the Collision Avoidance System implementation. *Get best control offset* returns the control offset minimizing the cost function. *Chi_ca loop* and *P_ca loop* iterates through the chosen possible control set. *Prediction method* is generating the trajectory of the vessel with given offsets. *Cost function* calculates the cost of the trajectory generated. The respectively loops are notated with the number of iteration.

Choosing number of segments is just a matter of time efficiency. The more segments there are, the more intersection checks are executed.

3.4.5 Method Weakneses

The current implementation of the anti-grounding system has its weaknesses. The most obvious issue is the segmentation of the predicted trajectories and the nature of the GDAL/OGR *intersects* call. The issue will show when the segmentation is chosen to be coarse. An intersection along a single segmentation will be penalized equivalently, i.e. an intersection with mapped hazards that is close to ASV is penalized the same as an intersection further away, but still within same trajectory segment. The issue is illustrated in Figure 3.4.

The method is greedy. This means that the method is only favoring the cheapest trajectory. A method that is greedy is prone to get stuck in a local minima, as addressed in Section 2.4.2. However, the method is integrated to the more sophisticated CAS that has additional decision-making schemes. The CAS is, among other things, evaluating retarded motions, which may prevent this issue.

3.5 Visualization

Empirical observations are used to analyze the effect of the anti-grounding cost weighting has on the ASV's motion. The observation is done by using the simulator implemented by Stenersen [25]. Extracted static hazards need to be visualized in the simulator for this purpose. This was done by creating a mesh file of the map data. Blender with BlenderGIS (`https://github.com/domlysz/BlenderGIS`) addon is used to create the mesh

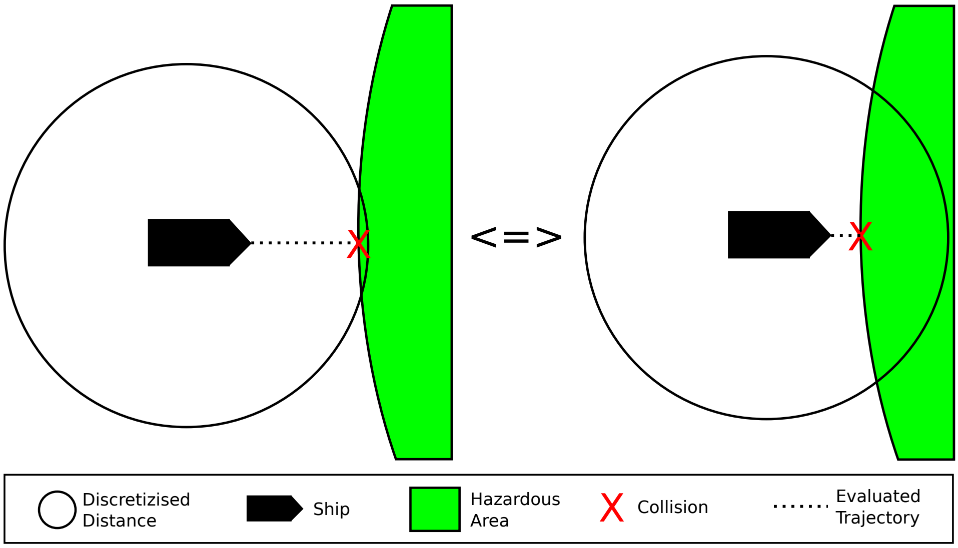

Figure 3.4: Two scenarios that will be penalized similarly by the anti-grounding system.

file. The creation of the mesh file was done manually.

To get a correct visualization of the mapped hazards, it is important to align the mesh file with the inertial frame of the simulator correctly. The origin of the inertial frame of the simulation environment is assigned a chosen coordinate. This can be advantageous chosen to be the same coordinate as the linear approximation is done about (ref. conversion from three dimensions to Cartesian space in Section 3.4). It is then important to make sure that the origin of the mesh file is corresponds to the chosen coordinate. BlenderGIS allows the user to choose the origin of a mesh file by a coordinate. Including to this, it is also necessary to scale the mesh correctly in the simulator environment. If these two procedures are done correctly, the map will appear similarly to a projection from a geodetic datum to a plane (see Section 2.3.2), and will correspond to the conversion from three to two dimensional space as described in Section 3.4.

Chapter 4

Simulation Study

This chapter presents the experiments done in this thesis through a simulation study. A comparison of the CAS with and without anti-grounding cost is done. The CAS implemented in [11] shows well performance and adheres COLREGS regulations. This behaviour is the basis for the comparison, and is equivalent with maneuvering at open sea, where there is sufficiently deep water and no hazardous area to traverse. To achieve the same behaviour of the simulations without the anti-grounding **cost/system** included, similar situations are set up for the simulations and the parameters used in Hagen's thesis are used (see Table 4.1). Any deviation from this will be pointed out and discussed in Chapter 5.

As mentioned in Section 3.4.4, the user has to decide segmentation of the predicted trajectory, and the penalty for intersection with the segments. Table 4.2 is the parameters used for the simulation study, and defines two segments and their penalty. *Safe* is the segment that is closest to the ASV and *close* is a segment that is adjacent to *safe*. An extensive parameter tuning of the parameters is not been done in this thesis, but rather chosen to show proof of concept of the anti-grounding system. An predicted trajectory that intersects in the *safe* segment must be eliminated in the optimization problem. Therefore the cost is chosen high. A predicted trajectory that intersects in the *close* segment is chosen to be considered in general less favorable, but still COLREGS compliance is more favorable than intersection in this region. Therefore, $cost_{close}$ is chosen to be less than κ in Table 4.1. κ is the cost of violating COLREGS regulations.

The simulations are observed empirically and are illustrated by the simulation environment as a time sequence of the simulation. The different positions of the ASV and the obstacle ship are marked with a number that correspond to a certain time. To get a perspective of distances, a grid is placed in the environment. The length of the grid is 100 meters. Corresponding offset controls, calculated by the CAS, are also presented in the same figure.

The obstacle ship is a "dumb" component of the simulations. It is programmed to keep its speed and course, and do not consider the COLREGS regulations. In some of the simulation scenarios, this will be prominent and affect the results of the respective

Parameter	Value	unit
T	300.0	
dt	0.5	
p	1.0	
q	4.0	
d_{close}	200	
d_{safe}	40	
K_{coll}	0.5	
Φ_{AH}	15.0	
Φ_{OT}	68.5	
Φ_{HO}	22.5	
Φ_{CR}	68.5	
κ	3.0	
K_P	4.0	
K_χ	1.3	
$K_{\Delta P}$	3.5	
$K_{\Delta,starboard}$	0.9	
$K_{\Delta\chi,port}$	1.2	

Table 4.1: Parameter values of CAS used in the simulation study. The values are the same as used in [11]

Parameter	Value
$cost_{safe}$	1000
$cost_{close}$	2.9
d_{safe}	20
d_{close}	40

Table 4.2: Parameters related to anti-grounded that are used in the simulation study. d specifies the numbers of samples from t_0 that defines the corresponding segment. *cost* specifies penalty for intersection with corresponding segment. *Subscript* indicates corresponding segment.

simulation.

4.1 Grounding Avoidance

The first simulation shows an unrealistic scenario. The planned path for the ASV goes through a hazardous area. A proper mission plan considers the maritime environment, and choose waypoints such that the path to follow is not intersecting with hazardous area. Nevertheless, it is still an interesting scenario with respect to the performance of the anti-grounding system. Figure 4.1 is showing the result of the simulation. The ASV is forced alter its course to port side by the hazardous area on starboard side.

sta

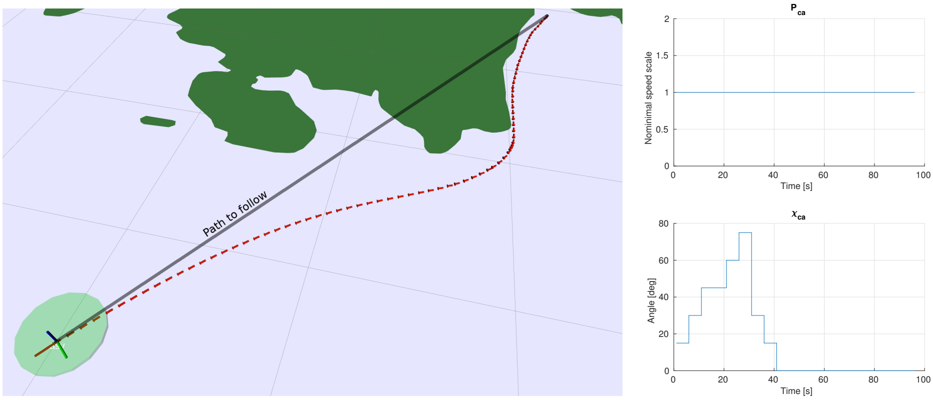

Figure 4.1: The path to follow is crossing hazardous area. However, the ASV is guided by CAS with anti-grounding system and is able to avoid the hazardous area.

4.2 Head-on Situation and Hazards on Starboard Side

The set up of this simulation is a head-on situation between the ASV and obstacle ship. The vessels are heading straight towards each other and according to COLREGS rule 14 (appedix A.6) the vessels are supposed to alter the course to starboard to pass. On the starboard of the ASV there is a hazardous area that the ASV must also consider and avoid. From the CAS point of view this will be an opposed conflict where the ASV registers hazardous situations from both sides. The expected behaviour is to find the mean and keep appropriate distance from both the obstacle ship and the mapped hazard.

Figure 4.2 shows a simulation of the head-on situation where the anti-grounding system is disabled. The simulation shows that the ASV keeps its speed and alters its course to starboard. This makes the ASV enter the hazardous area on starboard side.

Figure 4.3 shows the result of the simulation with anti-grounding system enabled. The figure shows that the ASV alters its course to starboard, which in compliance with COLREGS. When the ASV approaches the mapped hazard, it reduces its nominal speed to avoid grounding. As the obstacle passes the ASV, the ASV is gradually returning back to nominal speed.

4.3 Passing Through a Strait with Crossing Obstacle

Two possible situations can occur that are considered as a crossing situation. An obstacle ship can cross the ASV's path from starboard, and from port side. Both situations are presented in this section. According to COLREGS rule 15 (appendix A.7): *"the vessel which has the other on her own starboard side shall keep out of the way"*. This means, in a crossing situation, that the passive vessel is obliged to alter its course and/or speed to pass behind the other vessel. Common for both simulations are that the ASV is passing through a strait. This limits the space of the ASV to freely move in.

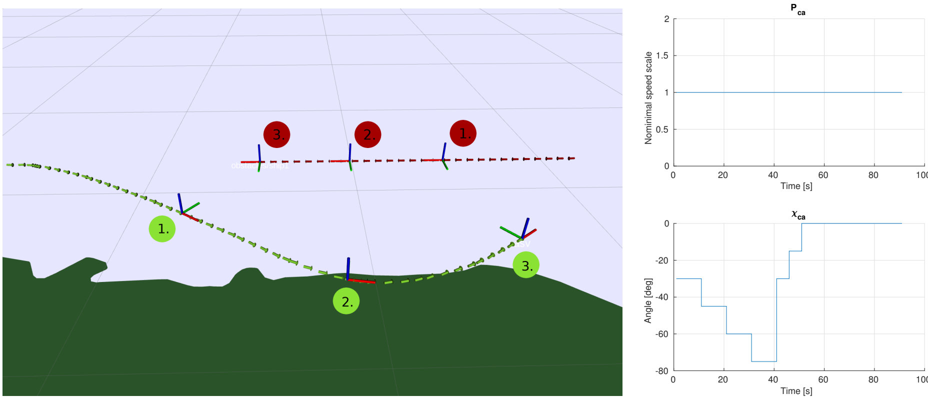

Figure 4.2: Head-on situation without anti-grounding cost. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

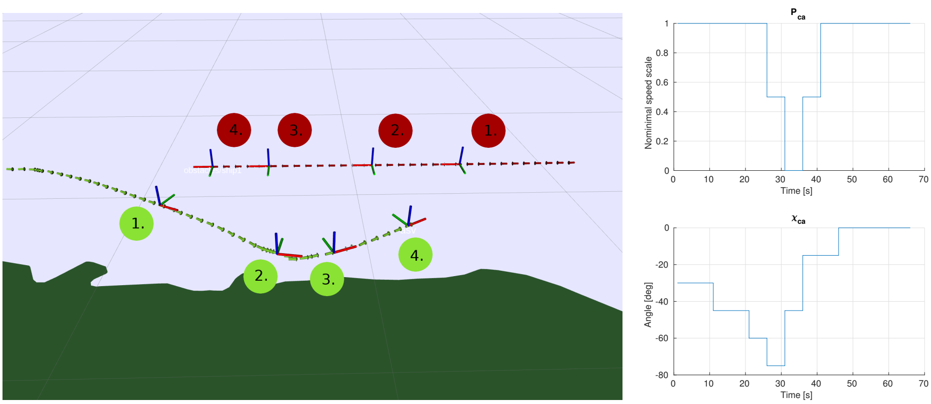

Figure 4.3: Head-On situation with anti-grounding cost. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

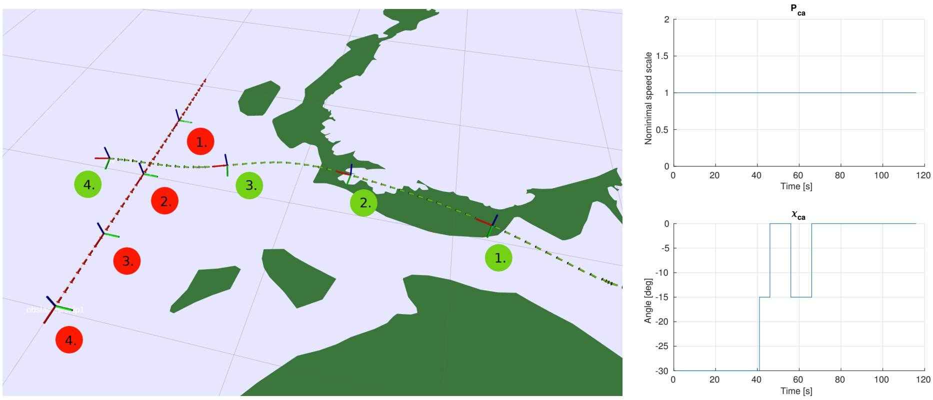

Figure 4.4: Crossing situation without anti-grounding cost and obstacle ship comes from starboard of ASV. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

4.3.1 ... from Starboard

The obstacle ship comes from starboard. This makes the ASV the passive vessel, and is suppose to give way for the obstacle ship, and pass behind. Figure 4.4 shows a simulation of CAS where the anti-grounding system is disabled. This simulation shows that the ASV alters its course to starboard, and keeps its nominal speed. It also clearly shows that the ASV traverses through hazardous area.

Figure 4.5 shows the simulations where the anti-grounding system is enabled. Here we can see that the ASV do not traverse the hazardous area. The anti-grounding system prevents the ASV to do large course alterations, since this would result in entering hazardous area. The ASV reduces the nominal speed instead to prevent collision with the obstacle ship.

4.3.2 ... from Port Side

This time the obstacle ship comes from port side of the ASV. The obstacle ship is then the passive vessel and is expected to give way to the ASV. However, the issue about "dumb" obstacle ship, addressed initially in this chapter, shows in this simulation. Figure 4.6 shows the result of a simulation of this situation with the anti-grounding system disabled. In order to maintain the right to pass in front of obstacle ship, the ASV alters its course more and more to starboard. The ASV is forced to do this since the obstacle ship do not follow COLREGS regulations. It is also observable that the ASV traverses through hazardous area with this collision avoiding maneuver.

Figure 4.7 shows simulation of the ASV with anti-grounding system enabled. It can be seen that the ASV avoids hazardous areas. Despite the right of passing in front of the obstacle ship, ASV reduces its speed to prevent a collision. The ASV shows similar behaviour as in previous simulation, i.e. Section 4.3.1.

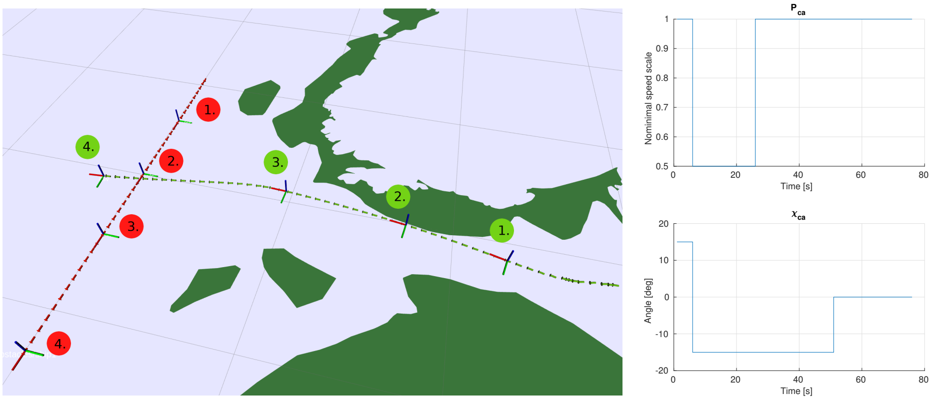

Figure 4.5: Crossing situation with anti-grounding cost and obstacle ship comes from starboard of ASV. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

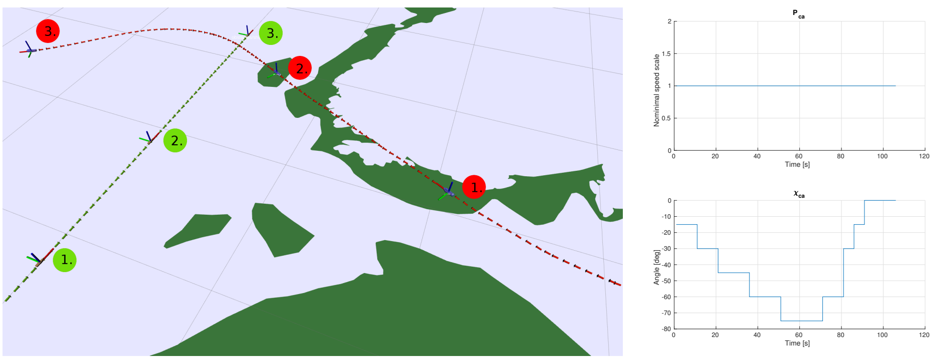

Figure 4.6: Crossing situation without anti-grounding cost and obstacle ship comes from port side of ASV. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

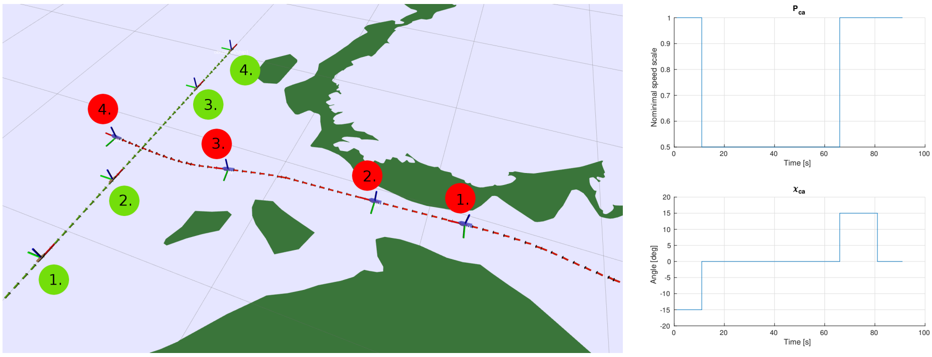

Figure 4.7: Crossing situation with anti-grounding cost and obstacle ship comes from port side of ASV. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

4.4 Overtaking Situation with Hazards on Starboard Side

The ASV is overtaking an obstacle ship in this simulation. An overtaking situation is where a vessel approaches an another vessel, and passes it, from behind. According to COLREGS rule 13 (appendix A.5): "...any vessel overtaking any other shall keep out of the way of the vessel being overtaken". This means that the ASV need to do maneuvers that do not interfere with the convoy of the obstacle ship.

The simulations are interrupted before the ASV passed the obstacle ship completely on purpose. The interesting part of the simulation is the initial phase of the overtaking maneuver, where the vessels have hazardous areas on their starboard.

Figure 4.8 shows simulation without anti-grounding system. The ASV simply changes its course to starboard and pass the obstacle ship at a safe distance. The change of course is applied already at first iteration of finding optimal control offset. This results in the ASV entering hazardous area.

The simulation with the anti-grounding system enabled is shown in Figure 4.9. Here we can see that the ASV avoids the hazardous area on the starboard side. The ASV changes its course to port side initially and at a time turns over to starboard again. When the ASV approaches hazardous area, it chooses to turn back to port side course offset again. It keeps its port side offset controls until it passes the obstacle ship.

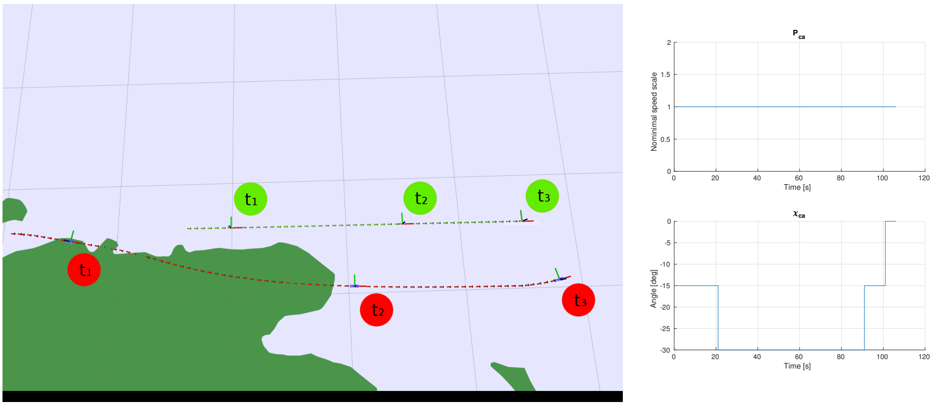

Figure 4.8: Overtaking situation without anti-grounding cost. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

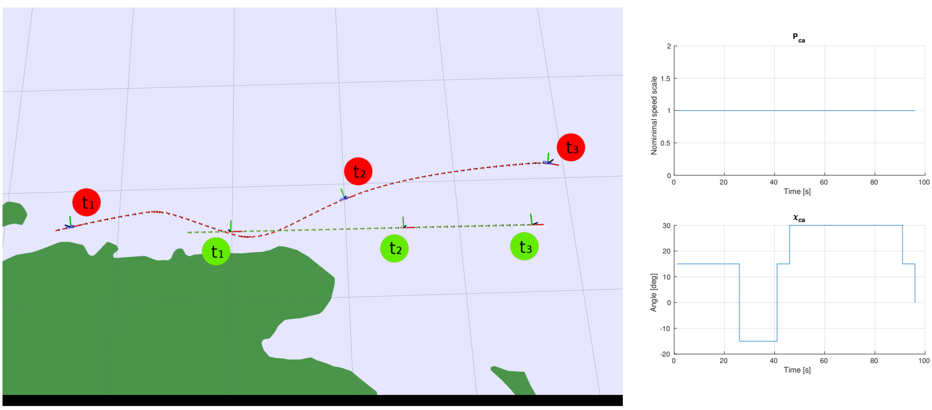

Figure 4.9: Overtaking situation with anti-grounding cost. A time sequence of the simulation is shown to the left, and the corresponding control offset input to the right.

Chapter 5

Discussion

The simulation study shows that the ASV, with the anti-grounding system enabled, is able to avoid the mapped hazards in different scenarios. Simple scenarios are analyzed, with only one moving obstacle included. It is intuitively easy to say what the best maneuver is, and that still comply with the COLREGS regulations, for most of the situations simulated. The most challenging scenario, with respect to COLREGS compliance, is shown to be where the obstacle ship crosses from port side and do not give way to the ASV. COLREGS rule 11 (appendix A.4) says: "*Rules in this section apply to vessels in sight of one another.*", where *section* is a reference to the section of COLREGS rules. Further COLREGS rule 3 (appendix A.1) states: " *Vessels shall be deemed to be in sight of one another only when one can be observed visually from the other*". This makes it hard to tell whether the ASV has the right to pass in front or not. The two respective vessels are separated by hazardous area. This area may be elevated landscape, preventing line of sight. I.e. COLREGS compliant behaviour is conditional and dependent by the environment. However, the ASV shows precautionary behaviour by reducing its speed.

The anti-grounding system is only considering the predicted trajectories of the ASV. This will result in accepting trajectories that are almost tangent to the boarder of the hazardous areas. This can be seen in the first simulation, where path of the ASV is set to go through a hazardous area (see Figure 4.1), and crossing situation from port side (see Figure 4.5). Lets say that the mapped hazards are perfectly accurate with the real world. The convoy close to the mapped hazards issues several different risks of grounding by choosing such trajectory. Environmental forces may force the vessel into the hazardous area issues one risk. The width of the vessel is expanded orthogonal on the trajectory, and even though the trajectory do not enter hazardous area, the spatial geometry of the ASV might do issues a second risk. Additionally, uncertainties of the hazard map makes it also hazardous to convoy close to the map. These issues must be addressed and taken care of. These issues indicate that the anti-grounding system may consider the distance to hazardous area as well as the predicted motions of the ASV. A safety margin added to the mapped hazards can also be considered as a necessity.

There are several factors that affects the accuracy of the anti-grounding system. The method used can not be more accurate than the representation of hazards it uses. On the other hand, a perfectly accurate hazard representation might be redundant, since the representation should be bounded by a safety margin. More important is the accuracy of the method itself. The method uses trajectory prediction of the MPC. This means that the accuracy of grounding predictions is dependent of accurate predictions. Furthermore, the linear approximation for translating coordinates to the Cartesian space is also affecting the accuracy of the method.

The method used is intuitively simple, but complex to optimize. The anti-grounding system introduce new parameters to be tuned, which are dependent of the tuning of the CAS. The tuning of the CAS is not straight forward, as pointed out by Hagen [11]. The new tuning parameters will further add complexity of the tuning. However, the simple tuning scheme, specified initially in this section, is shown to be sufficient for these simple scenarios. On the contrary, this might not be the case for more complex situations. The need of study the choices of parameters is necessary.

5.1 Future Work

This project has been experimenting with information extraction from ENC, and has been a contribution to the research of hazard representations and how this can be used. However, the work of this thesis alone is not enough to say a best suited method is found.

For further development and improvement of the implemented method, following points must be considered:

- Retrieve better knowledge of the objects described in the object catalogue of the S-57 publication, and understand the meaning of it with respect to maneuvering choices.
- Find a more suitable tuning scheme of the parameters.
- Consider to achieve a more seamless segmentation of the grounding check.
- Consider the distance from ASV to hazards to avoid motions close to hazardous areas.

With respect to the research of the possible usage of an ENC for autonomous navigation, following points must be considered:

- Further research of possible methods to use for both global and local usage.
- Address the possibility of automatic generation of routes based on the information of an ENC.

Chapter 6

Conclusion

This thesis started out with a set of questions about different ways that mapped hazards can be used in a CAS. Initially, the goal was to develop an application that extracts information about mapped hazards from an ENC and represents it as a suitable input for the CAS. My initial problem statement also specified that this was to be used in an absolute coordinate framework for automatic route planning, and a local coordinate framework relative to the position of the ASV for real-time decision-making. Additionally, the ambition was to carry out a full-scale experiment to verify the software module.

However, as the project progressed, the focus changed due to various constraints on interactions with ENC data. The extraction of hazards from ENC and experimentation with a suitable representation of hazards was therefore prioritized. This included the creation of a real-time decision-making application. Due to these unexpected challenges with accessing and handling ENC data formats, the assessment of automatic route planning and full-scale experiments fell beyond the scope of this study. Automatic route planning and full-scale experimentation should be developed in the future.

An in-depth assessment of the S-57 format is presented in Chapter 2. This information is used to parse and extract information from the ENC correctly. An application is created and implemented for this purpose, along with an assessment of deciding what to be considered as hazards. The output of this application is a representation of hazardous areas. This representation is used by a real-time decision-making application that has been implemented and described in Chapter 3. The real-time application provides anti-grounding cost weighting to the MPC-based CAS, and a real-time application was tested through several simulations of different scenarios, and described as a simulation study (Chapter 4).

To deal with ENC and the information these contain adequately, it was necessary to know its data structure. In the landscape of terms like feature objects and transfer layers, I found it necessary to do an assessment of the format. The assessment provided in this thesis provides the necessary knowledge of the S-57 format with respect to hazard extraction. The choice of software for accessing the data landed on GDAL/OGR. The chosen third party software shows qualities by providing both time efficient method calls and a wide range of format drivers. This software is a suitable choice regarding real-time programming. However, future users should be aware that the API requires effort to fully master. Still, the software demonstrates both flexibility and functionality in the hazard extraction application.

The representation of hazard map chosen here is experimental. Here, I have represented the hazard map as polygons in vector data format. I have also shown how real-time search for intersections in the representation is possible without exceeding the time limitations of the CAS. The chosen software and method are shown to be sufficiently efficient for this purpose.

I have presented an anti-grounding system that is able to detect and avoid the hazardous areas described in the hazard map. The simulation study shows that the ASV is able to avoid hazardous areas without violating the COLREGS regulations. This was achieved by a simple and time-efficient method that make use of the predicted trajectories. However, the method is not been optimized, and there are reasons to believe the method can be improved.

Bibliography

[1] Ari, I., Aksakalli, V., Aydogdu, V., Kum, S., 2013. Optimal ship navigation with safety distance and realistic turn constraints. European Journal of Operational Research 229 (3), 707–717.

[2] Austin, M., 2005. Creating a gis from noaa electronic navigational charts. In: OCEANS, 2005. Proceedings of MTS/IEEE. IEEE, pp. 839–841.

[3] Belker, T., Schulz, D., 2002. Local action planning for mobile robot collision avoidance. In: Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on. Vol. 1. IEEE, pp. 601–606.

[4] Carignan, K. S., Taylor, L. A., Eakins, B. W., Warnken, R. R., Lim, E., Medley, P. R., 2009. Digital elevation model of central oregon coast: procedures, data sources and analysis.

[5] Clementini, E., Di Felice, P., Van Oosterom, P., 1993. A small set of formal topological relationships suitable for end-user interaction. In: International Symposium on Spatial Databases. Springer, pp. 277–295.

[6] Clementini, E., Sharma, J., Egenhofer, M. J., 1994. Modelling topological spatial relations: Strategies for query processing. Computers & graphics 18 (6), 815–822.

[7] Decker, B. L., 1986. World geodetic system 1984. Tech. rep., Defense Mapping Agency Aerospace Center St Louis Afs Mo.

[8] Fiorini, P., Shiller, Z., 1993. Motion planning in dynamic environments using the relative velocity paradigm. In: ICRA (1). pp. 560–565.

[9] Fox, D., Burgard, W., Thrun, S., 1997. The dynamic window approach to collision avoidance. IEEE Robotics & Automation Magazine 4 (1), 23–33.

[10] GDAL/OGR contributors, 2018. GDAL/OGR Geospatial Data Abstraction software Library. Open Source Geospatial Foundation.
URL `http://gdal.org`

[11] Hagen, I. B., 2017. Collision avoidance for asvs using model predictive control. Master's thesis, NTNU.

[12] Hart, P. E., Nilsson, N. J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics 4 (2), 100–107.

[13] Hecht, H., 2006. The electronic chart: functions, potential and limitations of a new marine navigation system. Geomares publishing.

[14] IMO, 1972. Colregs - international regulations for preventing collisions at sea. Tech. rep., International Maritime Organization.

[15] International Hydrography Bureau Monaco, November 2000. IHO transfer standard for digital hydrographic data. Tech. rep., International Hydrography Organization.

[16] Johansen, T. A., Perez, T., Cristofaro, A., 2016. Ship collision avoidance and colregs compliance using simulation-based control behavior selection with predictive hazard assessment. IEEE transactions on intelligent transportation systems 17 (12), 3407–3422.

[17] Khatib, O., 1986. Real-time obstacle avoidance for manipulators and mobile robots. In: Autonomous robot vehicles. Springer, pp. 396–404.

[18] LaValle, S. M., 2006. Planning algorithms. Cambridge university press.

[19] Mąka, J., Magaj, J., 2012. Data extraction from an electronic s-57 standard chart for navigational decision systems. Zeszyty Naukowe/Akademia Morska w Szczecinie, 83–87.

[20] Niu, H., Lu, Y., Savvaris, A., Tsourdos, A., 2016. Efficient path planning algorithms for unmanned surface vehicle. IFAC-PapersOnLine 49 (23), 121–126.

[21] Park, M. G., Jeon, J. H., Lee, M. C., 2001. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. In: Industrial Electronics, 2001. Proceedings. ISIE 2001. IEEE International Symposium on. Vol. 3. IEEE, pp. 1530–1535.

[22] Park, M. G., Jeon, J. H., Lee, M. C., 2001. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. In: Industrial Electronics, 2001. Proceedings. ISIE 2001. IEEE International Symposium on. Vol. 3. IEEE, pp. 1530–1535.

[23] Peitz, S., Dellnitz, M., 06 2018. A survey of recent trends in multiobjective optimal control âĂŞ surrogate models, feedback control and objective reduction. Mathematical and Computational Applications 23.

[24] Statheros, T., Howells, G., Maier, K. M., 2008. Autonomous ship collision avoidance navigation concepts, technologies and techniques. The Journal of Navigation 61 (1), 129–142.

[25] Stenersen, T., 2015. Guidance system for autonomous surface vehicles. Master's thesis, NTNU.

Appendices

Appendix A

COLREGS Selection

A.1 Rule 3 - General definitions

For the purpose of these Rules, except where the context otherwise requires:

(k). Vessels shall be deemed to be in sight of one another only when one can be observed visually from the other.

A.2 Rule 6 - Safe Speed

Every vessel shall at all times proceed at a safe speed so that she can take proper and effective action to avoid collision and be stopped within a distance appropriate to the prevailing circumstances and conditions.

A.3 Rule 8 - Actions to Avoid Collision

(b). Any alteration of course and/or speed to avoid collision shall, if the circumstances of the case admit, be large enough to be readily apparent to another vessel observing visually or by radar; a succession of small alterations of course and/or speed should be avoided.

(d). Action taken to avoid collision with another vessel shall be such as to result in passing at a safe distance. The effectiveness of the action shall be carefully checked until the other vessel is finally past and clear.

A.4 Rule 11 - Application

Rules in this section apply to vessels in sight of one another.

A.5 Rule 13 - Overtaking

(a). Notwithstanding anything contained in the Rules of part B, sections I and II, any vessel overtaking any other shall keep out of the way of the vessel being overtaken.

(b). A vessel shall be deemed to be overtaking when coming up with another vessel from a direction more than 22.5 degrees abaft her beam, that is, in such a position with reference to the vessel she is overtaking, that at night she would be able to see only the sternlight of that vessel but neither of her sidelights.

A.6 Rule 14 - Head-On Situation

(a). When two power-driven vessels are meeting on reciprocal or nearly reciprocal courses so as to involve risk of collision each shall alter her course to starboard so that each shall pass on the port side of the other.

(b). Such a situation shall be deemed to exist when a vessel sees the other ahead or nearly ahead and by night she could see the masthead lights of the other in a line or nearly in a line and/or both sidelights and by day she observes the corresponding aspect of the other vessel.

A.7 Rule 15 - Crossing Situation

When two power-driven vessels are crossing so as to involve risk of collision, the vessel which has the other on her own starboard side shall keep out of the way and shall, if the circumstances of the case admit, avoid crossing ahead of the other vessel.

A.8 Rule 16 - Action by Give-Way vessel

Every vessel which is directed to keep out of the way of another vessel shall, so far as possible, take early and substantial action to keep well clear.

A.9 Rule 17 - Action by Stand-On Vessel

(a). (i). Where one of two vessels is to keep out of the way the other shall keep her course and speed.

(ii). The latter vessel may however take action to avoid collision by her manoeuvre alone, as soon as it becomes apparent to her that the vessel required to keep out of the way is not taking appropriate action in compliance with these Rules.

(b). When, from any cause, the vessel required to keep her course and speed finds herself so close that collision cannot be avoided by the action of the give-way vessel alone, she shall take such action as will best aid to avoid collision.