Local Navigation for Autonomous Maneuvering of Marine Vessels in Broken Ice Fields

by

Rodrigue de Schaetzen

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2024

© Rodrigue de Schaetzen 2024

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Rodrigue de Schaetzen was the sole author for Chapters 1, 2, 5, and 7 which were written under the supervision of Prof. Stephen L. Smith. This thesis includes first-authored peer-reviewed material that has appeared in conference proceedings published by the Institute of Electrical and Electronics Engineers (IEEE).

Research presented in Chapters 3, 4, and 6:

In Chapter 3, Rodrigue de Schaetzen and Alexander Botros collaboratively designed the problem formulation described in [26] and later refined for this thesis by Rodrigue de Schaetzen.

In Chapter 4, preliminary versions of the navigation framework, cost function, and heuristic were collaboratively designed by Rodrigue de Schaetzen and Alexander Botros described in [26]. Each of these components has been expanded on for this thesis by Rodrigue de Schaetzen. The code for the implementation of methods was written exclusively by Rodrigue de Schaetzen.

In Chapter 6, the real-world experiments were conducted by Rodrigue de Schaetzen, Kevin Murrant, and Robert Gash. All processing and analysis of experiment data was done by Rodrigue de Schaetzen.

Citations:

R. de Schaetzen, A. Botros, R. Gash, K. Murrant and S. L. Smith, "Real-Time Navigation for Autonomous Surface Vehicles In Ice-Covered Waters," 2023 IEEE International Conference on Robotics and Automation (ICRA), London, United Kingdom, 2023, pp. 1069-1075, doi: 10.1109/ICRA48891.2023.10161044. [26]

Abstract

The interest in ship navigation in the Polar regions, particularly the Arctic, is rapidly growing as a result of the increasing accessibility to these remote areas caused by rising global temperatures. The reduction in the extent of Arctic sea ice is gradually providing alternative shipping routes for global trade, which are significantly shorter in distance and can lead to lower emissions and lower costs associated with fuel consumption and transit time. Despite the range of potential benefits, ice navigation introduces numerous risks and challenges to safe ship operation due to the distinctly hazardous environment. Ice conditions often require ships to reduce speed and deviate from the main course to avoid maneuvers that are more likely to cause damage to the ship. In addition, broken ice fields are becoming the dominant ice conditions encountered in the Arctic, where the effects of collisions with ice on the ship are highly dependent on where contact occurs and on the particular features of the collided ice floes, including ice thickness, size, and shape. While existing works have shown success in computing global routes that minimize costs associated with ice navigation such as the average ice-induced forces exerted on the ship, none have considered the problem of optimizing ship navigation at the local level where costs are computed based on individual ship-ice collisions.

In this thesis, we present a framework for autonomous navigation of ships operating in ice floe fields. Trajectories are computed in a receding-horizon manner, where we frequently perform planning updates given updated ice-field data. During a planning iteration, we simplify the trajectory planning problem by assuming a fixed nominal speed that is safe with respect to the current ice conditions, and compute a smooth reference path that minimizes a collision cost. We formulate a novel cost function that minimizes the kinetic energy loss of the ship from ship-ice collisions and incorporate this cost as part of our lattice-based path planner. Solutions computed by the lattice planning stage are then used as an initial guess in our proposed optimization-based improvement step to compute a locally optimal path. We conducted extensive experiments to validate our approach both in simulation and in a physical test-bed managed by the National Research Council Canada (NRC).

Acknowledgments

First and foremost, I would like to thank my supervisor Stephen L. Smith for his support during my time as a Master's student. I feel very lucky to have been given the opportunity to join his lab and to have been matched with such an interesting project. To say that I have learned a great deal over the past three years would be an understatement, and I owe much of my development as a researcher to Steve's exceptional guidance and role as an advisor.

I attribute part of my growth (both personal and academic) to the many great friends I made in Waterloo. I am particularly grateful to fellow lab members Shamak Dutta, Megnath Ramesh, and Jack Dai. Together we shared countless fun moments, including ping pong sessions in E7 and weekly pints at the Graduate House followed by three-hour rounds of CATAN. I am also grateful for the numerous people I met from communities around the university and Waterloo in general, most notably the UWaterloo Triathlon Club, the UWaterloo Muay Thai Club, and the Grand River Rocks Climbing Gym.

I would like to thank Alexander Botros for his guidance during the initial phase of this project. His support was crucial in helping me navigate important technical details and taught me the importance of rigor in academic writing.

I would also like to thank Kevin Murrant and Robert Gash from the NRC. My two visits to St. John's were a few of the key highlights of my experience, and it was amazing to see my research materialize through the live experiments at the NRC research facility.

Finally, I would like to thank my parents, maman et papa, and my siblings, Constance, Apolline, Grace, and Esteban. I am constantly reminded how fortunate I am to have such an incredible family. Thank you.

Dedication

To my parents, my three sisters, and my brother.

Table of Contents

Author's Declaration ii

Statement of Contributions iii

Abstract iv

Acknowledgments v

Dedication vi

List of Figures x

List of Tables xiii

1 Introduction 1

1.1 Thesis Contributions 3

1.2 Thesis Outline 4

2 Literature Review 5

2.1 Ship Autonomy for Ice Navigation 5

2.2 General Ship Autonomy 6

3 Background **8**

3.1 Preliminaries . . . 8

3.2 Navigation Problem . . . 10

4 Navigation Framework **12**

4.1 Overview . . . 12

4.2 Path Planning using a State Lattice . . . 13

4.3 Cost Function . . . 16

4.3.1 Costmap Representation of the Environment . . . 16

4.3.2 Collision Cost . . . 17

4.3.3 Ice Concentration Cost . . . 21

4.4 Heuristic . . . 22

4.4.1 Nonholonomic without Obstacles . . . 23

4.4.2 Obstacles-only . . . 25

5 Optimization-Based Path Improvement Step **26**

5.1 Scalar Cost Field . . . 27

5.2 Body Points . . . 27

5.3 Transcription and Solver . . . 30

6 Experiments **33**

6.1 Simulation Experiments . . . 33

6.1.1 Vessel model and control system . . . 34

6.1.2 Physics simulation . . . 34

6.1.3 Ice field generation . . . 36

6.1.4 Metrics . . . 39

6.1.5 Baselines . . . 39

6.1.6 Lattice planner parameters . . . 40

6.1.7 Results 42
6.2 Real-World Experiments 46
6.2.1 Trial difficulty evaluation 48
6.2.2 Metrics 49
6.2.3 Lattice planner parameters 52
6.2.4 Results 52

7 Conclusion **55**
7.1 Limitations and Future Work 55

References **57**

List of Figures

1.1 Snapshot taken during preliminary testing done with a model scale vessel in the 76 m × 12 m ice tank located at the National Research Council Canada (NRC) research facility in St. John's, Newfoundland and Labrador. 3

3.1 The two coordinate systems used to describe ship motion in 3 DoF. 9

3.2 Depiction of the navigation problem of interest for a ship navigating a cluttered ice channel. 10

4.1 Example of a state lattice and control set. The lattice vertices correspond to the intersection of grid lines and the grid cell size illustrates the resolution of the discretized plane. In this example, a lattice vertex consists of a tuple (x, y, ψ) which means that each intersection of grid lines represents a collection of vertices with the same x, y values. The red curves show the result of concatenating the control set $\mathcal{P}$ with each primitive $\boldsymbol{\pi} \in \mathcal{P}$ shown in black. 14

4.2 The swath (grey grid cells) of a sample path (red line) and ship footprint (black outline). In this example, each cell in the discretized channel represents a region of size 2 m × 2 m. 17

4.3 (Left) The collision model of two disk-shaped bodies used to derive the cost function. (Right) An example of a costmap with a grid cell size of 1 m × 1 m. Brighter colors indicate higher cost. The centroid of each ice floe is indicated by the red 'x' and the bounding circles are shown in light blue. In this example, the ice thickness and density is the same across ice floes meaning the size of the polygon corresponds to its mass. 18

4.4 Diagram depicting the geometry of the Dubins path to an infinite line $\mathcal{G}_\infty$ for two possible cases. 24

5.1 A simple scenario to illustrate the problem with defining body points exclusively on the outline of the ship body. Two candidate paths are considered, $\boldsymbol{\pi}_a(s)$ and $\boldsymbol{\pi}_b(s)$, in the top and bottom parts of the figure, respectively. We examine the cost of these two paths by considering the contribution to the cost from two subsets of $\mathcal{B}$: $\{\boldsymbol{b}_1, \boldsymbol{b}_2\}$ and $\{\boldsymbol{b}'_1, \boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_j\}$, depicted as cyan dots. The corresponding paths $\boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b}_j)$ of the body points are shown with the cyan dotted lines. We see that in the bottom scenario, the magenta region of the cost field is disproportionately penalized from $\{\boldsymbol{b}'_1, \boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_j\}$ compared to the top scenario. 29

5.2 Three possible options to define the set $\mathcal{B}$. The dots represent body points sampled from the respective $\mathcal{B}$ and the cyan dotted lines show the body point paths. 30

5.3 Illustration of the components involved to derive (5.6). The square grid shows the discrete costmap with resolution Δ_{grid}. Body points are shown for two consecutive poses, $\boldsymbol{\eta}_i, \boldsymbol{\eta}_{i+1}$, where the points outlined in magenta are overlapping. For the sake of visualization, everything here is nicely aligned with the costmap grid. 31

6.1 The dimensions, bow geometry, and actuator configuration of the vessel considered in the simulation experiments. The origin of the vessel is depicted by the red 'x'. 35

6.2 Sample ice fields generated for the simulation experiments. Four different ice field concentrations are shown: 20%, 30%, 40%, and 50% ice coverage (top to bottom). 38

6.3 Primitives generated for the full-scale simulation experiments. There are 45 and 43 primitives for the 45 for axis aligned and 43 for non-axis aligned . . 41

6.4 Visualization of all impact forces and the corresponding contact-points from the ship-ice collisions that occurred during the simulation trials. Each vector represents the magnitude and direction of forces experienced by the ship's hull (grey outline). In total, the straight trials, skeleton trials, and lattice trials logged 11.3 million, 13.0 million, and 12.3 million collisions, respectively. 43

6.5 Results from evaluating our proposed improvement step. We show the results for the mean and maximum impact force metrics in the two methods. 44

6.6 Four snapshots taking from a sample simulation trial in a 40% concentration ice field using our proposed method. Here, we show both the initial plan computed in the lattice planning stage (cyan dotted line) and the optimized path (red line). Visually, we see that the red path tends to be better in terms of both the path length and the ship-ice collisions that occur along it. The actual path of the ship is displayed in blue and the green dotted line shows the intermediate goal $\mathcal{G}^{\text{int}}$. 45

6.7 Diagram of the experimental setup for our real world experiments. 47

6.8 Side view of the setup in the offshore engineering basin (OEB). 47

6.9 First frame from three sample trials with the corresponding difficulty level and mean difficulty score (in brackets) given the starting ship position (red star) and ice field configuration. 49

6.10 Total work done on the ice compared across planner and difficulty level. Work was computed using the object tracking data and (6.11). 53

6.11 Four snapshots from a sample lattice trial showing the camera image on the left and the costmap on the right. 54

List of Tables

6.1 Physics and other parameters used for our simulator. 37

6.2 Results from the simulation experiments for all 1200 trials. Scores are averaged across 100 trials for a particular ice concentration and planner. 42

6.3 Number of trials for each planner and difficulty level. 48

6.4 Results from experiments done in the model scale ice field. 53

Chapter 1

Introduction

Ice navigation and the associated growing economic incentives have recently become of significant interest to the maritime sector. Commercial shipping, for instance, has increasingly used the Arctic shipping routes in the last decade due to a reduction in ice cover [20]. Although still far from wide adoption, the Northern Sea Route (NSR) and the Northwest Passage (NWP) offer substantial benefits compared to standard international trade routes by providing a shortcut through the Arctic for certain journeys between Europe, Asia, and America. These benefits include a decrease in emissions and a reduction in costs attributed to shorter travel times, reduced travel distances (up to 40% for the NSR and NWP), and lower fuel consumption [39, 58, 5]. As the climate continues to warm, new Arctic shipping routes are expected to become viable (e.g., the Transpolar Sea Route), and navigation in the Arctic will likely become accessible for longer periods of time throughout the year [47, 46].

Despite these opportunities, the Arctic and Polar regions in general still present considerable challenges and risks to safe and efficient navigation due to seasonally and regionally varying sea ice [74, 35, 51]. Originating from freezing seawater, sea ice conditions are affected by a range of different factors, both environmental (e.g., ocean currents, wind, and temperature) and operational such as ship traffic and icebreaker assistance. This results in a wide variety of possible sea ice features that may be encountered during ship transit, each type presenting its own set of associated hazards [75]. Certain ice conditions, for example, pose a greater risk of ship besetting (i.e., the ship loses the ability to maneuver in the ice, effectively being trapped) and structural damage to the ship from ship-ice collisions, especially while navigating at excessive speeds [55, 35, 78]. In addition, other potential hazards, such as poor visibility and glacier ice, which includes icebergs, further complicate ice navigation. As a result, ships operating in ice tend to be specially designed

to manage some of these hazards and are equipped with specialized equipment such as sensors to detect the ice environment, while crew members, particularly the ship captain, often require very specific training [35, 75].

Addressing the challenges of ship navigation in ice-covered waters, therefore, fits well with the broader ongoing push toward greater ship autonomy and more intelligent systems to improve maritime safety and efficiency [30, 85, 67], particularly for navigating complex environments [63, 8, 10]. The increase in research interest in this area is also well aligned with the rapidly growing and evolving fields of optimization, control, robotics, and machine learning (e.g. [37, 12]). In the case of ice navigation, significant research effort has gone into modeling ship-ice interactions to estimate the ice-induced forces experienced by the ship [88, 45, 25, 24, 89, 80, 60], and global planning tools that compute optimized shipping routes (which may span several thousand kilometers) based on a set of specified objectives and constraints [83, 78, 21, 58, 82, 55, 52]. While both of these areas are still considered open research problems, few studies consider the task of local planning where maneuvers are planned over a short horizon and optimized for the ice conditions surrounding the current position of the ship [13].

In this thesis, we address this research gap by tackling the problem of local navigation for autonomous maneuvering of ships in ice-covered waters. We make the following key assumptions regarding our problem of interest: 1) the environment consists of broken sea ice, 2) the ice environment is too concentrated to achieve collision-free maneuvers, and 3) the ship is designed for ice navigation but has limited protection against ship-ice collisions (i.e., the ship is not an icebreaker). In contrast to level ice where continuous sheets are formed on the ocean surface requiring breaking of the ice, broken ice fields result in a much wider variety of possible ship-ice interactions [74]. Moreover, the most prevalent sea ice conditions expected to be encountered in the Arctic are ice floe fields [57, 81], and are the particular ice environments considered in this work. An ice floe field is defined as a collection of discrete floating ice pieces, called ice floes, which may be characterized by their thickness, size, and shape. Differences among floes in a given ice field often vary greatly in terms of size and shape [36, 90]. As a result, the collision responses of ship-ice interactions vary significantly [57, 61, 24] and generally involve the ice floe being partially crushed and then pushed by the ship, leading to a loss of kinetic energy [51]. An example of an ice floe field (albeit at model scale) can be seen in the snapshot shown in Figure 1.1, taken during our preliminary experiments performed at the National Research Council Canada (NRC) research facility in St. John's, Newfoundland and Labrador. This example provides a good illustration for the three key assumptions we make regarding our specific navigation problem. The snapshot also suggests that naive strategies like following a straight path may not always be desirable or, potentially in this case, even achievable without breaking

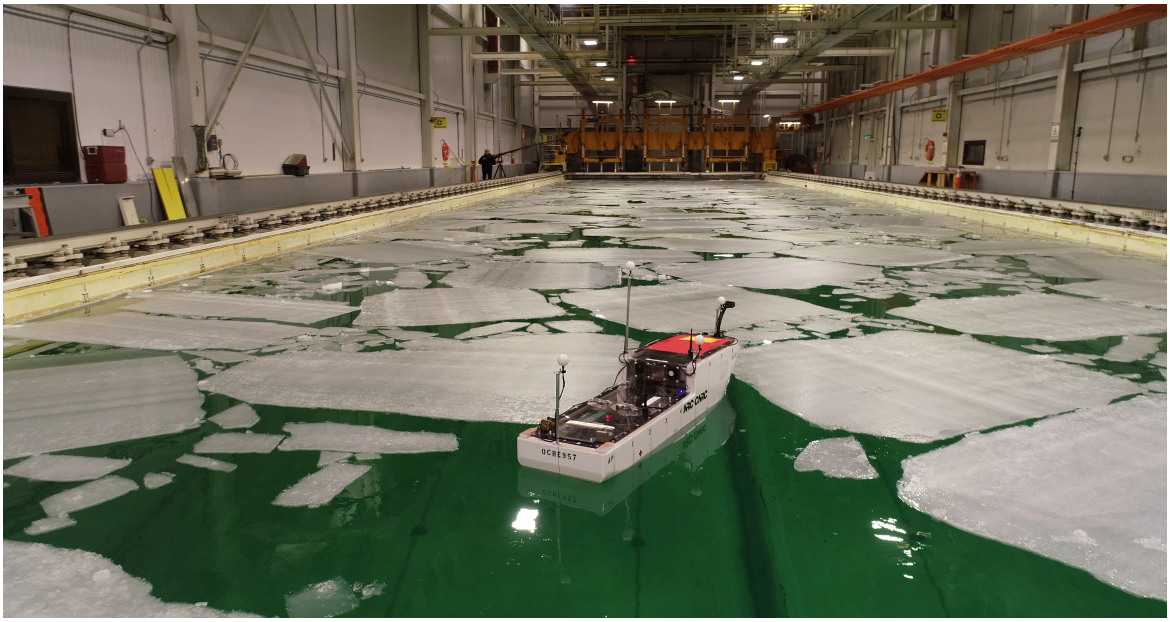


Figure 1.1: Snapshot taken during preliminary testing done with a model scale vessel in the 76 m $\times$ 12 m ice tank located at the National Research Council Canada (NRC) research facility in St. John's, Newfoundland and Labrador.

through the ice or performing additional maneuvers. The problem of local navigation in broken ice fields thus requires a method for efficiently finding a sequence of maneuvers that can be executed by the ship and are optimized in terms of the predicted costs associated from the ship-ice collisions. In this work, we provide a solution to this task by considering a local planning problem in which collisions along a candidate plan are evaluated from a kinetic energy loss perspective based on the currently detected ice conditions ahead of the ship. In our experiments, we show that our approach significantly reduces both the average and maximum impact forces experienced by the ship, as well as generally reducing overall total energy usage compared to naive navigation strategies.

1.1 Thesis Contributions

The following are the key contributions of this thesis:

- We present a framework for efficient real-time ship navigation of broken ice fields where low-cost trajectories are planned in a receding horizon manner and are fre-

quently updated using the latest ice field data. We formulate an objective function for the total collision cost based on a kinetic energy loss formulation of the ship-ice interaction. This cost function is employed within a graph search framework, adapting state lattice motion planning to discretize the space of possible ship maneuvers into a finite set of smooth reference paths that the ship is able to follow.

- We propose an optimization-based improvement step to locally refine the lattice planner solution as part of a two-stage planning pipeline. The discrete representation of the total collision cost from the first stage is transformed into a suitable differentiable function that can be minimized to a local minimum using popular gradient-based optimization techniques.

- We present results from extensive experiments performed both in simulation and in the model-scale setup at the NRC research facility. To address the lack of open-source physics simulators for ships in broken ice, we present a Python-based 2D simulator for efficient modeling of ship-ice interactions and release all code and documentation[1]. From our collection of 1200 simulation trials that span more than a thousand kilometers of ship transit in varying ice field conditions, our proposed method is empirically validated in terms of reducing the impact forces from ship-ice collisions and the kinetic energy loss of the ship leading to safer and more efficient navigation. In addition, these results are further validated through real-world experiments. The live physical tests proved to be the first successful attempts at fully autonomous navigation across the model-scale ice field with a model ship at the NRC.

1.2 Thesis Outline

The remainder of this thesis is organized into six chapters. In Chapter 2, we review relevant work in ship planning, control, and navigation, as well as provide an overview of related navigation tasks beyond maritime contexts. Chapter 3 provides a formal definition of the planning problem of interest, while in Chapter 4, we introduce our navigation framework for autonomous local navigation in broken ice fields. The solution computed by the lattice-based planner described in Chapter 4 is refined by our optimization-based improvement step proposed in Chapter 5. In Chapter 6, we describe the results and experimental setups for our simulation and real-world experiments. Finally, we provide a discussion on the various limitations of our proposed method and suggest important ways to improve and extend this work in Chapter 7.

[1] https://github.com/rdesc/Autonomous-Ship-In-Ice

Chapter 2

Literature Review

We review existing work relevant to our problem of interest. We start by highlighting work on ship planning for ice navigation, including the few works that consider the local navigation problem. We then discuss related work in trajectory and motion planning for ship navigation in contexts beyond our particular setting.

2.1 Ship Autonomy for Ice Navigation

Many researchers have considered the task of optimizing a global route for ice navigation [83, 82, 55, 68, 74, 21, 66, 91, 52, 78, 33]. The goal here is to compute a sequence of waypoints from a starting position to a target destination that minimizes an objective function subject to a set of constraints. These routes are computed online and updated as necessary to provide ship captains with real-time guidance. However, as previously mentioned, these routes only provide navigational guidance at the global level and are optimized based on a coarse representation of the environment. For example, in [78, 52, 55, 33], a discrete mesh is generated for each modality that describes the environmental conditions. Each cell in a particular mesh represents a subregion of the environment and is assigned a value corresponding to the specific input data (e.g., ocean currents, sea ice concentration, and ocean depth information) representing that region. Using ship performance models such as [74, 22], the mesh representation of the environment can be converted to a discrete map of the navigation costs associated with each subregion. Thus, the route planning problem can be efficiently solved by generating a graph from this discrete costmap. In our work, we follow a similar approach for representing the navigation cost, where we assign a collision cost to each cell in a fine grid representing

the local ice conditions. Our cost function is also motivated by the objective functions typically considered in route planning but considers the cost at the level of individual ship-ice collisions along the path. Both the route planner and the local planner systems are required to achieve full ship autonomy [13].

Few studies consider the local planning task for ice navigation. In total, we found four existing works that tackle this problem [2, 5, 43, 34]. In [2, 5, 43], the authors propose a method for computing collision-free paths in broken ice environments. Although the authors of [2, 5] consider the turning radius constraints of the ship, the collision-free assumption limits the practical relevance of these works in ice navigation. Recall that one of the core assumptions made in this work is that the ice environment generally does not allow for collision-free paths. The final local navigation approach we found is the shortest open-water planning approach described in [34]. Here, the authors leverage the popular image processing technique, called morphological skeleton, to generate a topological representation of the open water area given an overhead snapshot of the ice field. The information in the processed image is then extracted to build a graph from which a path can be computed. Among these four related works [2, 5, 43, 34], the most practical method is [34] since it can easily be extended to high-concentration ice fields. As a result, we considered the skeleton planner as one of the baselines in our experiments.

2.2 General Ship Autonomy

A larger body of work considers the broader task of autonomous ship operation in non-ice environments [19, 96, 53, 7, 6, 8, 77, 9, 76, 73]. This includes urban waterways [77], harbors [10, 8], and shorelines [19]. In [77], a receding-horizon path planner is proposed for an autonomous ship operating in contrained urban environments. Obstacles in the vicinity of the ship are detected using sensors mounted on-board, similar to how we expect the local ice field conditions to be captured in our problem. Since the obstacle environment is dynamic in nature, local planning is performed in an iterative fashion, where paths are continuously generated in real-time up to a small distance ahead of the ship's current position. These paths should therefore account for the kinodynamic constraints of the vessel for the path to be effectively followed. A challenge here is determining how to efficiently represent the space of feasible paths or, in the case of motion planning, the space of possible maneuvers that can be executed. Many methods address this problem by considering a discretization scheme. One popular technique is to define offline a finite set of optimized maneuvers as introduced in [71]. These maneuvers can be repeated in a particular sequence to produce a planned trajectory. The local planning problem can therefore be reduced to finding a

cost-minimizing sequence of these maneuvers using graph search techniques [54]. In this work, we adopt the so-called state lattice planning approach to produce smooth reference paths that consider the minimum turning radius constraints of the ship.

Treating local planning as a graph search problem has been shown to be an effective technique for autonomous ship navigation. In [8], the authors develop a two-stage optimization-based motion planner for ships navigating in confined environments (e.g., harbors). In the first stage, a lattice planner computes a trajectory by searching over a graph of motion primitives, while in the second stage the authors formulate an optimal control problem which is warm started with the lattice planning soluton. This two-stage approach ensures that the computed plan is locally optimal with respect to the continuous planning problem. A similar method is proposed in the two-stage trajectory planning framework described in [10], a key contribution being their novel encoding of polygonal obstacles into smooth and convex constraints. Smooth representations of the objective and constraints are common requirements for using fast optimization-based solvers. These works motivate our proposal for an optimization-based improvement step to enhance the path computed by the lattice planning stage in our ice navigation framework. We place particular emphasis on how we define a smooth objective function from the discrete collision cost formulated as part of the first stage.

Chapter 3

Background

3.1 Preliminaries

In this section, we introduce the notation and terminology used throughout this thesis. We follow the standard notation for motion of a marine craft described in [30] and [69]. We treat the ocean surface on which a ship moves as a 2D surface $\mathcal{W} \subseteq \mathbb{R}^2$. As such, it is sufficient to consider 3 degrees of freedom (DoF) for ship motion in the horizontal plane [30]. This corresponds to two translational movements, *surge* and *sway*, and one rotational movement, *yaw*. Surge describes motion along the longitudinal axis of the ship, while sway describes lateral (or side-ways) motion. Yaw, on the other hand, describes the rotation about the vertical axis.

We use two reference frames to characterize the motion of the ship: an inertial frame (x_n, y_n) and a body-fixed reference frame (x_b, y_b) with the origin o_b located at the center of gravity of the ship. The coordinates of o_b expressed in the inertial frame give the position of the ship $(x, y) \in \mathbb{R}^2$. The direction in which the front of the ship points is referred to as the heading angle $\psi \in [0, 2\pi) = \mathbb{S}^1$ which is defined by the angle from the inertial axis x_n to the body-fixed axis x_b. Together, the position and heading are referred to as the generalized position or the *pose* of the ship, denoted $\boldsymbol{\eta}$ where

$$\boldsymbol{\eta} = \begin{bmatrix} x & y & \psi \end{bmatrix}^\top \in \mathcal{C} = \mathbb{R}^2 \times \mathbb{S}^1. \tag{3.1}$$

The set $\mathcal{C}$ is considered the robot *configuration space* [54] of the ship and describes the set of all possible poses.

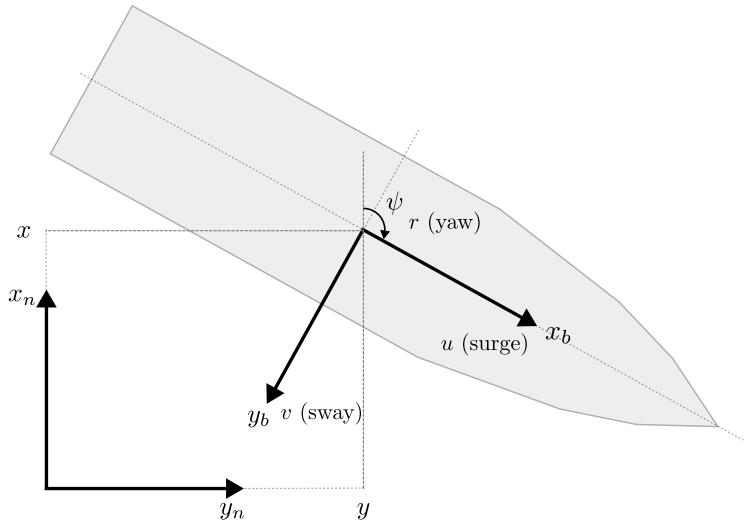


Figure 3.1: The two coordinate systems used to describe ship motion in 3 DoF.

The velocity of the ship is expressed in the body-fixed reference frame. The three components are surge velocity u, sway velocity v, and yaw rate r. As a vector, the generalized velocity is represented by

$$\boldsymbol{\nu} = \begin{bmatrix} u & v & r \end{bmatrix}^\top \in \mathbb{R}^3. \tag{3.2}$$

We can compute $\sqrt{u^2 + v^2}$ to obtain the speed $U \in \mathbb{R}$ representing the magnitude of the linear velocity. The vectors $\boldsymbol{\eta}$, $\boldsymbol{\nu}$ are related via the kinematic relationship

$$\dot{\boldsymbol{\eta}} = \mathbf{R}(\psi)\boldsymbol{\nu}, \tag{3.3}$$

where

$$\mathbf{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

is the rotation matrix. A visual summary of the notation described above is provided in Figure 3.1.

It is useful to mention the standard terminology used to describe the different parts of the ship referenced in this work. The watertight body of the ship is called the *hull* and the front and rear parts are called the *bow*, and *stern*, respectively. The design of the hull varies greatly between different ship classes. In the case of ice-going vessels, the hull and other parts of the ship are reinforced to a certain level. These ships are typically characterized by their assigned ice class, which indicates the extent of the strengthened hull and the overall maneuvering capabilities of the ship in different ice conditions.

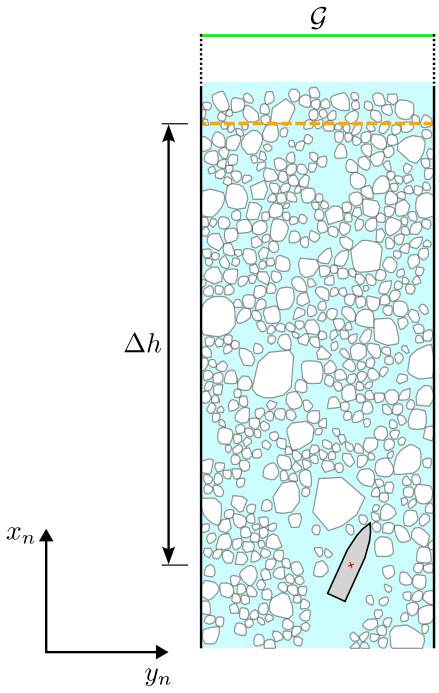


Figure 3.2: Depiction of the navigation problem of interest for a ship navigating a cluttered ice channel.

3.2 Navigation Problem

The problem we address is autonomous navigation of a ship transiting a cluttered ice channel while minimizing the cost associated with ship-ice collisions, namely the kinetic energy of the ship dissipated and transferred during collisions [57]. Our objective is to make forward progress along an ice channel $\mathcal{I} \subseteq \mathcal{W}$ which we model as a long rectangle. Without loss of generality, we assume that the channel length is aligned with the x_n axis of the inertial frame as in Figure 3.2. Note that while we treat the navigation area as rectangular throughout this work, it is easy to adapt our method to other environment shapes. For example, in the case of a curved ice channel, one option is to partition the environment into a set of rectangles and to assign an appropriate cost of navigating in the regions outside of the channel boundaries. We consider the navigation goal $\mathcal{G}$ to be a line segment perpendicular to the length of the channel and is defined in the configuration space $\mathcal{C}$ as

$$\mathcal{G} = \{\boldsymbol{\eta} \in \mathcal{C} \mid (x, y) \in \mathcal{I},\ x = x_{\text{goal}}\}. \tag{3.5}$$

Given m ice floes in the environment, we treat each floe as an obstacle with a *footprint* $\mathcal{O}_i \subset \mathcal{I}$ in the 2D environment, where $i = 1, \ldots, m$. Using the footprints, we can extract important ice properties (e.g. shape and size [93, 90]) of both the individual ice floes and of the overall distribution. The collection of all ice floes is denoted by $\mathcal{W}_{\text{obs}} = \{\mathcal{O}_1, \ldots, \mathcal{O}_m\} \subset 2^{\mathbb{R}^2}$ and the corresponding open water area $\mathcal{W}_{\text{free}}$ is defined as a set difference $\mathcal{I} \setminus \bigcup_i \mathcal{O}_i$. We assume that a top-down view of the ice field is actively being captured and processed via an ice image processing system and vision sensors that may be mounted on a mobile platform (e.g., unmanned aerial vehicle [92]) or onboard the ship [90]. The notion of a footprint is also useful for characterizing the region in $\mathcal{I}$ occupied by the ship's body at a given pose $\boldsymbol{\eta}$ and is denoted by $\mathcal{F}(\boldsymbol{\eta}) \subset \mathcal{I}$ (see grey polygon in Figure 3.2 for an example).

To navigate a ship through an ice field, three components are required: a *reference path* $\boldsymbol{\pi}(s) : \mathbb{R} \to \mathcal{C}$ parameterized by arc length $s \in [0, L_f]$, a *velocity profile* $\mathbf{v}(s) : \mathbb{R} \to \mathbb{R}^3$ that describes how the path is traversed over time, and a *controller* that tracks the path at the desired velocity. The tuple $(\boldsymbol{\pi}, \mathbf{v})$ is called a *trajectory*. To capture the navigation cost, we use an objective function J that penalizes the total path length L_f and a total *collision cost* C_f:

$$\begin{aligned} J(\boldsymbol{\pi}, \mathbf{v}) &= L_f + \alpha C_f \\ C_f &= \int_0^{L_f} c_{\text{obs}}\big(\boldsymbol{\pi}(s), \mathcal{W}_{\text{obs}}, \mathbf{v}(s), \mathcal{F}(\boldsymbol{\pi}(s))\big)\, ds. \end{aligned} \tag{3.6}$$

While the exact form of C_f will be refined later, we claim that a suitable function for the total collision cost consists of an integral that accumulates a cost c_{obs} as the ship, characterized by the footprint $\mathcal{F}$, follows a trajectory $(\boldsymbol{\pi}, \mathbf{v})$ in an ice channel with ice floes $\mathcal{W}_{\text{obs}}$. Here, the function $c_{\text{obs}} \geq 0$ is described in detail in Section 4.1 and $\alpha \geq 0$ is a tuning parameter that controls the trade-off between the two terms in the objective. Thus, we seek to solve the following problem.

Problem 3.2.1. *Given an initial pose $\boldsymbol{\eta}_{\text{cur}}$, a goal $\mathcal{G}$, obstacles $\mathcal{W}_{\text{obs}}$, a ship with a footprint $\mathcal{F}$, and an ice channel $\mathcal{I}$, compute a trajectory $(\boldsymbol{\pi}, \mathbf{v})$ from $\boldsymbol{\eta}_{\text{cur}}$ to $\mathcal{G}$ that minimizes J.*

In what follows, we propose a solution to this problem and describe how it is incorporated into a navigation system for real-time (re)planning.

Chapter 4

Navigation Framework

4.1 Overview

We present a navigation framework and provide an overview of our approach to solving Problem 3.2.1. The navigation framework is summarized in Algorithm 1, which takes as input a final goal $\mathcal{G}$, a receding horizon parameter Δh, and a tracking duration parameter Δt. To account for the large and evolving environment, we frequently replan a reference path and velocity profile for a controller in real-time. Replanning is performed over a moving horizon, as illustrated by the part of the channel lying below the orange dotted line in Figure 3.2.

At the start of each iteration, we obtain updated ice floe information $\mathcal{W}_{\text{obs}}$ from the vision system along with the current pose $\boldsymbol{\eta}_{\text{cur}}$ and the current velocity $\boldsymbol{\nu}$ of the ship (Lines 2, 4, 5) — we assume that other important ice properties may be queried at this stage, including the ice thickness, but it is often sufficient to treat these as constant for a given local region [1, 78, 74]. Using this new information, we update the horizon planning window and set a new limit U_{nom} on the nominal speed (Lines 6, 7). The intermediate goal $\mathcal{G}^{\text{int}}$ is set at a distance Δh ahead of the ship's position. Regarding U_{nom}, several works stress the importance of setting a suitable speed limit given the particular ice conditions and the particular ice class of the ship [35, 78, 55]. For example, in [78], the maximum acceptable speed is given as a function of several parameters of the ship and ice properties. This includes ice thickness, ice density, and sea ice concentration obtained by computing the ratio between the area covered by $\mathcal{W}_{\text{obs}}$ and the ice channel $\mathcal{I}$. In the next step of Algorithm 1, we run our proposed path planning algorithm (see Section 4.2), returning a smooth path $\boldsymbol{\pi}$ from the current ship pose $\boldsymbol{\pi}(0) = \boldsymbol{\eta}_{\text{cur}}$ to somewhere along the intermediate

Algorithm 1 Receding Horizon Navigation Framework

Input: $\mathcal{G}, \Delta h, \Delta t$

1: **while** `True` **do**
2: $\quad \boldsymbol{\eta}_{\text{cur}} \leftarrow$ current ship pose
3: $\quad$ **if** ship pose $\boldsymbol{\eta}_{\text{cur}}$ has not crossed $\mathcal{G}$ **then**
4: $\quad\quad \boldsymbol{\nu} \leftarrow$ current ship velocity
5: $\quad\quad \mathcal{W}_{\text{obs}} \leftarrow$ obstacles from vision system
6: $\quad\quad$ Set intermediate goal $\mathcal{G}^{\text{int}}$ a dist Δh ahead of $\boldsymbol{\eta}_{\text{cur}}$
7: $\quad\quad U_{\text{nom}} \leftarrow$ nominal speed given $\mathcal{W}_{\text{obs}}$
8: $\quad\quad \boldsymbol{\pi} \leftarrow$ path from $\boldsymbol{\eta}_{\text{cur}}$ to $\mathcal{G}^{\text{int}}$ given $\mathcal{W}_{\text{obs}}$, U_{nom}
9: $\quad\quad \mathbf{v}_{\text{nom}} \leftarrow$ velocity profile for $\boldsymbol{\pi}$ given $\boldsymbol{\nu}$, U_{nom}
10: $\quad\quad$ Send $(\boldsymbol{\pi}, \mathbf{v}_{\text{nom}})$ to controller to track for time Δt
11: $\quad$ **else**
12: $\quad\quad$ Exit loop

goal $\boldsymbol{\pi}(L_f) \in \mathcal{G}^{\text{int}}$ (Line 8). The planned path $\boldsymbol{\pi}$, current velocity $\boldsymbol{\nu}$, and speed limit U_{nom} are then passed to the velocity profile planning step (Line 9). In our planner, we use a speed profile with a linear ramp from ship speed U to U_{nom} with a constant acceleration parameter. In the final step, a controller tracks the planned trajectory $(\boldsymbol{\pi}, \mathbf{v}_{\text{nom}})$ for time Δt (Line 10). Here, we use a Dynamic Positioning control system which we discuss as part of the experimental setup in Section 6.1.1.

The steps described above are repeated until the ship reaches the goal $\mathcal{G}$. Note that in each planning iteration, the ice is treated as static when computing a trajectory $(\boldsymbol{\pi}, \mathbf{v}_{\text{nom}})$. However, we frequently perform planning updates to account for collisions and other disturbances, such as ocean current, affecting changes in the ice environment. As a result, we maintain low runtime in our path planning algorithm. The following sections describe how paths $\boldsymbol{\pi}$ are computed.

4.2 Path Planning using a State Lattice

In the previous section, we introduced our navigation framework and established a basis for how we solve Problem 3.2.1: the path $\boldsymbol{\pi}$ is computed assuming a maximum speed limit U_{nom} followed by a simple speed profile generation step. Solving the problem of interest therefore reduces to planning cost-minimized paths that can be effectively tracked by a ship moving at a speed $U \leq U_{\text{nom}}$.

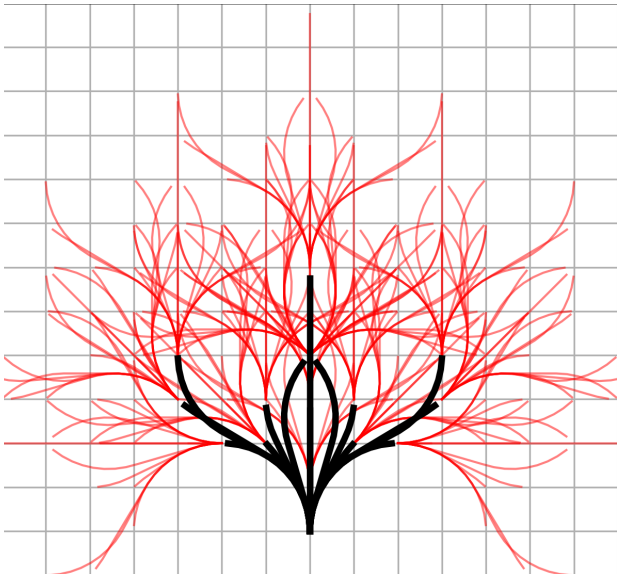


Figure 4.1: Example of a state lattice and control set. The lattice vertices correspond to the intersection of grid lines and the grid cell size illustrates the resolution of the discretized plane. In this example, a lattice vertex consists of a tuple (x, y, ψ) which means that each intersection of grid lines represents a collection of vertices with the same x, y values. The red curves show the result of concatenating the control set $\mathcal{P}$ with each primitive $\boldsymbol{\pi} \in \mathcal{P}$ shown in black.

We adopt a lattice-based planner from [71] where planning is cast as a search problem in a specially discretized space of the robot state and control spaces. A precomputed finite set of robot actions, typically characterized by a set of differential equations, are defined such that applying repeated actions generates a regularly repeating arrangement, or lattice, of states. More precisely, we require a *control set* $\mathcal{P}$ where for each *motion primitive* $\boldsymbol{\pi} \in \mathcal{P}$ we have a path that terminates at a state $p_1 \in L$ of the lattice L, if and only if $\boldsymbol{\pi}(0) = p_0 \in L$. This means, for an appropriately defined control set $\mathcal{P}$, we can construct a graph $G^{\mathcal{P}} = (L, E, w)$ with vertices L, edges E, and a function $w : E \to \mathbb{R}$ to assign weights to edges. Observe that each edge that connects a pair of vertices (p_0, p_1) is a result of rotating and translating (in robot configuration space) a particular motion primitive at the vertex p_0. In addition, since the control set is defined using a particular model of the robot motion characteristics, a sequence of concatenated primitives will by construction produce a *feasible* path, or in other words, a path that satisfies the robot model. We can therefore use popular graph search techniques to efficiently plan cost-minimizing feasible paths in the graph $G^{\mathcal{P}}$. The key properties of a state lattice are illustrated in Figure 4.1.

We discuss how the above-mentioned state lattice framework is applied to our path planning problem. Following several other works in ship navigation on a 2D surface [48, 73, 18, 59], we assume paths generated by a ship can be approximated by a unicycle model where the curvature κ is constrained to some limit $\kappa_{\max}$:

$$\begin{aligned} \dot{\boldsymbol{\eta}}(s) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} \cos(\psi) \\ \sin(\psi) \\ \kappa \end{bmatrix} \\ |\kappa| &\leq \kappa_{\max}. \end{aligned} \tag{4.1}$$

Here, the derivatives $\dot{(\cdot)}$ in the ODE are taken with respect to arc length s. We can set the limit $\kappa_{\max}$ based on the physical limits of the ship given the particular conditions in the environment and the target ship speed [35, 30]. This limit is typically expressed in terms of the minimum turning radius of the ship, $r_{\min}$:

$$r_{\min} = \frac{1}{\kappa_{\max}}. \tag{4.2}$$

Given the model in (4.1), we can consider the discretization of the configuration space $\mathcal{C}$. We generate a state lattice L by discretizing the plane $\mathbb{R}^2$ into a uniform grid and the unit circle $\mathbb{S}$ into uniformly spaced angles. Recalling our navigation framework summarized in Algorithm 1, in each path planning iteration, we build a graph $G^{\mathcal{P}} = (L, E, w)$ with a lattice L initialized at the current pose of the ship $\boldsymbol{\eta}_{\text{cur}}$. To define our motion primitives, we compute shortest paths between states $\boldsymbol{\eta} = [x\ y\ \psi]^\top$ for the unicycle model (4.1), known as Dubins' paths [29]. These paths are comprised of sequences of straight lines and circular arcs of radius $r_{\min}$. Ideally, we would like to have a minimum turning radius $r_{\min}$ that accurately reflects the current nominal speed U_{nom} from Algorithm 1. However, since the motion primitives are defined offline we can instead generate a suite of control sets with different minimum turning radii and then select online the best matching control set and $r_{\min}$ for the given U_{nom}. The control set $\mathcal{P}$ (or series of control sets with different $r_{\min}$) is generated using the method proposed in [15] which computes a control set that balances a trade-off between $|\mathcal{P}|$ and path optimality. For the weights of the graph, we set the function $w : E \to \mathbb{R}$ to be equal to $J_d(\boldsymbol{\pi}, U_{\text{nom}})$, our proposed discretized implementation of the cost function (3.6) described in detail in the next section. Finally, we use the A* graph search algorithm [40] to plan a path $\boldsymbol{\pi}$ from the current pose $\boldsymbol{\eta}_{\text{cur}}$ to the current goal $\mathcal{G}^{\text{int}}$. In Section 4.4 we present a heuristic to improve the performance of A* and in Section 5 we describe an optimization-based improvement step to locally refine the lattice planning solution.

4.3 Cost Function

We describe our implementation of the cost function J from (3.6) given a feasible path $\boldsymbol{\pi}$, obstacles $\mathcal{W}_{\text{obs}}$, and the current nominal speed U_{nom}. Recall the two terms in J: path length L_f of path $\boldsymbol{\pi}$ and total collision cost αC_f with weight $\alpha > 0$. If we assume ice-free conditions, the path length provides a reasonable model for the energy usage of the ship in a local region where other environment conditions are likely less variable. As a result, for the cost C_f we use the notion of total *kinetic energy* loss during ship-ice collisions to capture the additional cost incurred during ship navigation in an ice channel. We start by describing how we can efficiently compute the cost of a candidate path, $\boldsymbol{\pi}$, given obstacles, $\mathcal{W}_{\text{obs}}$, in a lattice planning framework.

4.3.1 Costmap Representation of the Environment

We discretize the channel $\mathcal{I}$ into a uniform grid with resolution Δ_{grid} where each grid cell is assigned an identifying tuple $\boldsymbol{k} \in \mathbb{N}^2$ corresponding to the center of the grid cell. Let $\mathcal{I}_d \subseteq \mathbb{N}^2$ denote the discretized ice channel. Each ice floe $\mathcal{O} \in \mathcal{W}_{\text{obs}}$ is mapped to a footprint $\mathcal{O} \subset \mathcal{I}_d$ and we denote $\mathcal{W}_{\text{obs},d}$ as the set of discrete obstacle footprints. Each grid cell $\boldsymbol{k}$ is an element of at most one obstacle footprint $\mathcal{O} \in \mathcal{W}_{\text{obs},d}$ and $\boldsymbol{k}$ is considered an element of an obstacle $\mathcal{O}$ if the continuous footprint representation covers any part of the cell. Using this discrete representation of the planar environment, we can generate a costmap [71] by assigning a collision cost to each grid cell. To compute the cost of being at the pose $\boldsymbol{\eta}$ we require an appropriate mapping from the configuration space to the costmap as well as a footprint $\mathcal{F}_d(\boldsymbol{\eta}) \subset \mathbb{N}^2$ defining the set of grid cells occupied by the ship body given the current position and heading. We use the scaling factor that determines the resolution of the costmap to scale the x and y coordinates, then round the values to the nearest grid cell. The footprint of the ship also allows us to check whether the ship is outside the bounds of the ice channel for a given pose $\boldsymbol{\eta}$. The total collision cost for a path $\boldsymbol{\pi}$ is computed as a sum over the set of grid cells $\mathcal{S}(\boldsymbol{\pi}) \subset \mathbb{N}^2$ swept by the ship footprint $\mathcal{F}_d$ from tracing the path $\boldsymbol{\pi}$. The set $\mathcal{S}(\boldsymbol{\pi})$ is referred to as the *swath* of the path $\boldsymbol{\pi}$, and we show an example in Figure 4.2. Hence, given the inputs $\mathcal{W}_{\text{obs}}$, U_{nom} from Algorithm 1, we define the total collision cost C_f as

$$C_f = \sum_{\boldsymbol{k} \in \mathcal{S}(\boldsymbol{\pi})} c_{\text{obs}}(\boldsymbol{k}, \mathcal{W}_{\text{obs},d}, U_{\text{nom}}). \tag{4.3}$$

The function $c_{\text{obs}}(\boldsymbol{k}, \mathcal{W}_{\text{obs},d}, U_{\text{nom}})$ is our proposed collision cost function presented in the following section. As a small detail, the ship footprint is no longer an input to c_{obs} as

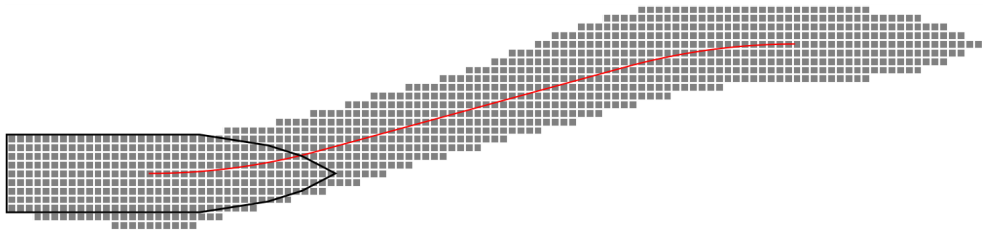


Figure 4.2: The swath (grey grid cells) of a sample path (red line) and ship footprint (black outline). In this example, each cell in the discretized channel represents a region of size 2 m $\times$ 2 m.

initially introduced in (3.6) since this is captured by the summation over the grid cells $\boldsymbol{k}$ in the swath $\mathcal{S}(\boldsymbol{\pi})$ given the footprint $\mathcal{F}_d$. The resulting cost function used to compute the edge cost between pairs of lattice vertices during graph search is as follows:

$$J_d(\boldsymbol{\pi}, U_{\text{nom}}) = L_f + \alpha C_f. \tag{4.4}$$

Proper tuning of the weight α is discussed in the experiment sections 6.1, 6.2. Note that in addition to precomputing a control set, we compute offline the corresponding path length and the corresponding swath of each motion primitive. This means that online we can translate / rotate the swath of a particular primitive as necessary to efficiently lookup the cost (4.3) of a particular edge in the graph. Special care is required to avoid double counting of collision costs, since there is generally an overlap of grid cells between the swaths of two concatenated primitives. Another important consideration is selecting an appropriate grid resolution to maintain sufficient accuracy in the spatial information of the ice floes which may consist of a range of different sizes and shapes.

4.3.2 Collision Cost

To formulate the collision cost function, we consider a simple 2D ship-ice collision model and show a derivation of the kinetic energy loss of the ship from the collision. We ignore more complex ship-ice interactions, such as floe-splitting and rafting behavior [36]. Although the model used here significantly simplifies the mechanics of a collision, it will enable us to efficiently capture sufficient detail in the collision cost function. We highlight the particular planning behaviors induced by our cost function at the end of this section.

Let m_{ship} and m_{ice} denote the mass of the ship and ice floe, respectively. We assume that ice floes have uniform density and thickness. Thus, the mass m_{ice} is given by the

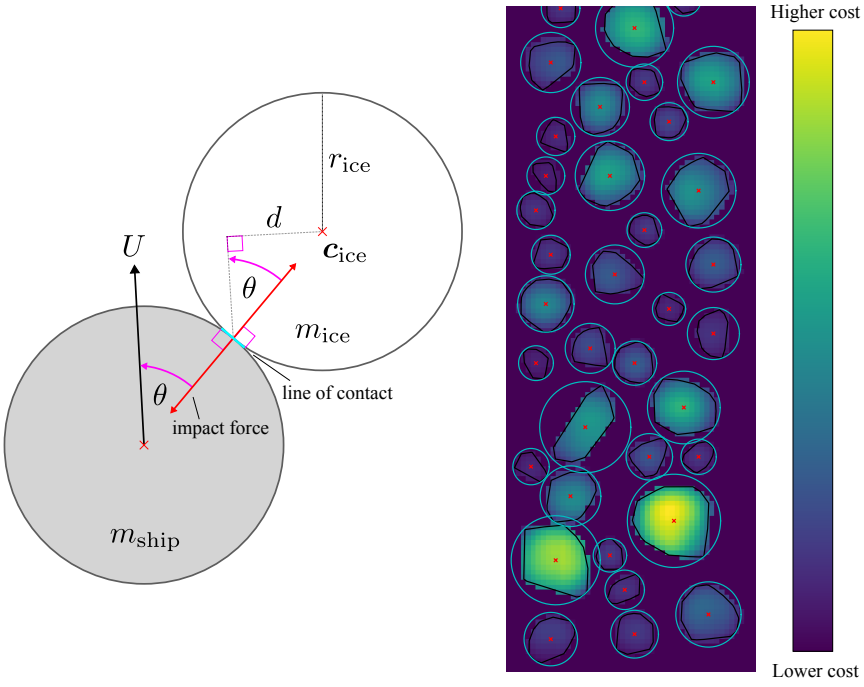


Figure 4.3: (Left) The collision model of two disk-shaped bodies used to derive the cost function. (Right) An example of a costmap with a grid cell size of 1 m $\times$ 1 m. Brighter colors indicate higher cost. The centroid of each ice floe is indicated by the red 'x' and the bounding circles are shown in light blue. In this example, the ice thickness and density is the same across ice floes meaning the size of the polygon corresponds to its mass.

product of density, thickness, and area of the ice floe [24], where the area is determined by the extent of its footprint $\mathcal{O} \in \mathcal{W}_{\text{obs}}$ (usually a convex polygon). Similar to the collision model proposed in [24, 25] and then experimentally validated in [3, 49], we reduce the collision to a one-body problem by making the following assumptions: the collision is of short duration, the momentum of the system is conserved (i.e., friction is ignored), and the impact force is normal to the line of contact between the two bodies (see Figure 4.3 (left)). In addition, the collision is assumed to be inelastic, where the maximum amount of kinetic energy of the system is lost to ice crushing [23, 24].

With this collision model, the change in kinetic energy ΔK_{sys} in the system is given by

$$\Delta K_{\text{sys}} = -\frac{1}{2} M_{\text{eq}} V_{\text{eq}}^2, \tag{4.5}$$

where M_{eq}, called effective mass, is the mass in the center of mass frame, and V_{eq} is the

relative velocity of the two bodies before the collision with respect to the line of contact. If we consider two disk-shaped bodies, the effective mass is only a function of the two masses and is therefore constant across collisions:

$$M_{\text{eq}} = \frac{m_{\text{ship}} m_{\text{ice}}}{m_{\text{ship}} + m_{\text{ice}}}. \tag{4.6}$$

From [23], the ice velocity is assumed to be negligible compared to the ship velocity $\boldsymbol{\nu}$. Hence, the relative velocity V_{eq} is a function of the ship speed U and the angle θ between the direction in which U is pointing and the vector normal to the line of contact:

$$V_{\text{eq}} = U \cos(\theta). \tag{4.7}$$

The angle θ is illustrated in Figure 4.3 (left). The negative sign in (4.5) indicates the system has lost kinetic energy, but we can equivalently consider the energy dissipated during the collision by considering $|\Delta K_{\text{sys}}|$. Next, we isolate the change in kinetic energy of the ship ΔK_{ship}.

The goal of our collision cost will be to minimize the kinetic energy loss of the ship due to the kinetic energy transferred to the ice and absorbed by ice crushing. Since the ice is treated as static prior to collision, it must hold that the change in kinetic energy of the ice ΔK_{ice} is greater than 0. In summary, we have the inequalities

$$\Delta K_{\text{sys}} < 0, \quad \Delta K_{\text{ice}} > 0, \quad \Delta K_{\text{ship}} < 0, \tag{4.8}$$

as well as the following relationship between the changes in kinetic energy for the ship, ice, and system:

$$\Delta K_{\text{sys}} = \Delta K_{\text{ice}} + \Delta K_{\text{ship}}. \tag{4.9}$$

After the collision has occurred, we treat the ice floe as having a final velocity equal to $V_{\text{eq}} m_{\text{ship}} (m_{\text{ship}} + m_{\text{ice}})^{-1}$. Therefore,

$$\begin{aligned} \Delta K_{\text{ice}} &= \frac{1}{2} m_{\text{ice}} \left(\frac{V_{\text{eq}} m_{\text{ship}}}{m_{\text{ship}} + m_{\text{ice}}} \right)^2 \\ &= \frac{1}{2} \frac{m_{\text{ice}} m_{\text{ship}}^2}{(m_{\text{ship}} + m_{\text{ice}})^2} (U \cos(\theta))^2 \end{aligned} \tag{4.10}$$

and from (4.5) (4.9)[1]:

$$|\Delta K_{\text{ship}}| = \frac{m_{\text{ice}} m_{\text{ship}} (m_{\text{ice}} + 2m_{\text{ship}})}{2(m_{\text{ice}} + m_{\text{ship}})^2} (U \cos(\theta))^2 \tag{4.11}$$

[1]The expression given in our preliminary work [26] contained a small error.

Suppose that the disk representing the ice floe has a radius of r_{ice} and a center located at $\boldsymbol{c}_{\text{ice}} \in \mathbb{R}^2$. The angle θ in the expression (4.11) can be written in terms of r_{ice} and the lateral distance d between $\boldsymbol{c}_{\text{ice}}$ and the center of the line of contact (this is also where the impact force is exerted) measured with respect to the component parallel to the direction of U. From Figure 4.3,

$$\theta = \arcsin\left(\frac{d}{r_{\text{ice}}}\right). \tag{4.12}$$

As a result, given a particular ship with constant mass m_{ship}, ΔK_{ship} from (4.11) can be written as a function of d, r_{ice}, m_{ice}, and U:

$$\begin{aligned}
&\Delta K_{\text{ship}}(d, r_{\text{ice}}, m_{\text{ice}}, U) = \\
&= \frac{m_{\text{ice}} m_{\text{ship}}(m_{\text{ice}} + 2m_{\text{ship}})}{2(m_{\text{ice}} + m_{\text{ship}})^2} \left[U \cos\left(\arcsin\left(\frac{d}{r_{\text{ice}}}\right)\right)\right]^2 \\
&= \frac{U^2 m_{\text{ice}} m_{\text{ship}}(m_{\text{ice}} + 2m_{\text{ship}})}{2(m_{\text{ice}} + m_{\text{ship}})^2} \left(\frac{r_{\text{ice}}^2 - d^2}{r_{\text{ice}}^2}\right), \quad d \in [0, r_{\text{ice}}].
\end{aligned} \tag{4.13}$$

Here, we have omitted the negative sign so that now ΔK_{ship} denotes the kinetic energy loss of the ship. Two key points to note are that the function ΔK_{ship} increases as the collision becomes more head-on ($d \to 0$) and as the ice mass m_{ice} increases. As expected, varying either of these variables would also increase the impact force exerted on the ship [35, 24, 93]. Using ΔK_{ship}, we describe the cost function c_{obs} from (4.4).

For each ice floe $\mathcal{O}_j \in \mathcal{W}_{\text{obs},d}$, let $\boldsymbol{c}_{\text{ice},j}$ denote the position of the centroid of $\mathcal{O}_j$, $r_{\text{ice},j}$ the radius of its bounding circle centered at $\boldsymbol{c}_{\text{ice},j}$, and $m_{\text{ice},j}$ its mass. The function $c_{\text{obs}}(\boldsymbol{k}, \mathcal{W}_{\text{obs},d}, U_{\text{nom}})$ for a grid cell $\boldsymbol{k}$ and current nominal speed U_{nom} uses the ship kinetic energy loss from (4.13):

$$c_{\text{obs}}(\boldsymbol{k}, \mathcal{W}_{\text{obs},d}, U_{\text{nom}}) = \begin{cases} \Delta K_{\text{ship}}(d', r_{\text{ice},j}, m_{\text{ice},j}, U_{\text{nom}}), & \exists \mathcal{O}_j \mid \boldsymbol{k} \in \mathcal{O}_j \\ 0 & \text{otherwise,} \end{cases} \tag{4.14}$$

where $d' = ||\boldsymbol{k} - \boldsymbol{c}_{\text{ice},j}||$ is the Euclidean distance from the grid cell $\boldsymbol{k}$ to the centroid of the obstacle $\mathcal{O}_j$. This function is well defined, since each cell can be occupied by at most one obstacle. Figure 4.3 (right) illustrates a sample costmap and shows the bounding circles and radii of the obstacles.

We discuss the various behaviors we expect from using (4.14) in our path planner in terms of the penalty applied to the swath $\mathcal{S}(\boldsymbol{\pi})$ of a path $\boldsymbol{\pi}$. Clearly, a larger penalty is applied if the swath overlaps ice floes with larger mass. In addition, the cost of the swath increases as the grid cells $\boldsymbol{k} \in \mathcal{S}(\boldsymbol{\pi})$ become closer to the centroid of an obstacle. This

is particularly visible in Figure 4.3 in the ice floes that are less circular (e.g., rectangular shaped), where the edges of the shape that are closer to the centroid are assigned a higher cost than the edges farther away. In practice, this means that paths with fewer head-on collisions with large ice floes are favored during planning, which typically results in the ship pushing the ice floes off to the side during navigation with few instances of sustained contact.

It is worth noting that, in contrast to typical cost functions used in navigation problems, (4.14) does not apply a penalty for being in proximity to obstacles that are not in collision with the path. In our particular problem, planning paths arbitrarily close to obstacle edges (particularly large ice floes) without penalty may not be desirable behavior, especially considering that we expect some error in tracking. To address this, a straightforward extension is to scale up the obstacle footprints $\mathcal{W}_{\text{obs}}$ by a small factor (e.g., 0.1) proportional to their size. This would effectively generate a cost buffer around the ice floes in the costmap. Regarding the potential issue of having overlap between multiple scaled-up obstacle footprints, the solution used here is to assign the maximum of the costs computed for a grid cell $\boldsymbol{k}$.

4.3.3 Ice Concentration Cost

Another extension of the collision cost function (4.14) formulated above is presented. Our goal is to account for scenarios where the ship displaces a greater amount of ice than the set of ice floes that were in direct contact with the ship. More formally, for a given path $\boldsymbol{\pi}$ and a corresponding swath $\mathcal{S}(\boldsymbol{\pi})$, there may exist a set of $n > 0$ obstacles $\{\mathcal{O}_j\}_{j=1}^{n} \subset \mathcal{W}_{\text{obs},d}$ where for each $\mathcal{O}_j$ we have $\mathcal{O}_j \cap \mathcal{S}(\boldsymbol{\pi}) = \emptyset$, and yet each $\mathcal{O}_j$ creates additional resistance to ship motion, resulting in an increase in kinetic energy loss ΔK_{ship}. It is easy to see that such scenarios are more likely to occur as the concentration of the ice field increases. Moreover, in [22] the relationship between the average ice resistance force exerted on the ship and the ice concentration $C_{\text{ice}} \in [0, 1]$ is modeled as a linear function in $(C_{\text{ice}})^{\beta}$ where $\beta \geq 1$ is a constant parameter determined by the geometry of the particular ship hull. This motivates our use of ice concentration as an additional factor in the cost to address the limitation present in the current cost function (4.14).

To assign an ice concentration cost to grid cells in the costmap, we first generate a binary occupancy grid of the ice channel from $\mathcal{W}_{\text{obs},d}$ indicating the pixels that are occupied by an ice floe. We then compute an image convolution between the binary image and a mean filter with kernel size $z \times z$, where z is odd. This means that for a given pixel $\boldsymbol{k} \in \mathbb{N}^2$ in the binary image, the mean filter will compute the average ice concentration for a region

of size $z \times z$ in pixel units centered around $\boldsymbol{k}$. For the purpose of local navigation, the kernel should therefore be appropriately sized so that at a given location the extent of the kernel can capture a collection of ice floes in a region that is still relatively local compared to the rest of the ice channel. The resulting image, $C_{\text{ice}} : \mathbb{N}^2 \to [0, 1]$, provides a 'smooth' representation of the ice concentration at every position in the discretized ice channel. This contrasts with the coarse grid or mesh representation [78], where each cell represents a large area and is assigned a single ice concentration value, which is more suitable for route planning. Although not particularly important for lattice planning, ensuring smoothness in the costmap (except at the boundaries of the obstacles) will help in the optimization-based improvement step described in the next section, where we generate a smooth differentiable function of the costmap.

As a final important detail, to prevent the loss of ice information at the boundaries of the image, we apply mirror padding to the binary image prior to computing C_{ice}. We also compute the image convolution such that C_{ice} is the same size as the costmap resulting in a one-to-one correspondence between the grid cells in the costmap and C_{ice}. The updated cost function c_{obs} computed for each cell $\boldsymbol{k}$ in the costmap is now defined as follows (the term for the kinetic energy loss is the same as before):

$$c_{\text{obs}}(\boldsymbol{k}, \mathcal{W}_{\text{obs},d}, U_{\text{nom}}) = \begin{cases} (C_{\text{ice}}(\boldsymbol{k}))^{\beta} \Delta K_{\text{ship}}(d', r_{\text{ice},j}, m_{\text{ice},j}, U_{\text{nom}}), & \exists \mathcal{O}_j \mid \boldsymbol{k} \in \mathcal{O}_j \\ 0 & \text{otherwise.} \end{cases} \tag{4.15}$$

4.4 Heuristic

We present an admissible heuristic h to improve the runtime of A* search during path planning. Given the two terms in our cost function (4.4), we can compute two separate heuristics, each admissible with respect to a term in the cost. Specifically, we require a function $h_1(\boldsymbol{\eta})$ such that it lower bounds the path length L_f and another function $h_2(\boldsymbol{\eta})$ such that it lower bounds the total collision cost C_f for all $\boldsymbol{\eta}$ given the current goal $\mathcal{G}^{\text{int}}$ and costmap. We can therefore do a summation to get an admissible function $h(\boldsymbol{\eta})$:

$$h(\boldsymbol{\eta}) = h_1(\boldsymbol{\eta}) + h_2(\boldsymbol{\eta}). \tag{4.16}$$

We refer to the two terms in h as the *nonholonomic without obstacles* heuristic [27, 10] and the *obstacles-only* heuristic, respectively.

4.4.1 Nonholonomic without Obstacles

At a high-level, we compute the length of the shortest path from $\boldsymbol{\eta}$ to an infinite line $\mathcal{G}_\infty$ that is collinear with the intermediate goal $\mathcal{G}^{\text{int}}$, subject to the unicycle model (4.1). Recall our definition of the goal in (3.5) with parameter x_{goal} denoting the constant x coordinate of the goal and recall our world coordinate frame shown in Figure 3.2. To characterize the proposed heuristic, we offer the following result:

Theorem 4.4.1 (Closed-form Heuristic)**.** *The shortest path from $\boldsymbol{\eta} = [x\ y\ \psi]^\top$ to the infinite straight line segment $\mathcal{G}_\infty$ with minimum turning radius $r_{\min}$ is a Dubin's path. Referencing Figure 4.4,*

(a) if $\mathcal{G}_\infty$ lies above the point $\boldsymbol{c} = [x_c\ y_c]^\top$, the path is of the form CS (circular arc C of radius $r_{\min}$, followed by a straight line segment S) and S intersects $\mathcal{G}_\infty$ at $\theta = \pi/2$ (Figure 4.4 (left));

(b) otherwise, the path is of the form C (Figure 4.4 (right)).

Given these two cases, $h_1(\boldsymbol{\eta})$ is the path length and is given analytically by:

$$h_1(\boldsymbol{\eta}) = \begin{cases} h_a(\boldsymbol{\eta}), & \text{if } x_c \leq x_{goal} \text{ (case a)}, \\ h_b(\boldsymbol{\eta}), & \text{otherwise (case b)} \end{cases},$$

where

$$h_a(\boldsymbol{\eta}) = r_{\min} \left| \frac{\pi}{2} - \phi \right| + x_{goal} - x_c$$

$$h_b(\boldsymbol{\eta}) = r_{\min} \left| \phi - \arccos\left(\frac{x_c - x_{goal}}{r_{\min}} \right) \right|$$

$$x_c = x + r_{\min} \left| \cos(\phi) \right|$$

and where $\phi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ is the heading measured with respect to the line segment $\mathcal{G}_\infty$.

Proof. Consider a circle of minimum turning radius $r_{\min}$ that is tangent to $\boldsymbol{\eta}$. The position $\boldsymbol{c} = [x_c\ y_c]^\top$ is denoted as the center of this circle. Without loss of generality, we consider a left turn at an angle of $\phi \in [0, \frac{\pi}{2}]$ as in Figure 4.4 (a similar analysis can be made for right turns and $\phi \in [-\frac{\pi}{2}, 0]$ — heading angles that point 'down' with respect to $\mathcal{G}_\infty$).

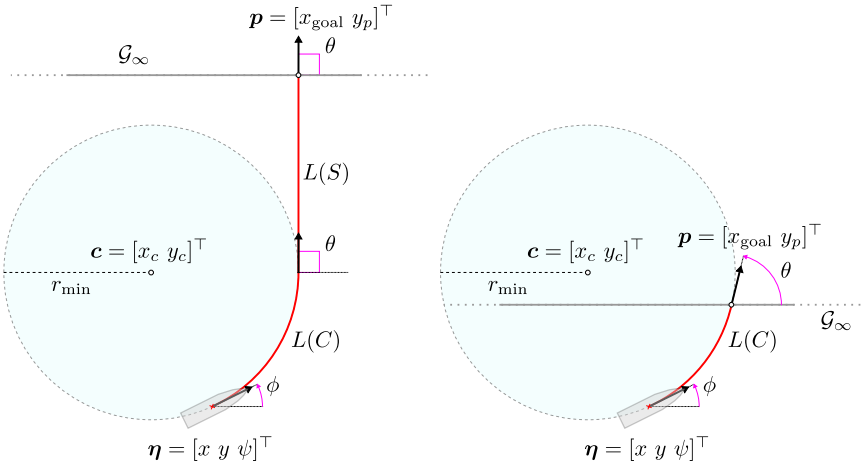


Figure 4.4: Diagram depicting the geometry of the Dubins path to an infinite line $\mathcal{G}_\infty$ for two possible cases.

Case 1: Suppose that $\mathcal{G}_\infty$ lies above $\boldsymbol{c} = [x_c\ y_c]^\top$. This is equivalent to the condition that x_c is less than x_{goal}, i.e.

$$x_c = x + r_{\min}\cos(\phi) \leq x_{\text{goal}}. \tag{4.17}$$

Let $\boldsymbol{p} = [x_{\text{goal}}\ y_p]^\top$ be any point on $\mathcal{G}_\infty$ and is outside of the circle of radius $r_{\min}$ centered at $\boldsymbol{c}$ (it is trivial to show the point $\boldsymbol{p}$ cannot be inside this circle for the shortest path from $\boldsymbol{\eta}$ to $\mathcal{G}_\infty$). Since the heading of $\boldsymbol{p}$ is not specified, the shortest path from $\boldsymbol{\eta} = [x\ y\ \psi]^\top$ to $\boldsymbol{p} = [x_{\text{goal}}\ y_p]^\top$ is of the form CS [17, 62] where the straight line segment S intersects $\mathcal{G}_\infty$ at an angle θ. Therefore, determining the shortest path from $\boldsymbol{\eta}$ to $\mathcal{G}_\infty$ reduces to computing

$$\min_{\theta} \quad L(C) + L(S), \tag{4.18}$$

where $L(\cdot)$ denotes length. We observe

$$\begin{aligned} L(C) &= r_{\min}(\phi - \theta), \\ L(S) &= \frac{x_{\text{goal}} - [x + r_{\min}(\cos(\phi) - \cos(\theta))]}{\sin(\theta)}. \end{aligned} \tag{4.19}$$

Further, we observe that the total path length $L(C) + L(S)$ is minimized by $\theta = \pi/2$. Replacing this value in $h_a(\boldsymbol{\eta}) = L(C) + L(S)$ yields the result of the Theorem for the case $x_c \leq x_{\text{goal}}$.

Case 2: Suppose instead that $\mathcal{G}_\infty$ lies below $\boldsymbol{c}$. In this case, the shortest path from $\boldsymbol{\eta}$ to $\mathcal{G}_\infty$ consists only of a circular arc C which intersects $\mathcal{G}_\infty$ and has length equal to the result of the Theorem for the case that $x_c > x_{\text{goal}}$. □

In our particular scenario, where we have channel boundaries perpendicular to $\mathcal{G}^{\text{int}}$, the function $h_1(\boldsymbol{\eta})$ can easily be extended to account for additional cases including the case where we can set $h_1(\boldsymbol{\eta}) = \infty$ indicating no path exists from $\boldsymbol{\eta}$ to $\mathcal{G}^{\text{int}}$. This is achieved by comparing the positions of both $\boldsymbol{c}$ and $\boldsymbol{p}$ with respect to the channel boundaries.

4.4.2 Obstacles-only

The following is a simple admissible heuristic for (4.3), i.e., a lower bound for the cost of the swath $\mathcal{S}(\boldsymbol{\pi})$ given a pose $\boldsymbol{\eta}$ and the current goal $\mathcal{G}^{\text{int}}$. Let $w \in \mathbb{N}$ be the width of the ship footprint in costmap grid units. Now suppose that we have a path $\boldsymbol{\pi}$, starting from $\boldsymbol{\pi}(0) = \boldsymbol{\eta}$ and ending at $\boldsymbol{\pi}(L_f) \in \mathcal{G}^{\text{int}}$, and a corresponding swath $\mathcal{S}(\boldsymbol{\pi})$ that sweeps rows i through j in the costmap for a total of $N = j - i + 1$ rows. It must hold that the part of the swath that covers a particular row must consist of a set of grid cells $\mathcal{S}' \subset \mathcal{S}(\boldsymbol{\pi})$ with at least w consecutive elements, for all N rows. In other words, we have $|\mathcal{S}'| \geq w$ in the general case and $|\mathcal{S}'| = w$ when the ship heading ψ is aligned with the length of the channel / costmap. A lower bound for the cost of the single-row swath $\mathcal{S}'$ for any particular row k in the costmap is therefore achieved by selecting the lowest sum of w consecutive grid cells in the k-th row. If we extend this to all N rows in the costmap that make up the swath $\mathcal{S}(\boldsymbol{\pi})$, then we get a lower bound for the cost of the swath $\mathcal{S}(\boldsymbol{\pi})$ for all paths $\boldsymbol{\pi}$ from $\boldsymbol{\eta}$ to $\mathcal{G}^{\text{int}}$ and for all $\boldsymbol{\eta}$. This gives us an admissible heuristic function $h_2(\boldsymbol{\eta})$ for the total collision cost C_f in (4.3). Although this heuristic relies on a loose lower bound on the cost, it is efficient to compute (linear search for each row), and we are guaranteed admissibility in contrast to the standard holonomic with obstacles heuristic for planning on costmaps in [27, 10].

Chapter 5

Optimization-Based Path Improvement Step

In the previous chapter, we described our proposed lattice-based planning method to efficiently compute low-cost smooth reference paths for a tracking controller. While the planner computes optimal solutions to the discretized planning problem defined by the graph $G^{\mathcal{P}}$, planned paths often contain excessive oscillations [16, 28], and the solution quality can vary with how the lattice is initialized given the current pose of the ship $\boldsymbol{\eta}_{\text{cur}}$. More broadly, the solutions are suboptimal for the following continuous path planning problem:

$$
\begin{aligned}
\underset{\boldsymbol{\pi},\,\kappa,\,L_f}{\text{minimize}} \quad & J = L_f + \alpha C_f && && \text{(5.1a)}\\
\text{subject to} \quad & \dot{x}(s) = \cos(\psi) && s \in [0, L_f], && \text{(5.1b)}\\
& \dot{y}(s) = \sin(\psi) && s \in [0, L_f], && \text{(5.1c)}\\
& \dot{\psi}(s) = \kappa(s) && s \in [0, L_f], && \text{(5.1d)}\\
& -r_{\min}^{-1} \leq \kappa(s) \leq r_{\min}^{-1} && s \in [0, L_f], && \text{(5.1e)}\\
& \mathcal{F}(\boldsymbol{\pi}(s)) \subset \mathcal{I} && s \in [0, L_f], && \text{(5.1f)}\\
& \boldsymbol{\pi}(0) = \boldsymbol{\eta}_{\text{cur}}, && && \text{(5.1g)}\\
& \boldsymbol{\pi}(L_f) \in \mathcal{G}^{\text{int}}, && && \text{(5.1h)}
\end{aligned}
$$

where the constraints (5.1b)-(5.1e) are from the unicycle model (4.1) and the constraint (5.1f) ensures the ship body (characterized by the footprint $\mathcal{F}$) remains within the ice

channel. In this version of the problem, the total collision cost C_f extends the function (4.3) to its continuous form, where we have an arc length parameterized line integral similar to the abstract expression given in (3.6). Although the precise form of the new function C_f remains unclear, we will show here how we can define a suitable function for the total collision cost that maintains the notions of a footprint and a swath using ideas from [97].

Similar to [27, 8, 10, 95, 64], we propose an optimization-based improvement step in a two-stage planning pipeline. In particular, we solve the path planning problem (5.1) in the continuous domain to locally refine the path computed by the lattice planner, which is used as an initial feasible solution to warm-start an optimization solver.

5.1 Scalar Cost Field

In order to leverage gradient-based methods to solve our optimization problem, we require a smooth representation of the constraints and objective. A challenge here is to transform the total collision cost C_f (using the notion of a swath) into a suitable differentiable function. In the first step, we define a smooth function $c_{\text{obs}}(x, y) \geq 0$ representing a scalar field for the collision cost from the grid-based costmap generated by the original discrete collision cost in (4.15). We use bicubic interpolation which considers patches of 4×4 grid cells from the costmap and solves a series of linear equations for the coefficients of an interpolated surface, while ensuring that the derivatives on the boundaries of the patches are consistent [38]. The use of cubic functions to fit the costmap data ensures that we have a smooth differentiable function for the collision cost everywhere in the environment. This includes the boundaries of the obstacles where there are clear discontinuities in the discrete costmap.

5.2 Body Points

With $c_{\text{obs}}(x, y)$ defined, we now need a way to calculate the total collision cost using the notion of a swath and a ship footprint. Let $\mathcal{B} \subset \mathbb{R}^2$ denote a set of *body points* where each body point $\boldsymbol{b} \in \mathcal{B}$ is a position $(x_{\mathcal{B}}, y_{\mathcal{B}})$ on the 2D ship body in the body-fixed coordinate frame (recall that the origin is at the center of gravity of the ship). In contrast to how $\mathcal{B}$ is defined in [97], we define the body points to cover both the exterior and the interior (as opposed to just the exterior) of the ship body. The importance of this design decision will be made clear after we have reformulated the total collision cost C_f. Using these body points, we can define the ship footprint $\mathcal{F}(\boldsymbol{\eta}) \subset \mathbb{R}^2$ given a pose $\boldsymbol{\eta}$ in the following way:

$$\mathcal{F}(\boldsymbol{\eta}) = \{\boldsymbol{g}(\boldsymbol{\eta}, \boldsymbol{b}) \mid \boldsymbol{b} \in \mathcal{B}\}. \tag{5.2}$$

The affine function $\boldsymbol{g} : \mathcal{C} \times \mathcal{B} \to \mathbb{R}^2$ maps a pose and a body point to the corresponding position in the world coordinate frame:

$$\boldsymbol{g}(\boldsymbol{\eta}, \boldsymbol{b}) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \boldsymbol{b} + \begin{bmatrix} x \\ y \end{bmatrix}. \tag{5.3}$$

Now consider the path of a particular body point $\boldsymbol{b}_j$ given a path $\boldsymbol{\pi}(s)$. We see that this is simply given by the arc length parameterized curve $\boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b}_j)$. Moreover, we can compute the cost that the body point $\boldsymbol{b}_j$ collects in the cost field by computing the line integral along the path $\boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b}_j)$. We can therefore define the total collision cost C_f as the cost of each of the body point paths integrated over all the body points:

$$C_f = \int_0^{L_f} \int_{\mathcal{B}} c_{\text{obs}}(\boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b}))) \left\| \frac{d}{ds} \boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b})) \right\| d\boldsymbol{b}\, ds. \tag{5.4}$$

The operator $||\cdot||$ denotes the Euclidean norm of the vector $d\boldsymbol{g}/ds$, i.e., the arc length derivative of $\boldsymbol{g}$, which we can immediately get by looking at (5.3) and recalling the unicycle model in the constraints (5.1b)-(5.1d):

$$\frac{d}{ds}\boldsymbol{g}(\boldsymbol{\eta}, \boldsymbol{b}) = \begin{bmatrix} -\kappa \sin(\psi) & -\kappa \cos(\psi) \\ \kappa \cos(\psi) & -\kappa \sin(\psi) \end{bmatrix} \boldsymbol{b} + \begin{bmatrix} \cos(\psi) \\ \sin(\psi) \end{bmatrix}. \tag{5.5}$$

Note that in (5.5), we have suppressed the arc length parameter s on the right-hand side of the equation; hence, κ and ψ should be interpreted as $\kappa(s)$ and $\psi(s)$, respectively. Multiplying the cost c_{obs} by $\|d\boldsymbol{g}/ds\|$ ensures that we are integrating along the particular path $\boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b})$ for each body point $\boldsymbol{b} \in \mathcal{B}$, and not the path $\boldsymbol{\pi}(s)$ unless $\boldsymbol{b} = [0\ 0]^\top$ (i.e., unless $\boldsymbol{b}$ is defined at the origin of the body-fixed frame where $\boldsymbol{\eta}$ is defined).

The total collision cost C_f in (5.4) shares similarities with the obstacle objective in the collision-free trajectory optimization approach from [97]. However, in addition to optimizing a collision-prone path rather than a collision-free trajectory, our cost function c_{obs} does not assign costs in the obstacle-free region $\mathcal{W}_{\text{free}}$ given $\mathcal{W}_{\text{obs}}$ in contrast to the obstacle objective in [97]. This means that we need to define the body points slightly differently, specifically both exterior and interior of the ship body, as mentioned earlier. To see why, consider the simple scenario illustrated in Figure 5.1. It consists of an ice floe $\mathcal{O}$ and two candidate straight paths $\boldsymbol{\pi}_a(s)$, $\boldsymbol{\pi}_b(s)$, both of which result in a collision with $\mathcal{O}$ where $\boldsymbol{\pi}_a$ results in a more head-on collision than $\boldsymbol{\pi}_b$. Now suppose that the set of body points $\mathcal{B}$ is defined along the exterior of the body as in [97]. In particular, consider the following two subsets of $\mathcal{B}$: $\{\boldsymbol{b}_1, \boldsymbol{b}_2\}$ located at the stern and bow, respectively, and $\{\boldsymbol{b}'_1, \boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_j\}$ positioned along one of the straight sides of the ship body. We want to

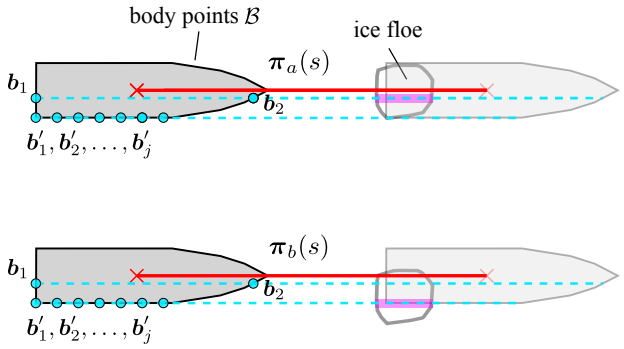


Figure 5.1: A simple scenario to illustrate the problem with defining body points exclusively on the outline of the ship body. Two candidate paths are considered, $\boldsymbol{\pi}_a(s)$ and $\boldsymbol{\pi}_b(s)$, in the top and bottom parts of the figure, respectively. We examine the cost of these two paths by considering the contribution to the cost from two subsets of $\mathcal{B}$: $\{\boldsymbol{b}_1, \boldsymbol{b}_2\}$ and $\{\boldsymbol{b}'_1, \boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_j\}$, depicted as cyan dots. The corresponding paths $\boldsymbol{g}(\boldsymbol{\pi}(s), \boldsymbol{b}_j)$ of the body points are shown with the cyan dotted lines. We see that in the bottom scenario, the magenta region of the cost field is disproportionately penalized from $\{\boldsymbol{b}'_1, \boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_j\}$ compared to the top scenario.

compare the total collision cost C_f from (5.4) for the two paths by examining how $\mathcal{B}$ affects this cost. Observe that for path $\boldsymbol{\pi}_b$ each body point in $\{\boldsymbol{b}'_1, \boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_j\}$ accumulates a cost from the region in the cost field highlighted in magenta. By comparison, consider how this cost field region affects the cost of $\boldsymbol{\pi}_a$. Here, there are exactly two body points, $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$, that accumulate cost in this region. This suggests that the cost computed from (5.4) will be higher for $\boldsymbol{\pi}_b$ compared to $\boldsymbol{\pi}_a$. However, if we recall the original swath cost formulation for C_f in (4.3), we know that $\boldsymbol{\pi}_a$ should be assigned a higher cost between these two candidate paths. As a result, from the example shown in Figure 5.1, defining body points exclusively on the outline of the ship body leads to cases where certain subsets of $\mathcal{B}$ disproportionately accumulate cost from our cost field. This example also illustrates that such a $\mathcal{B}$ would create an incentive to reduce the cost of a path by slightly adjusting it such that the interior region of the ship body now passes through a high-cost region that was part of the original path.

We emphasize the original goal of defining an appropriate function for the total collision cost C_f where there is a clear relationship with the original cost used to compute the lattice planner path. To address the problems discussed above with regard to body points, we present three possible options to define $\mathcal{B}$ and show an example for each in Figure 5.2. The first option is to define body points across the entire ship body (Figure 5.2 (top)),

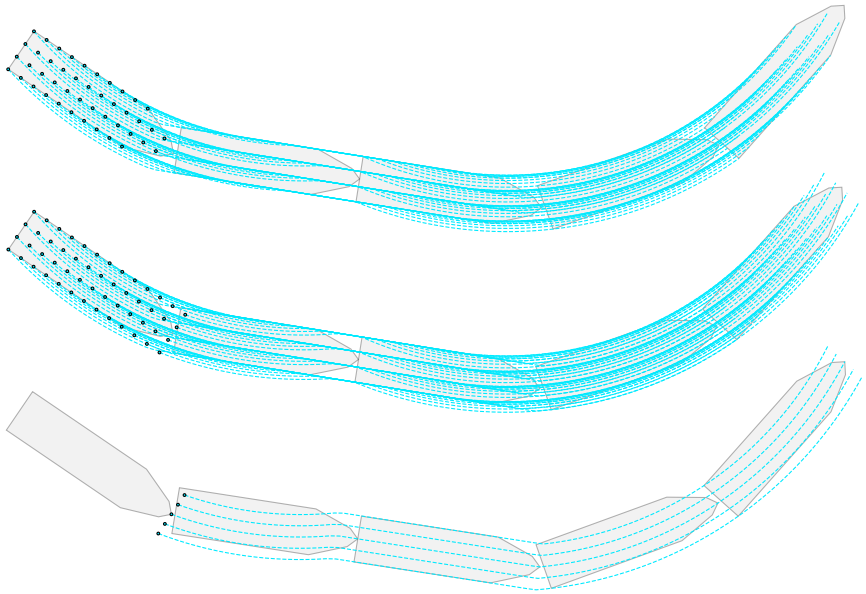


Figure 5.2: Three possible options to define the set $\mathcal{B}$. The dots represent body points sampled from the respective $\mathcal{B}$ and the cyan dotted lines show the body point paths.

resulting in a footprint computed as in (5.2), consistent with our previous definition of the ship footprint. However, now the points in the interior accumulate cost disproportionately, but can be addressed by multiplying each body point $\boldsymbol{b}_j \in \mathcal{B}$ by a weight $w_j > 0$. These weights also allow for fine-tuning of planning behavior (e.g., preference for collisions on the left side of the ship hull in the case that the right side has been slightly damaged). Alternatively, $\mathcal{B}$ is defined as the rectangle that covers the ship body (Figure 5.2 (middle)). The final option discussed here is to reduce $\mathcal{B}$ to a single straight line perpendicular to ψ, with a length equal to the width of the ship body and positioned at the tip of the bow (Figure 5.2 (bottom)). Here, we see that points in the cost field only get counted once since the body point paths are parallel to each other. Although having a line for $\mathcal{B}$ would also reduce computation, the resulting 'sweep' of the body points would miss some parts of the cost field particularly around sharp turns.

5.3 Transcription and Solver

The continuous planning problem in (5.1), incorporated with the collision cost C_f from (5.4), is solved numerically using tools from optimal control. In particular, we transform the

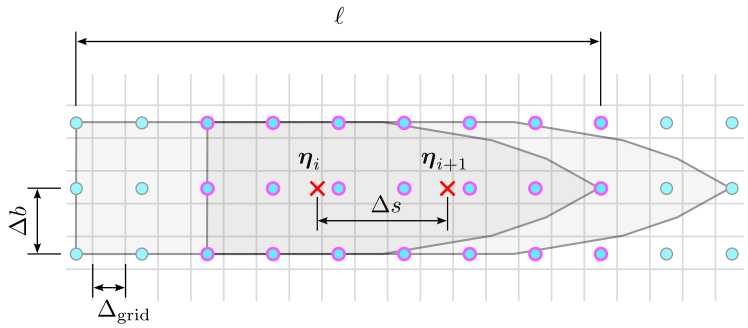


Figure 5.3: Illustration of the components involved to derive (5.6). The square grid shows the discrete costmap with resolution Δ_{grid}. Body points are shown for two consecutive poses, $\boldsymbol{\eta}_i, \boldsymbol{\eta}_{i+1}$, where the points outlined in magenta are overlapping. For the sake of visualization, everything here is nicely aligned with the costmap grid.

continuous problem into a discrete nonlinear program (NLP) using direct multiple shooting. The path is discretized into $N+1$ decision variables $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{N+1}$ and N decision variables $\kappa_1, \ldots, \kappa_N$ for the curvature κ (i.e. the signal that controls $\dot{\psi}$). Each control interval occurs over an arc length step of Δs, analogous to a time step in trajectory optimization, which sets the curvature between two sequential variables $\boldsymbol{\eta}_j, \boldsymbol{\eta}_{j+1}$ on the path. To avoid large changes in curvature over a small distance, we add a smoothness term to the objective function with a weight parameter $\lambda \geq 0$. Integration of the ODE for the unicycle model is performed using the 4th-order Runge-Kutta method (RK4).

We define a finite set $\mathcal{B}_d$ of discrete body points uniformly spaced on a square grid with spacing Δb, covering the rectangle that encloses the ship body (this is the second option discussed in the previous section and shown in Figure 5.2 (middle)). Each body point $\boldsymbol{b}_j \in \mathcal{B}_d$ has a corresponding weight $w_j > 0$. Let ℓ denote the length of the ship body and recall that Δ_{grid} is the resolution of the discrete costmap. By default, each weight w_j is assigned a constant value of

$$w_j = \frac{\alpha \Delta b^2}{\ell \Delta_{\text{grid}}^2}, \tag{5.6}$$

such that the terms in the objective—path length L_f (which is equal to $N\Delta s$ in the discrete problem) and total collision cost C_f—in the optimization problem have the same scaling as (4.4) in the lattice planning step. This expression is determined by calculating the average number of times a grid cell $\boldsymbol{k} \in \mathbb{N}$ in the original discrete costmap is covered by a body point $\boldsymbol{b}_j \in \mathcal{B}_d$ along a path $\{\boldsymbol{\eta}_i\}_{i=1}^{N+1}$ with an arc length step of Δs. Suppose we have a straight path (i.e. $\kappa_i = 0$ for $i = 1, \ldots, N$) where ℓ, Δs, and Δb are all multiples of Δ_{grid}

as shown in Figure 5.3. In this example, it is easy to see the two important ratios, namely $\ell/\Delta s$ and $\Delta_{\text{grid}}^2/\Delta b^2$ which control how often a grid cell is sampled on average.

The constraint (5.1f) in the original optimization problem is removed by assigning a sufficiently high cost in the cost field for points in $\mathbb{R}^2$ outside the boundaries of the ice channel $\mathcal{I}$. The following is the resulting discrete optimization problem:

$$\underset{\substack{\boldsymbol{\eta}_1,\ldots,\boldsymbol{\eta}_{N+1},\\ \kappa_1,\ldots,\kappa_N,\\ \Delta s}}{\text{minimize}} \quad \sum_{i=1}^{N+1} \sum_{\boldsymbol{b}_j \in \mathcal{B}_d} w_j c_{\text{obs}}(\boldsymbol{g}(\boldsymbol{\eta}_i, \boldsymbol{b}_j)) \left\| \frac{d}{ds} \boldsymbol{g}(\boldsymbol{\eta}_i, \boldsymbol{b}_j) \right\| \Delta s \tag{5.7a}$$

$$+ L_f + \lambda \sum_{i=1}^{N} \left(\frac{\kappa_{i+1} - \kappa_i}{\Delta s} \right)^2 \tag{5.7b}$$

$$\text{subject to} \quad \boldsymbol{\eta}_{i+1} = \boldsymbol{f}_{\text{RK4}}(\boldsymbol{\eta}_i, \kappa_i, \Delta s) \quad i = 1, \ldots, N, \tag{5.7c}$$

$$-r_{\min}^{-1} \leq \kappa_i \leq r_{\min}^{-1} \quad i = 1, \ldots, N, \tag{5.7d}$$

$$(x_i, y_i) \in \mathcal{I} \quad i = 1, \ldots, N, \tag{5.7e}$$

$$\boldsymbol{\eta}_1 = \boldsymbol{\eta}_{\text{cur}}, \tag{5.7f}$$

$$\boldsymbol{\eta}_{N+1} \in \mathcal{G}^{\text{int}}, \tag{5.7g}$$

This problem is solved using the NLP solver, IPOPT [86] (interfaced with CasADi [4] in our experiments), and warmed started with the initial plan computed by the lattice planner stage by resampling the path to $N+1$ intervals.

Chapter 6

Experiments

6.1 Simulation Experiments

We performed extensive simulation experiments to evaluate our framework's ability to autonomously navigate a vessel in ice-covered waters. The particular scenario considered is a platform supply vessel (PSV), a specially designed ship used to supply offshore installations, navigating through broken ice fields of various ice concentrations and containing variable-sized medium thick first-year ice floes. We considered a narrow channel of size 1000 m $\times$ 200 m (L$\times$W) with four possible ice concentrations: 20%, 30%, 40%, and 50% ice coverage. For each concentration, we generated 100 different ice fields where for each ice field we performed three trials: two baselines and the proposed method. At the start of each trial, the vessel was positioned 100 meters before the beginning of the ice field, aligned with the channel length, and centered with respect to the channel width (see Figure 6.2). The trials ended when the vessel reached the end of the ice field, a distance of 1100 m from the starting position. In total, we conducted 1200 trials, spanning 206 hours of simulation time and logging more than 36 million ship-ice collisions across 1320 kilometers of ship transit. A standard PC with an Intel Core i7-7700K processor with 32 GB of memory was used for these experiments. The total computation time to run all simulation tests and evaluation scripts was 60 hours (not including time to render animations). In the next sections, we describe our experimental setup in more detail, including our open-source physics simulator, followed by the simulation results.

6.1.1 Vessel model and control system

We used the full-scale PSV model from [32, 31] which has dimensions of 76.2 m × 18 m (L×W) and weighs 6000 metric tons. For propulsion, the vessel is equipped with two main propellers and four tunnel thrusters (two aft and two fore), for a total of six actuators. The main propellers and tunnel thrusters are symmetrically located about the longitudinal and transversal axes of the vessel, respectively. The actuator configuration is illustrated in Figure 6.1 and is characterized by the *actuator configuration matrix* $\mathbf{T} \in \mathbb{R}^{3\times 6}$. The dynamics of the vessel are described by a 3 DoF (i.e., surge u, sway v, and yaw rate r) linear state-space model intended for low-speed maneuvering on a horizontal plane. Hence, the model is of the form

$$\dot{\boldsymbol{\nu}} = \mathbf{A}\boldsymbol{\nu} + \mathbf{B}\boldsymbol{\tau}, \tag{6.1}$$

where recall $\boldsymbol{\nu} = [u\ v\ r]^\top \in \mathbb{R}^3$ is the generalized velocity vector while the generalized force vector is $\boldsymbol{\tau} = [X\ Y\ N]^\top \in \mathbb{R}^3$ for a force X in surge, force Y in sway, and torque N in yaw [69]. The model matrices $\mathbf{A}$ and $\mathbf{B} \in \mathbb{R}^{3\times 3}$ are taken from [32, 31]. In addition, we adapted the Dynamic Positioning (DP) based controller from [31] to track a moving setpoint that traces the reference path at a target speed computed by the generated speed profile with parameters $U_{\text{nom}} = 2.0$ m/s and constant acceleration 0.04 m/s^2. The nominal speed U_{nom} was set as constant to facilitate the analysis across trials with different ice concentrations. The DP controller uses pole placement for non-linear PD control, which computes the force vector $\boldsymbol{\tau}$ based on the error signals in surge, sway, and yaw. The propeller RPM commands for the six actuators are then calculated from $\boldsymbol{\tau}$ using the simple thrust allocation algorithm from [31]. It uses the generalized inverse to solve for the control input $\boldsymbol{u} \in \mathbb{R}^6$ in the equation

$$\boldsymbol{\tau} = \mathbf{T}\mathbf{K}\boldsymbol{u}. \tag{6.2}$$

The diagonal matrix $\mathbf{K} \in \mathbb{R}^{6\times 6}$ is referred to as the *thrust coefficient matrix* [30]. The main propellers are capable of producing a maximum of 799 kN of thrust each, while the tunnel thrusters each have a thrust limit of 200 kN. In the final step, we get the RPM commands $\{n_i\}_{i=1}^6$ where $n_i = \text{sgn}(u_i)\sqrt{|u_i|}$.

Note that the vessel model from [32, 31] lacks details on the shape and geometry of the vessel. As a result, we used the bow geometry for an ice-going supply vessel from [24] to define our ship footprint as shown in Figure 6.1.

6.1.2 Physics simulation

In this section we present our physics simulator used for the simulation experiments. Although there exist numerous works on numerical simulation of ship interactions with broken

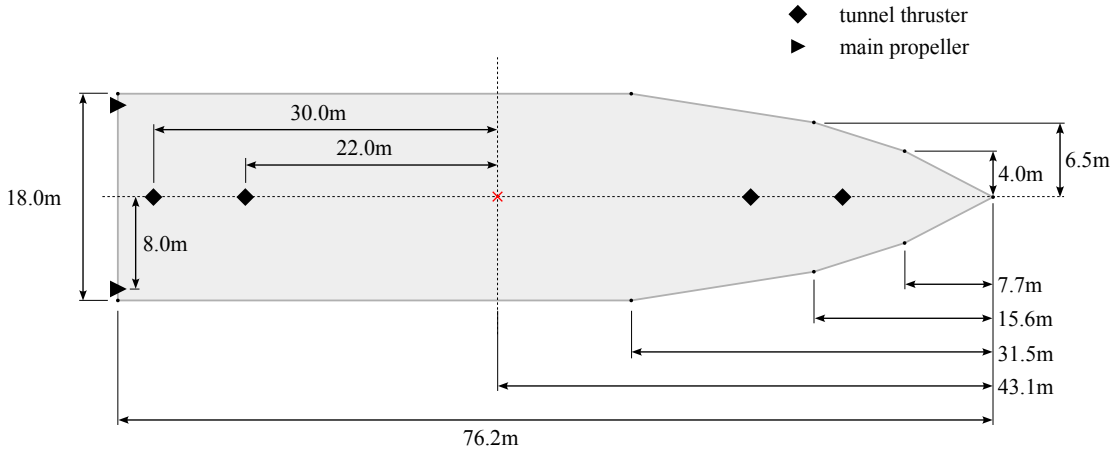


Figure 6.1: The dimensions, bow geometry, and actuator configuration of the vessel considered in the simulation experiments. The origin of the vessel is depicted by the red 'x'.

ice, e.g. [88, 45, 25, 24, 80, 60, 50, 51, 3, 57, 42, 61, 70], none provide open source code and few are sufficiently efficient to perform extensive experiments. We therefore developed our own simulation solution and provide all source code as well as accompanying documentation.

Our simulator is built on top of the Python library Pymunk [11], which is a wrapper for the 2D real-time rigid body impulse-based physics engine Chipmunk2D [56]. Collisions between bodies, i.e. ice-ice and ship-ice collisions, are therefore treated as rigid body contacts and inelastic. This means that we ignore more complex scenarios, such as ice crushing and ice splitting. Similar approximations have been made in other models, including [88, 25, 24, 45]. We used existing simulation work from [88, 45, 25, 24, 89, 80, 60] and field data [79] to set the majority of the simulator parameters. This includes values for the ice properties and physics-related parameters such as *friction* and *restitution* coefficients for ice-ice and ship-ice contacts, which determine the collision response between bodies. We provide a summary of these parameters along with the respective sources in Table 6.1. Some of these parameters are described in more detail in Section 6.1.3 where we discuss our ice field generation process.

We model the hydrodynamic force acting on the ice floes as a quadratic drag force [51, 88],

$$\boldsymbol{F_d} = -\frac{1}{2}\rho_w C_d A_{proj} \boldsymbol{v} \|\boldsymbol{v}\|, \tag{6.3}$$

acting in the opposite direction of the ice floe linear velocity, $\boldsymbol{v} \in \mathbb{R}^2$. The values for

the water density ρ_w and the drag coefficient C_d are from [60] and [80], respectively. The factor A_{proj} is the projection of the submerged area of the ice floe along the direction of $\boldsymbol{v}$. However, since our simulation is in 2D, we assume that the ratio of the submerged ice floe is equal to ρ_{ice}/ρ_w and the ice floes have a constant thickness of $h = 1.2$ m, which means that $A_{proj} \propto h\rho_{ice}/\rho_w$. Regarding the drag torque, we model the effect on the angular velocity as an exponential decay with a decay rate of 0.03.

We note that the drag forces experienced by the ship are captured by the system inertia and damping matrices embedded in the model of the vessel dynamics described in the previous section. This means that the physics engine only handles the kinematics of the ship and we therefore require the dynamics to account for the ice resistance forces. We do this by computing a force vector $\boldsymbol{\tau}_{\text{env}}$ updating the dynamics from (6.1) to

$$\dot{\boldsymbol{\nu}} = \mathbf{A}\boldsymbol{\nu} + \mathbf{B}(\boldsymbol{\tau} + \boldsymbol{\tau}_{\text{env}}). \tag{6.4}$$

To obtain $\boldsymbol{\tau}_{\text{env}}$, we aggregate all the impact forces and the corresponding contact points on the ship hull logged by the physics engine during the current control interval to get a net force and net torque. In practice, other environmental forces may be present in $\boldsymbol{\tau}_{\text{env}}$ such as wind and wave induced forces [30], but we ignore these additional disturbances here. To ensure we maintain reasonable behavior from the vessel dynamics, we set the following limit on $\boldsymbol{\tau}_{\text{env}}$ based on experimental data in [61]:

$$|\boldsymbol{\tau}_{\text{env}}| \leq \begin{bmatrix} 1500 \text{ (kN)} & 1500 \text{ (kN)} & 10^5 \text{ (kN} \cdot \text{m)} \end{bmatrix}^\top \tag{6.5}$$

Although we did not validate our simulator with real-world data, we expect a certain level of accuracy since the majority of our parameters and design decisions are based on existing works. Additionally, we observed some consistency in our logged simulation data with those reported in other studies [25, 24, 61]. We also note that our reported impact forces may be inaccurate or inflated (e.g., collisions with large ice floes) due to our assumption of rigid-body physics and our use of an impulse-based collision solver which sacrifices some physical accuracy for efficient collision handling [65].

6.1.3 Ice field generation

Our ice field generation method produces random ice fields given a specified ice concentration where the ice floe statistics are consistent with the literature. Figure 6.2 shows an example of a generated ice field for each of the four ice concentrations considered for

Parameter	Value	Reference
Physics engine time step Δt_{sim} (s)	0.005	-
Controller sampling time Δt_{ctrl} (s)	0.02	[31]
Channel length (m)	1000	-
Channel width (m)	200	[24]
Ship mass (kg)	$6000 \cdot 10^3$	[31]
Ship dimensions (L×W) (m)	76.2×18	[32]
Ship target speed (m/s)	2	[31]
Ship-ice friction coefficient	0.05	[45]
Ice-ice friction coefficient	0.35	[45]
Ship-ice restitution coefficient	0.1	[89]
Ice-ice restitution coefficient	0.1	[89]
Form drag coefficient C_d	1.0	[80]
Angular velocity decay	0.03	-
Water density ρ_w (kg/m^3)	1025	[60]
Ice density ρ_{ice} (kg/m^3)	900	[60]
Ice thickness h (m)	1.2	[24]
Ice size range (m^2)	16–10,000	[79]
Ice side count range	5–20	[25]

Table 6.1: Physics and other parameters used for our simulator.

our experiments. Following [24, 44], we modelled the ice mass as a random variable that follows a log-normal distribution with probability density function $f_Y(y)$ given by

$$\begin{aligned} f_Y(y) &= \frac{1}{b} f_X\left(\frac{y-a}{b}\right) \\ f_X(x) &= \frac{1}{x\sigma\sqrt{2\pi}} \exp\left[-\frac{(\ln x)^2}{2\sigma^2}\right]. \end{aligned} \tag{6.6}$$

The location-scale parameters, with values $a = 10.21$ and $b = 0.9324$, are from [24] and we set $\sigma = 0.54$. Using the set of sampled mass values $\{m_i\}_{i=1}^N$ (in kg), we calculated the corresponding floe area A_i given the ice density ρ_{ice} and the thickness h, clamping the area between 16 m^2 and 10,000 m^2. We therefore restricted our attention to ice floes with *effective width* (i.e. the square root of area) between 4 m and 100 m, a reasonable range for small ice floes [79]. To generate a random arrangement of ice floes, we first applied the circle packing algorithm described in [87] with inputs $\{r_i\}_{i=1}^N$ as the set of radii calculated from the set of floe sizes $\{A_i\}_{i=1}^N$, i.e., $r_i = \sqrt{A_i/\pi}$. The arrangement of nonoverlapping

circles was generated such that it covered the whole ice field. From here, the centers and radii of the circles were used to initialize the algorithm from [84] to generate random convex polygons consisting of 5 to 20 sides [25]. We used an occupancy grid of the ice field to check for overlap between polygons. In the final step, we randomly removed polygons from the ice field until we reached our target ice concentration.

We generated 100 different ice fields for each of the four ice concentrations considered, for a total of 400. The generated ice floes have a mean effective width of 8.39 m and a standard deviation of 4.68 m. For the floe size, the mean is 92.29 m^2 with a standard deviation of 231.34 m^2. The number of floes per ice field ranges from 300 for low-concentration ice fields to 1300 for high-concentration ones. The resulting distribution of floe sizes shows similarities to field data collected in [88, 72, 79], where the distribution is modeled as a power law.

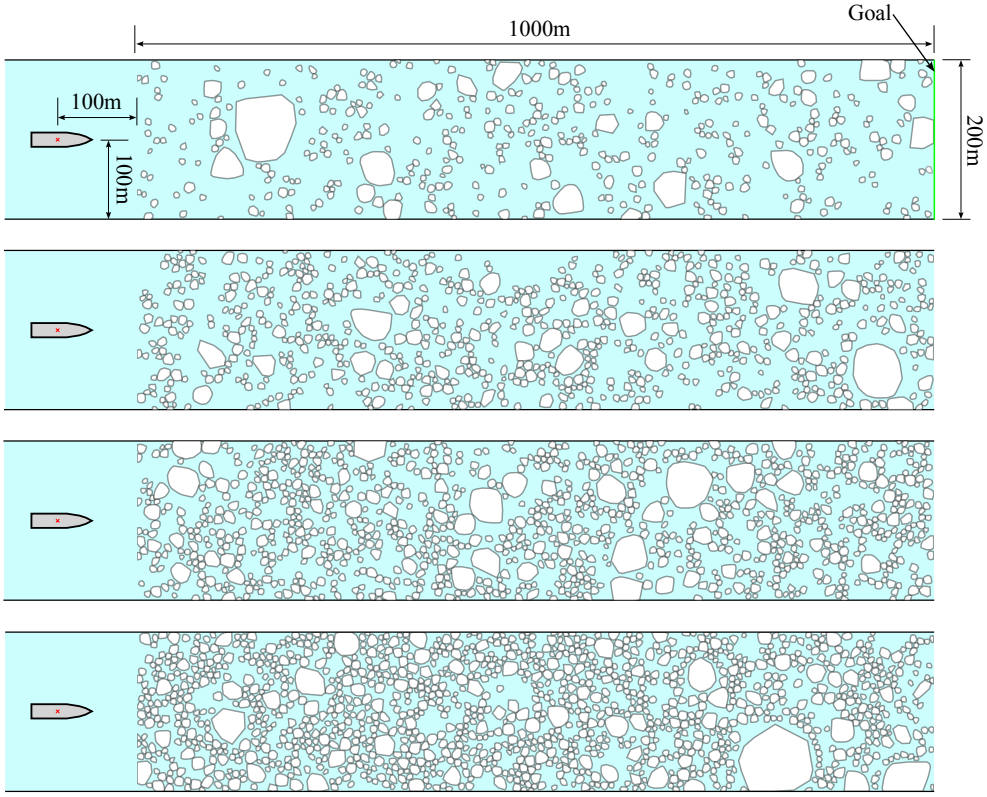


Figure 6.2: Sample ice fields generated for the simulation experiments. Four different ice field concentrations are shown: 20%, 30%, 40%, and 50% ice coverage (top to bottom).

6.1.4 Metrics

We report several key metrics to evaluate navigation performance. The first set of metrics relate to the logged ship-ice collisions. The following information was stored for each of these collisions: ice mass, collision impulse, kinetic energy loss in the system ΔK_{sys}, change in kinetic energy of the ice ΔK_{ice}, and contact points on the ship hull. Using this collision data, we computed the mean collided ice mass, maximum magnitude of the impact force, mean magnitude of the impact force, and the total kinetic energy loss ΔK_{ship} of the ship from collisions (using (4.9)), for each trial. We approximated the impact force as the collision impulse divided by the simulation time step Δt_{sim} [41].

The other two metrics of interest are the total energy use and the total transit time. Given the simulated velocities $\{\boldsymbol{\nu}_k\}_{k=1}^N$ and forces $\{\boldsymbol{\tau}_k\}_{k=1}^N$ for control steps $1, ..., N$, we computed the total energy, E, as

$$E = \Delta t_{\text{ctrl}} \sum_{k=1}^{N} |\boldsymbol{\nu}_k|^\top \cdot |\boldsymbol{\tau}_k|, \tag{6.7}$$

where $|\cdot|$ denotes the element-wise absolute value operator. These additional metrics offer insight into the trade-off between efficiently traversing the ice field and altering course to avoid significant impacts with ice.

6.1.5 Baselines

We considered two baselines, which we refer to as *straight* and *skeleton*. The former plans a straight path from the initial position of the ship to the goal (i.e., the end of the channel), while the latter refers to the shortest open-water path routing approach described in [34, 67]. Given a binary occupancy grid of the ice channel, the skeleton planner starts by generating a morphological skeleton image [94] of the open water area. The resulting image consists of a series of 1 pixel wide lines which tend to arrange in a network-like structure. This provides a useful representation of the environment's topology from which a graph can be constructed and then searched, producing a path. The path is then post-processed with a smoothing operation, which accounts for the minimum turning radius r_{min} of the ship and the current heading ψ. We added additional logic that progressively erodes the binary occupancy grid until an open-water path is found since no path may exist in the original image for high concentration ice fields. Both baselines are integrated with our navigation framework from Algorithm 1 where the different path planning methods generate a path on Line 8. The parameters for the tracking time Δt and the receding horizon parameter

Δh were set to 30 seconds and 500 meters, respectively, and are the same for the three planners. At 2 m/s, the ship travels a distance of 60 meters, or nearly one ship length, between planning iterations. Note that although the speed profile is recomputed during planning updates, in the straight experiments the path is only planned once at the start of the trial.

6.1.6 Lattice planner parameters

The following outlines the parameter values used in our lattice planning framework for the full-scale simulation experiments. We set the resolution of the costmap to 2 m $\times$ 2 m meaning an ice floe covered between 4 and 2500 grid cells while the ship footprint consisted of 290 grid cells. For our environment with dimensions of 1100 m $\times$ 200 m, this resulted in a costmap size of 551 $\times$ 101. Figure 4.2 illustrates the grid cell size relative to the ship footprint. The configuration space, on the other hand, was discretized to 30 m $\times$ 30 m (roughly half a ship length) for planar position, and the heading angle was discretized into 8 uniform intervals. The constraint on yaw rate was set based on recommendations from a ship captain associated with the NRC, who suggested 45 deg/min as an acceptable limit for low-speed maneuvering. At a nominal speed U_{nom} of 2 m/s, we get a minimum turning radius r_{min} of 150 m. The discretization of the configuration space and the minimum turning radius were used as inputs to the algorithm from [14] to generate our motion primitives. We have 45 motion primitives for the even-numbered headings ($n\pi/4$, where n is even) and 43 motion primitives for the odd-numbered headings ($n\pi/4$, where n is odd), as shown in Figure 6.3. Our goal was to generate a set of motion primitives suitable for a range of different channel dimensions. Note that while some of the primitives may not always 'fit' in the particular environment used in these experiments, they are quickly discarded as possible edges during node expansion in the graph search.

Recall that we use the path length to represent the energy used in transiting open water and the total collision cost to account for the additional energy required to collide with and push ice floes. The weighting between these two terms in (4.4) is specified by the parameter α. Calibration of this weight was done by comparing the planner cost for a straight path to the actual simulation cost in the straight trials. Specifically, our goal was to match the ratio of the total collision cost to the total cost in (4.4) with the ratio of the kinetic energy loss ΔK_{ship} to the total energy use E:

$$\frac{\alpha C_f}{L_f + \alpha C_f} = \frac{\Delta K_{\text{ship}}}{E}. \tag{6.8}$$

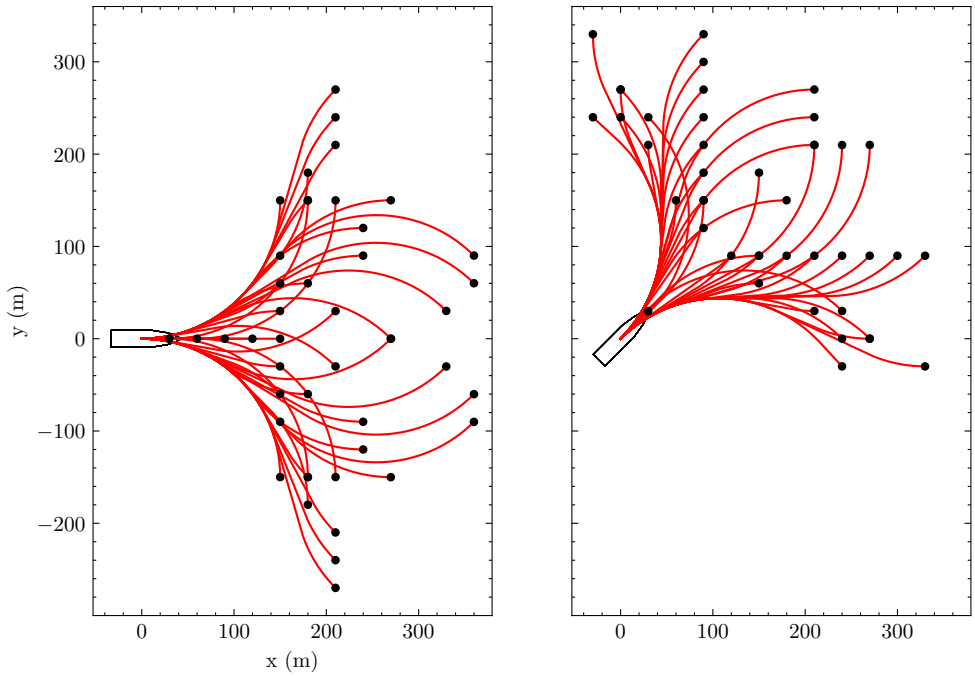


Figure 6.3: Primitives generated for the full-scale simulation experiments. There are 45 and 43 primitives for the 45 for axis aligned and 43 for non-axis aligned

Solving for α gives us the expression,

$$\alpha = \frac{\Delta K_{\text{ship}}}{E} \frac{L}{C_f \left(1 - \frac{\Delta K_{\text{ship}}}{E}\right)}. \tag{6.9}$$

We used (6.9) to compute the weights $\{\alpha_i\}_{i=1}^{100}$, with each α_i corresponding to a sampled straight trial, and then took the mean to obtain the weight used in the actual lattice trials: $\alpha = 4.8 \times 10^{-7}$. Regarding the parameters for the ice concentration term in the cost function, we set $\beta = 1$ and used a 51×51 kernel for the image convolution operation. For the path improvement step from Section 5, the set $\mathcal{B}_d$ consisted of 52 body points equally spaced along 4 rows, similar to Figure 5.2 (middle). The weight λ for the smoothness term in (5.7) was set to 5×10^4 and the lattice planner solution was downsampled to an initial arc length step Δs of 4 meters (so, roughly 125 control intervals). This concludes our description of the experimental setup. In the next section, we present the results from our simulation experiments.

Concentration	Planner	Mean collided ice mass (10^3 kg)	Max impact force (10^3 kN)	Mean impact force (kN)	Ship KE loss ΔK_{ship} (10^3 kJ)	Total Energy E (10^3 kJ)	Total time (s)
20%	Straight	360	1025	576	100	245	**604**
	Skeleton	224	461	252	37	233	620
	Lattice	**99**	**134**	**133**	**22**	**218**	615
30%	Straight	196	960	383	125	273	**604**
	Skeleton	133	447	205	82	259	618
	Lattice	**82**	**147**	**140**	**41**	**236**	614
40%	Straight	168	936	394	147	316	**613**
	Skeleton	107	392	200	102	296	620
	Lattice	**78**	**156**	**165**	**63**	**281**	618
50%	Straight	137	786	362	167	**335**	**609**
	Skeleton	99	360	235	128	363	638
	Lattice	**74**	**160**	**210**	**93**	**335**	621

Table 6.2: Results from the simulation experiments for all 1200 trials. Scores are averaged across 100 trials for a particular ice concentration and planner.

6.1.7 Results

A total of 1200 trials were conducted: 4 concentrations $\times$ 3 planners $\times$ 100 ice fields, spanning 206 hours of simulation time. We present a summary of the results in Table 6.2. The scores for each metric in the table are averaged over the 100 trials performed for a particular ice concentration and planning method. With our method (lattice), the ship collided with smaller ice pieces on average and experienced fewer head-on collisions, resulting in a significant decrease in the mean and maximum impact forces generated by the ship-ice collisions. Compared to the straight and skeleton tests, we achieved a decrease of 62% and 27% in the mean impact force and a decrease of 84% and 64% in the maximum impact force. These improvements are best visualized in Figure 6.4, where we plot the force vectors at their respective contact points for all 36.6 million collisions recorded between the three methods.

In addition to the reductions in impact force, which decrease the potential for damage to the vessel, our experiments also demonstrated improvements in energy use with our method. The smaller impact forces resulted in a decrease in the total kinetic energy loss of the ship from colliding with ice floes. If we sum ΔK_{ship}, our method resulted in the ship losing 59% less kinetic energy compared to the straight trials. This efficiency gain more than offset the additional energy required to follow the longer curved paths generated by our proposed planner. In particular, our method achieved a 8.5% decrease in total energy use E compared to the straight trials. This suggests that the weighted sum of the path length and the swath cost is effective in capturing the true cost of transiting an ice field. Moreover, this means that the weight α was properly calibrated for these experiments. Recall also the guidelines established by the Canadian Coast Guard for ice

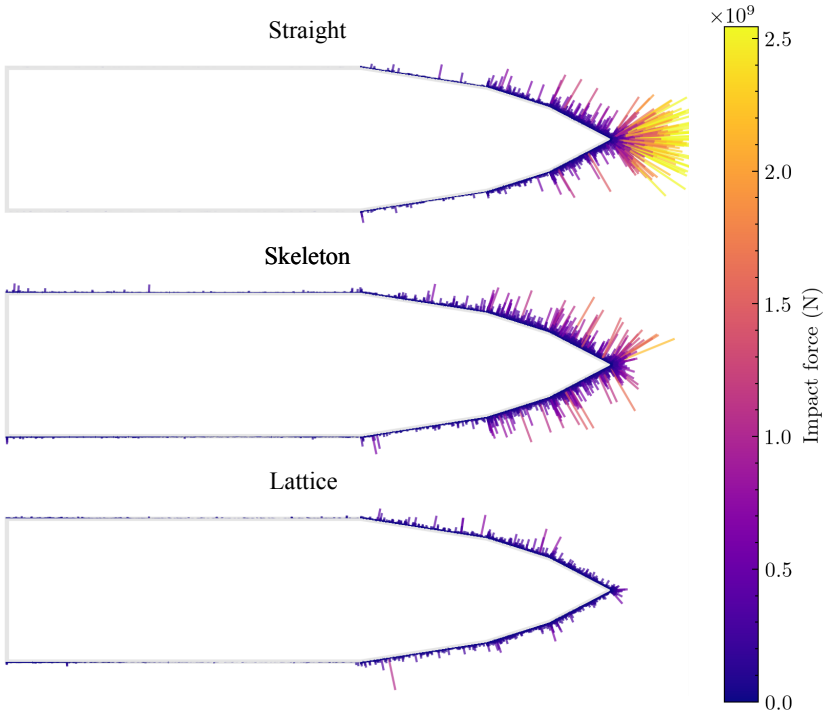


Figure 6.4: Visualization of all impact forces and the corresponding contact-points from the ship-ice collisions that occurred during the simulation trials. Each vector represents the magnitude and direction of forces experienced by the ship's hull (grey outline). In total, the straight trials, skeleton trials, and lattice trials logged 11.3 million, 13.0 million, and 12.3 million collisions, respectively.

navigation advise vessels to extend their journey as necessary to reduce the risk of damage, potentially leading to voyages with less fuel consumption [35]. The possibility of achieving both of these objectives was clearly demonstrated in our simulation results. Of course, the additional cost associated with longer travel time should also be factored in, however, this cost is likely negligible compared to the possible cost incurred from damaging the ship. In our experiments, across 400 trials or 67.5 hours of ship transit, the total travel time was extended by only 1.5 hours with our method compared to going straight. It is also worth noting that the controller performed adequately in tracking the reference path at the target speed. The average cross-track error was 2.0 meters, while the average heading error was 1 degree.

In addition to testing our method against the straight and skeleton planner baselines,

we evaluated the effectiveness of our proposed optimization-based improvement step. We conducted the same 400 trials described above for our method *without* the second stage. From these additional trials, we see that the improvement step provides a clear benefit in further improving navigation. On average, the second stage provided a decrease in the maximum and mean impact force of 19% and 10%, respectively, while using 1% less in total energy use compared to the baseline lattice trials without the improvement step. Figure 6.5 shows the results for the impact force metrics, and Figure 6.6 shows four representative snapshots of a sample trial with our proposed method (i.e., with the improvement step) in a 40% concentration ice field.

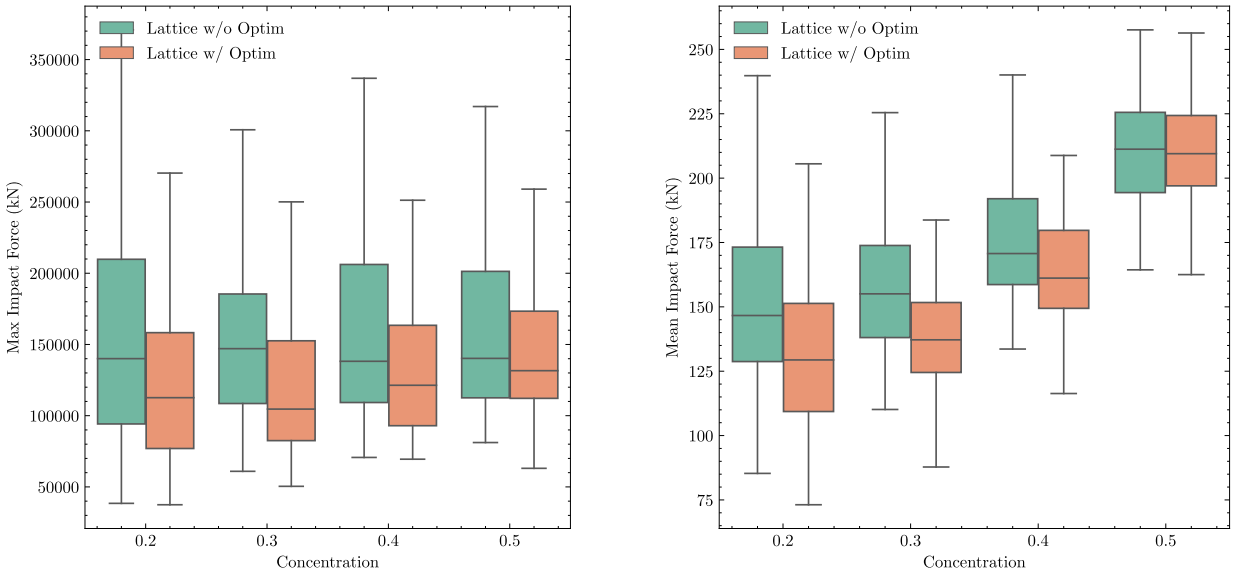


Figure 6.5: Results from evaluating our proposed improvement step. We show the results for the mean and maximum impact force metrics in the two methods.

We conclude this section with a discussion on the limitations of these experiments and provide ways in which to improve our simulation setup. One notable shortcoming is the lack of testing our approach under wind- and wave-induced forces. Such experiments would require extending both the vessel equations of motion in (6.4) [30] and the forces acting on the ice floes including the drag force in (6.3), handled by the physics engine. These additional environmental disturbances would produce a more dynamic environment and thus provide new insights into the performance of our method. For example, such experiments may provide a clearer picture on the question of whether frequent replanning on updated static maps is a sufficient strategy to account for the dynamic ice while maintaining the desired behavior of avoiding costly collisions. Another interesting extension would be to

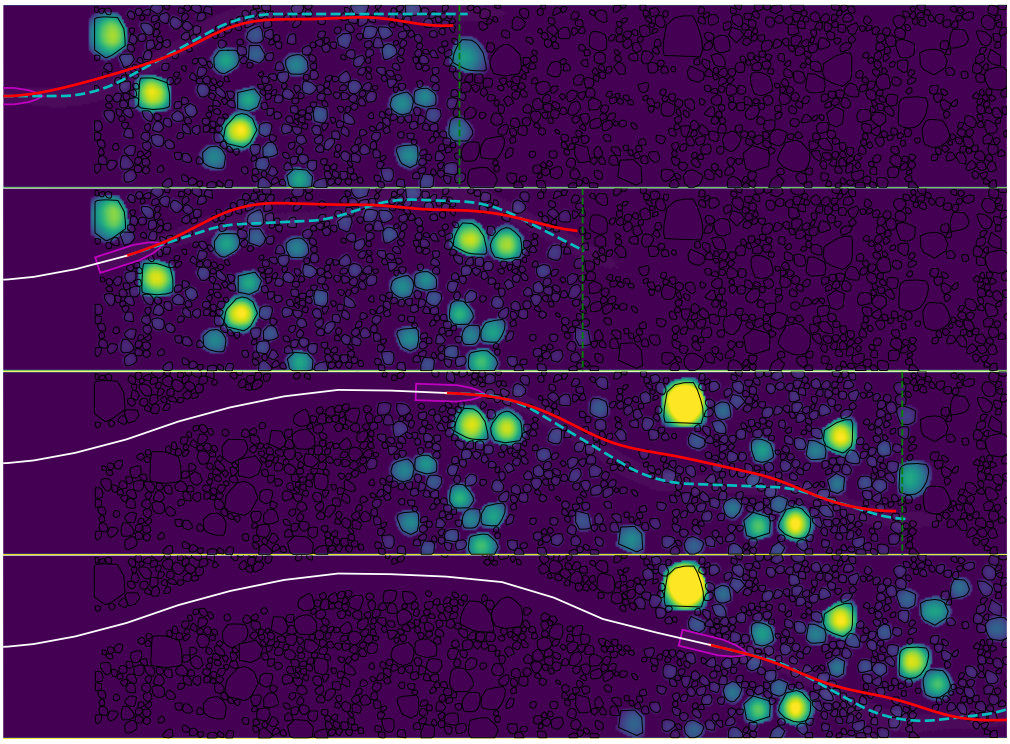


Figure 6.6: Four snapshots taking from a sample simulation trial in a 40% concentration ice field using our proposed method. Here, we show both the initial plan computed in the lattice planning stage (cyan dotted line) and the optimized path (red line). Visually, we see that the red path tends to be better in terms of both the path length and the ship-ice collisions that occur along it. The actual path of the ship is displayed in blue and the green dotted line shows the intermediate goal $\mathcal{G}^{\text{int}}$.

consider larger and wider ice environments. The particular dimensions chosen for these experiments may have limited the range of possible scenarios simulated due to the large turning radius relative to the channel width. Finally, a higher-fidelity model (i.e., nonlinear equations of motion) for the vessel dynamics may provide a better simulation of ship maneuverability in ice conditions.

6.2 Real-World Experiments

In addition to the experiments performed in simulation, we conducted real-world experiments at model scale in the offshore engineering basin (OEB) research facility in St. John's, Newfoundland and Labrador. The NRC-managed facility consists of a large basin and is used primarily to assess the performance of marine systems in controlled environments. We used a collection of plastic polygon pieces cut from polypropylene sheets (density 991 kg / m^3 and thickness 0.012 m) commonly used for artificial ice floes[1] [74, 88]. The size of the polygons ranged from 0.01 m^2 to 0.36 m^2 with a mean area of 0.094 m^2. In total, we had around 300 polygons which provided an ice concentration of 30%. The basin was configured with a 20 m $\times$ 6 m (L$\times$W) region confined with nets for our ice channel.

The ship model used in these experiments is the 1:45 scale platform supply vessel (PSV), designed and built by the NRC. The dimensions of the vessels are 1.84 m $\times$ 0.38 m $\times$ 0.43 m (L$\times$W$\times$H) and weighs about 70 kg. Sensors on the PSV include an IMU and feature two tunnel thrusters and two main propellers for propulsion. More details on the PSV are described in [67]. To control the vessel, we employed a Dynamic Positioning (DP) controller, regulating both position and heading given a moving setpoint along the planned path. The forward thrust was set constant and the nominal speed was set to 0.2 m/s. To capture ice information, we used a ceiling camera pointing down on the basin and used a similar image segmentation pipeline (updated at 30 Hz) described in [34]. In addition, the Qualisys Motion Capture system provided state information of the vessel at 50 Hz.

We performed a total of 60 trials and used the same baselines (i.e. skeleton planner and straight planner) from our simulation experiments to compare against our approach. Before starting each test, the PSV was manually controlled to be placed at a point along the starting line and its direction was aligned with the channel length. We considered three general starting positions along the starting line relative to the channel width: *center*, *right*, and *left*. For each navigation strategy, we performed 20 trials, initializing six of them with a center start, seven with a left start, and seven with a right start. The goal line was established at the opposite end of the channel, located 16 m from the starting point. Note that we required a buffer area between the goal and the end of the channel to avoid the ship from hitting and then getting stuck in the net. Trials were performed in both directions—'up' and 'down' the channel—by interchanging the start and goal regions at the completion of a particular test. Finally, we randomized the ordering of the 60 trials. The experimental setup is shown in Figure 6.7, and a side view is shown in Figure 6.8.

[1] The ice tank research facility shown in Figure 1.1 was under renovations at the time of this work.

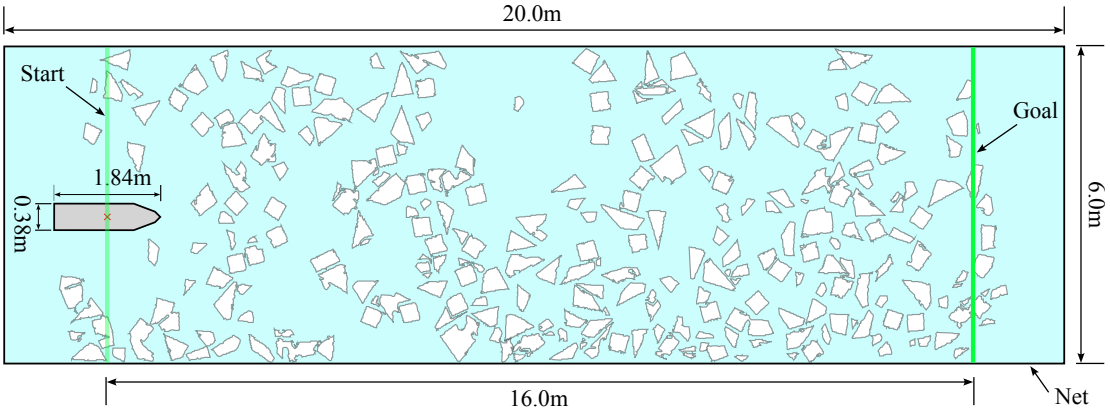


Figure 6.7: Diagram of the experimental setup for our real world experiments.

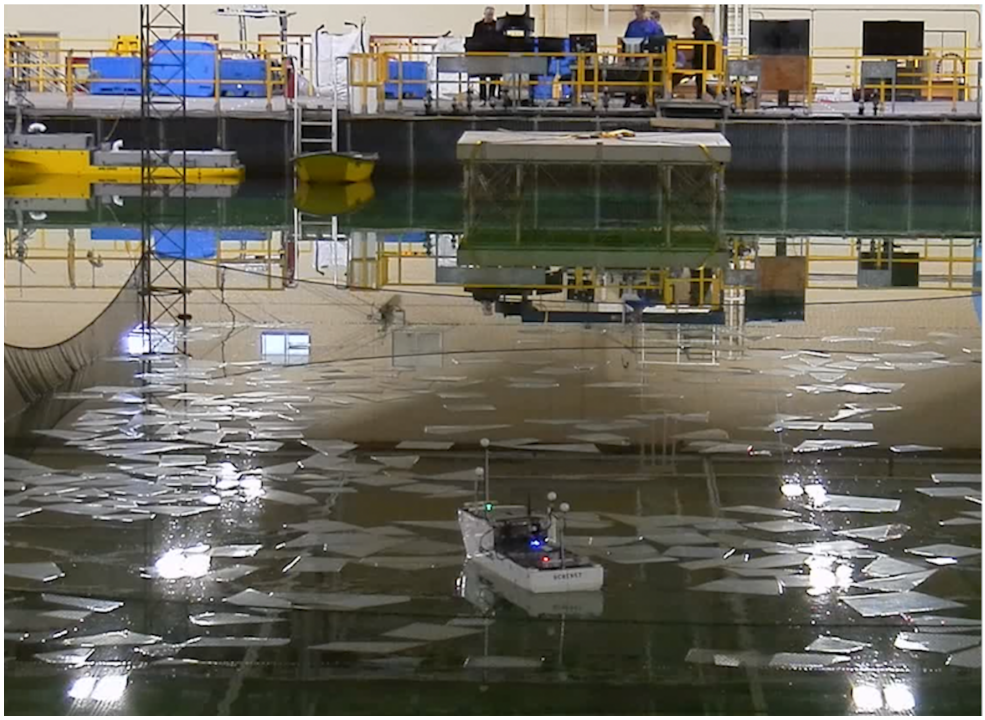


Figure 6.8: Side view of the setup in the offshore engineering basin (OEB).

6.2.1 Trial difficulty evaluation

A challenge encountered during the experiments was ensuring consistency between the three planning approaches with respect to the initial arrangement of ice floes. For example, a trial may start with the ice being distributed such that one side of the channel contains the majority of the ice, leaving the other side mostly open water. Although periodic 'resets' of the ice field were performed via manual control of the PSV to mitigate this effect, the starting ice field configuration remained a factor on trial performance. As a result, following the end of performing experiments, the researchers affiliated with this project were tasked with assigning a difficulty score for each of the trials. The evaluation process involved examining the first frame captured by the ceiling-mounted camera from a particular trial and responding to the following question:

On a scale from 1 to 5, where 1 is 'Very Easy' and 5 is 'Very Difficult,' how would you rate the overall difficulty of navigating the ship through the depicted ice field, considering the starting position of the ship and the characteristics of the ice field captured in the provided image?

The evaluations were carried out independently, with the trial sequence randomized, and planner information hidden. Overall, the evaluations were generally consistent with an average difference of 1.0 between the highest and lowest assigned difficulty scores in a trial. These scores yielded a mean difficulty for each trial and we binned the scores into three difficulty levels: *easy*, *medium*, and *hard*, with intervals $[1, 2.5]$, $(2.5, 3.5)$, and $[3.5, 5]$, respectively. Note that the intervals were chosen to maintain some consistency in the number of trials per planner for a particular difficulty level. A summary of the trials per difficulty level is given in Table 6.3, and we show three sample trials with their corresponding difficulty in Figure 6.9.

	Difficulty level and interval		
	Easy $[1, 2.5]$	Medium $(2.5, 3.5)$	Hard $[3.5, 5]$
Straight	8	5	7
Skeleton	3	8	9
Lattice	6	7	7

Table 6.3: Number of trials for each planner and difficulty level.

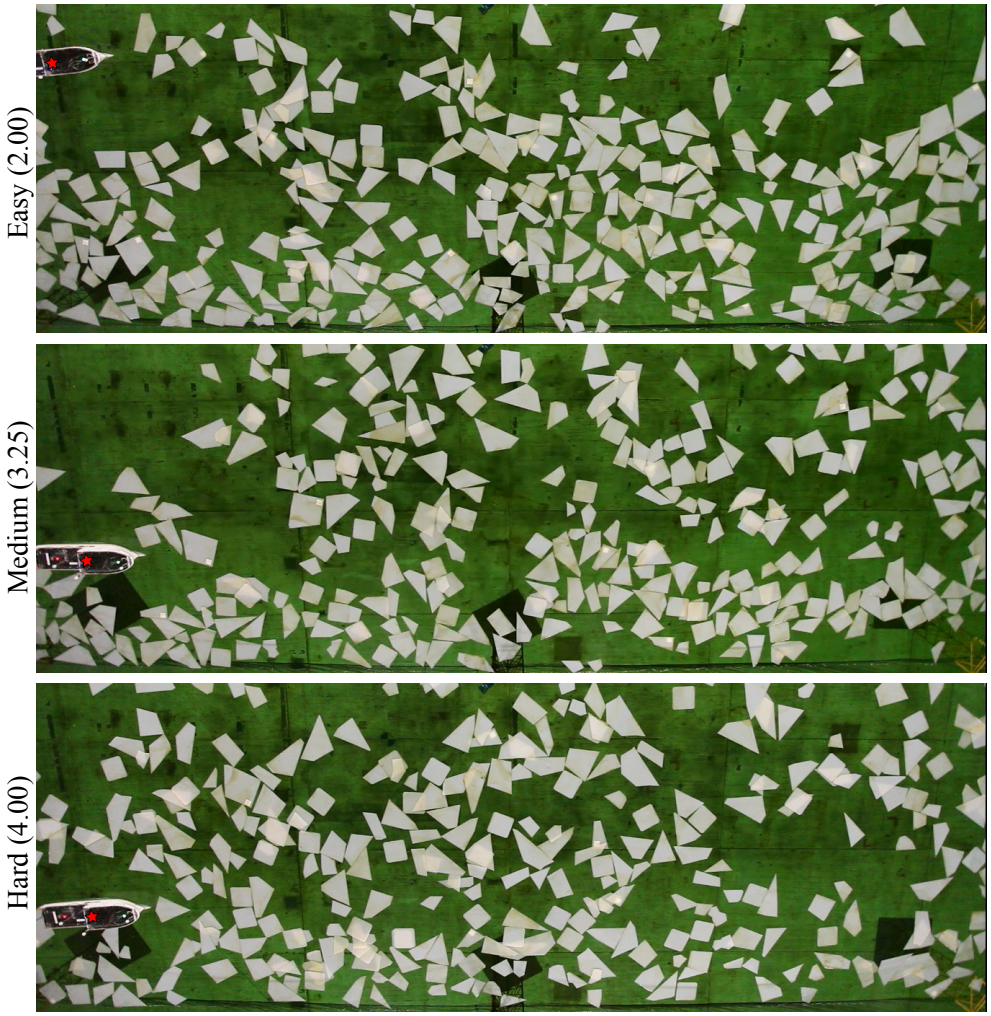


Figure 6.9: First frame from three sample trials with the corresponding difficulty level and mean difficulty score (in brackets) given the starting ship position (red star) and ice field configuration.

6.2.2 Metrics

We present a new set of metrics to evaluate the real-world experiments. The small-scale setup (e.g., lightweight ice and low ship speed) limited our ability to rely on some of the metrics discussed in the simulation section including impact forces from acceleration measurements logged by the IMU. A more detailed discussion of these limitations is provided

in the discussion portion of the results section. Instead, we focused our efforts on finding ways to approximate the total kinetic energy loss ΔK_{ship} of the ship from collisions with ice.

We computed an approximation for the energy transferred from the ship to the ice field by performing object tracking on the recorded ice segmentation data. Each ice floe i, represented as a sequence of vertices (i.e., a polygon) at time t_k where $k \in 1 \dots N$, was tracked using a Kalman filter to estimate the position $\boldsymbol{p}^i \in \mathbb{R}^2$ and the velocity $\boldsymbol{v}^i \in \mathbb{R}^2$ of the polygon's centroid. We assumed a constant acceleration model for the dynamics of the centroid. In addition, as part of the state vector we considered the area A of the polygon to be a fixed quantity, with the measurement noise being normally distributed. The estimated area was useful for ignoring particularly poor measurements for a given tracked ice floe. If the IoU (Intersection over Union) between a circle of radius $(\hat{A}^i/\pi)^{1/2}$ centered at $\hat{\boldsymbol{p}}^i$ (the hat denotes the estimate) and the observed polygon was less than a small threshold of 0.2, then the measurement was ignored. This logic helped address the noise present in the segmented ice field, where polygons would occasionally disappear or merge/split between frames. Here we define a 'measurement' to be the particular polygon i' from the segmentation data at time step k that has the smallest Euclidean distance to $\hat{\boldsymbol{p}}_k^i$ (i.e., the predicted a priori state estimate) for the tracked ice floe i. We initialized the set of tracked ice floes by selecting the frame with the largest number of polygons within the first 60 frames or 2 seconds of segmentation data.

From the object tracking data, we identified the set of polygons $\mathcal{P}$ where for each $P \in \mathcal{P}$ the speed in the sequence $\{\|\hat{\boldsymbol{v}}_k^P\|\}_{k=1}^N$ exceeded a threshold of 0.03 m/s over an interval of at least 2 seconds. These two thresholds, picked experimentally, outputted the ice floes believed to have been pushed by the ship at some point during the trial. This includes scenarios where a pushed ice floe is in direct contact with the ship or is touching a chain of other ice floes that are all in contact with each other and are being pushed by the ship. From here, we computed several approximations for the *work* done on the ice by the ship, which is an equivalent definition of ΔK_{ice} when the ice floes are rigid bodies. This provides a reasonable way to compute the kinetic energy loss of the ship ΔK_{ship} in (4.5) where we treat ΔK_{sys} as negligible.

Given the application of a variable force $\boldsymbol{F}$ along a displacement, work is defined as the integral of the dot product between force and displacement. If we consider the work of a net force $\boldsymbol{F}$ applied to a particle with mass m we get

$$W = \int_{t_1}^{t_2} \boldsymbol{F} \cdot \boldsymbol{v} dt = m \int_{t_1}^{t_2} \boldsymbol{a} \cdot \boldsymbol{v} dt, \tag{6.10}$$

where $\boldsymbol{a}$, $\boldsymbol{v}$ is the acceleration and velocity, respectively, of the particle. For our purposes,

we consider the net force acting on the ice floes to consist of the drag force and the force from the ship (either via direct contact or from a chain of ice floes). This means in the absence of a ship force, work W done on the ice will be negative if the ice floe speed $\|\boldsymbol{v}\|$ is nonzero. Leveraging this fact, we can thus approximate the work done by the ship on the ice by only considering the instances of positive work. We used the velocity estimates $\{\hat{\boldsymbol{v}}_k^P\}_{k=1}^N$ for each pushed ice $P \in \mathcal{P}$ tracked by the Kalman filter to give us the total positive work,

$$W_1 = \sum_{P \in \mathcal{P}} m^P \sum_{k=1}^{N-1} (t_{k+1} - t_k) \max(0, \boldsymbol{a}_k^P \cdot \hat{\boldsymbol{v}}_k^P). \tag{6.11}$$

Note, the ice mass m^P is computed as the product of ice thickness, ice density, and the converged area estimate $\hat{A}^P$. Acceleration $\boldsymbol{a}_k^P \in \mathbb{R}^2$ on the other hand was approximated via finite differencing of the velocity estimates. If we ignore the work associated from a resultant torque, (6.11) gives a good approximation for the positive work done by the resultant force applied on a rigid body.

We wanted to see whether a coarser metric (i.e., worse approximation but perhaps less prone to noise) for work would produce similar results as (6.11). Hence, instead of considering an accumulation of positive work, we simply consider the change in kinetic energy for each polygon $P \in \mathcal{P}$ given the maximum speed attained $\hat{v}_{\max}^P = \max\{\|\hat{\boldsymbol{v}}_k^P\|\}_{k=1}^N$ and the initial speed $\|\hat{\boldsymbol{v}}_1^P\|$. Using the work-energy principle, we get our second metric W_2 for the work done by the ship on the ice,

$$W_2 = \frac{1}{2} \sum_{P \in \mathcal{P}} m^P ((\hat{v}_{\max}^P)^2 - \|\hat{\boldsymbol{v}}_1^P\|^2). \tag{6.12}$$

Observe that W_2 does not capture the scenario where an ice floe is pushed on several distinct occasions by the ship meaning $W_1 \geq W_2$. As a second coarse metric for work, we calculate W_3 as the product of the ice mass m^P with the arc length s^P of the path $\{\hat{\boldsymbol{p}}_k^P\}_{k=1}^N$ taken by the polygon $P \in \mathcal{P}$

$$W_3 = \sum_{P \in \mathcal{P}} m^P s^P. \tag{6.13}$$

While this metric lacks a direct connection to work, it offers a straightforward way for capturing the amount by which the ship changed the ice environment.

Other metrics we computed from the object tracking data are the number of pushed ice floes given by the cardinality of the set $\mathcal{P}$ and the mean mass of the set of ice floes that collided with the ship. The latter metric was computed by finding the subset of polygons $\mathcal{P}' \subseteq \mathcal{P}$ that intersected the ship footprint at some point during the trial. We also included the total energy use E given by (6.7) as well as the total time to transit the channel.

6.2.3 Lattice planner parameters

The parameters of our planning framework were similar to the values used for the simulation experiments, and were scaled as necessary for the 20 m $\times$ 6 m ice channel. We increased the costmap resolution to 1/16 m $\times$ 1/16 m. At a nominal speed U_{nom} of 0.2 m/s, we set a corresponding minimum turning radius of $r_{\text{min}} = 1.5$ m. To generate the state lattice, we set the discretization for planar position to 0.75 m $\times$ 0.75 m, and we used the same discretization for heading as in the simulation setup. The algorithm from [14] was used here again to generate the motion primitives.

We deviated from the calibration method described in Section 6.1.4 for setting the parameter α in (3.6). Preliminary tests conducted in the OEB indicated that the ice provided limited resistance to ship motion. This meant that in terms of energy use, the optimal path was simply a straight path. As a result, we manually tuned α to apply a higher penalty on the total collision cost C_f, enabling the planner to produce more interesting paths than simply straight down the channel. As such, the parameter was set to $\alpha = 10$. We set the horizon parameter Δh to 6 meters and the tracking duration parameter Δt to 1 second.

6.2.4 Results

We present a summary of the results from the experiments done in the OEB in Table 6.4 and provide several snapshots of a sample lattice trial in Figure 6.11. The table displays the metrics discussed in the previous section, where scores are averaged across trials for each planner and difficulty level. We start by highlighting the scores for the work metrics. Here, our approach outperformed the baselines in all three difficulty levels. Compared to the second-best score for W_1 from (6.11), we achieved improvements of 29%, 52%, and 48% for the easy, medium, and hard tests, respectively. These relative improvements are similar across the other two work metrics, suggesting that even the lower fidelity metric W_3 provides a useful measure for performance. Figure 6.10 provides a complete picture of the results for W_1. Observe that for the easy trials, our planner provided the smallest gains over navigating straight through the ice field. This gap between straight and lattice grows as the trials become more difficult. In addition, straight is outperformed by both lattice and skeleton for the hard trials. Interestingly, the skeleton planner exhibits similar performance across difficulty levels and performed worse on average for the easy trials. However, it is difficult to draw definitive conclusions due to the limited number (only 3) of easy skeleton trials.

Difficulty	Planner	Mean collided ice mass (kg)	No. pushed ice	W_1 (J)	W_2 (J)	W_3	Total Energy E (J)	Total time (s)
Easy	Straight	1.28	44.13	0.28	0.20	40.91	**20.87**	**91.39**
	Skeleton	1.21	128.00	0.75	0.47	127.11	25.66	115.27
	Lattice	**1.15**	**35.00**	**0.20**	**0.15**	**28.87**	24.55	101.91
Medium	Straight	1.27	68.40	0.44	0.31	65.72	**20.30**	**89.52**
	Skeleton	1.31	102.63	0.59	0.38	106.55	24.29	110.06
	Lattice	**1.17**	**38.00**	**0.21**	**0.13**	**28.60**	24.55	102.90
Hard	Straight	1.22	109.14	0.85	0.52	144.69	**20.82**	**97.15**
	Skeleton	1.24	97.11	0.64	0.41	109.82	24.24	110.73
	Lattice	**1.20**	**49.33**	**0.33**	**0.21**	**53.86**	26.35	109.35

Table 6.4: Results from experiments done in the model scale ice field.

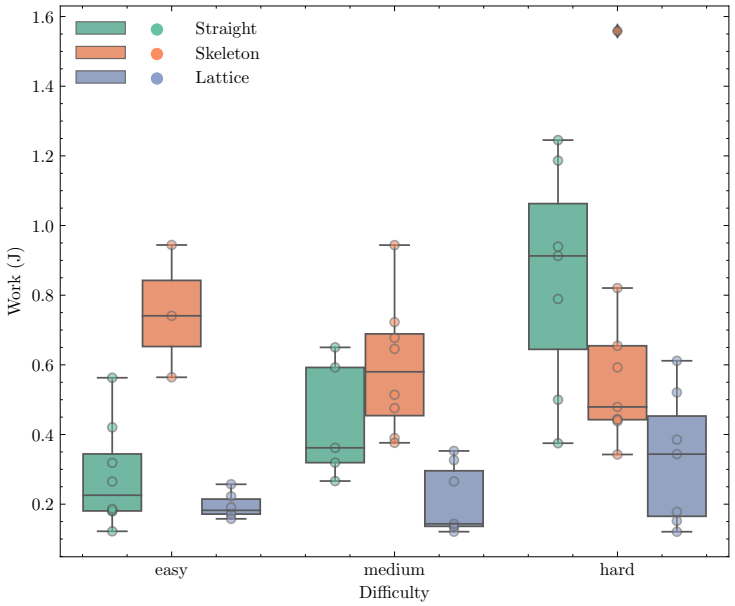


Figure 6.10: Total work done on the ice compared across planner and difficulty level. Work was computed using the object tracking data and (6.11).

We highlight several key limitations of these experiments. Due to resource constraints, we were unable to perform trials with varying ice concentrations like we did in the simulation experiments. It would be interesting to see how our improvements over the baselines compare for higher concentration ice fields, e.g. 40% and 50% ice coverage. We hypothesize that such experiments would show similar performance trends compared to the simulation results described in Section 6.1.7. However, in contrast to the simulation experiments, the plastic ice floes had a narrower range in size and shape, which may have limited the range

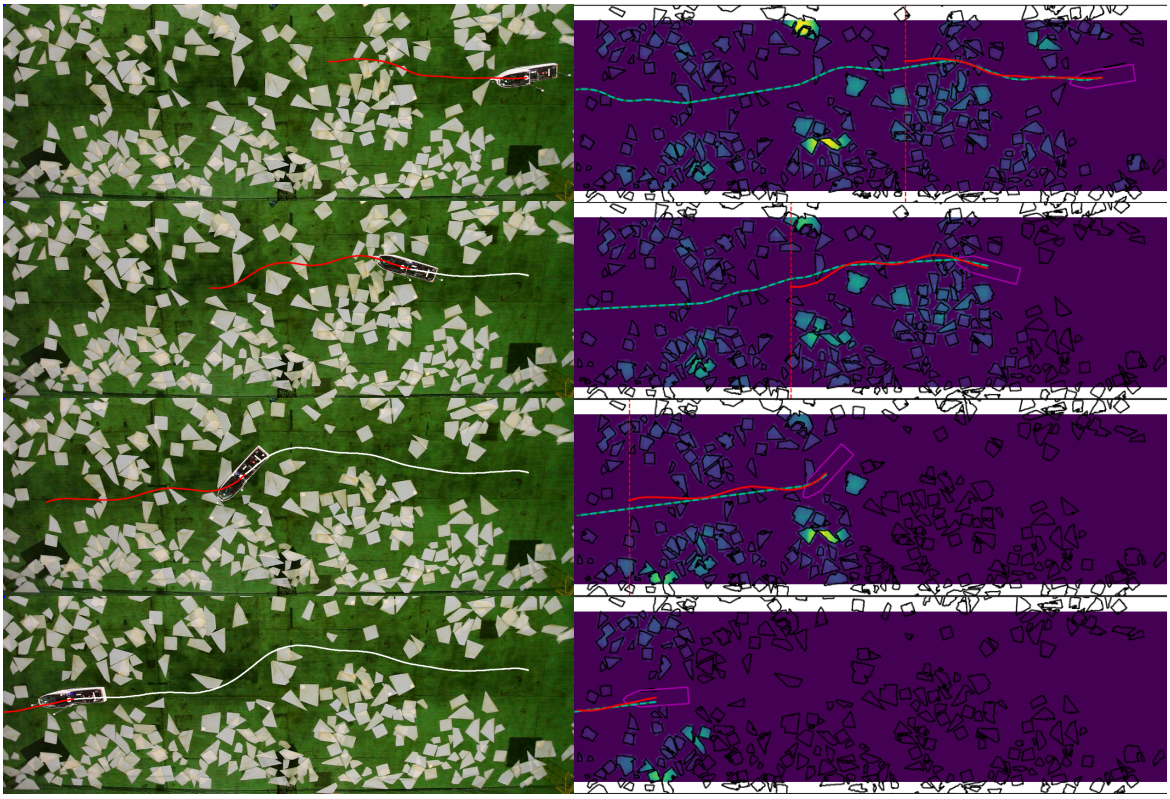


Figure 6.11: Four snapshots from a sample lattice trial showing the camera image on the left and the costmap on the right.

of possible scenarios encountered during ship transit. At a broader level, the small-scale experiment setup and relatively lightweight fake ice made it challenging to simulate full-scale behavior. For instance, the acceleration measurements from the onboard IMU were not very meaningful for comparing the impact forces of ship-ice collisions across trials and planning approaches. We were also unable to make meaningful comparisons in total energy E due to the lack of resistance provided by the fake ice. This means that the approximations we computed for the work done on the ice by the ship seem less significant if we compare them with the total energy used by the ship actuators to transit the ice field. Future experiments would therefore benefit from an increase in ice concentration and ice floe thickness. Alternatively, a better setup would be to use the NRC's 76 m $\times$ 12 m ice tank as in our preliminary work [26] (experiments were only qualitative), which simulates realistic Arctic marine conditions having the ability to grow ice to a maximum thickness of 200 mm.

Chapter 7

Conclusion

We conclude with an overview of the main results presented in this work, followed by a discussion on future work. In this thesis, we examined the problem of local navigation for autonomous ships operating in broken ice fields. This problem requires a local planning strategy that can compute and update solutions in real-time given dynamic ice conditions. We proposed a navigation framework where a smooth reference path and velocity profile are planned in a receding-horizon manner. To capture the cost of ice navigation in the presence of varying ice floes, we formulated an energy-based cost function to calculate the total collision cost accrued from following the planned path under the current ice conditions. Using a simple collision model of ship-ice collisions, we are able to efficiently capture a range of desirable planning behaviors based on the ice floe size and shape distributions. The resulting collision cost is used as part of the objective function in a lattice-based path planner. The planned path is then locally optimized using our proposed second-stage path improvement step. Extensive evaluations validated our method's ability to successfully reduce important costs associated with ice navigation in broken ice fields. Specifically, by decreasing mean and maximum impact forces and improving navigation efficiency compared to naive navigation strategies.

7.1 Limitations and Future Work

Perhaps the most interesting direction for future work and possibly the most significant limitation of our approach is accounting for the changing ice field during the planning stage. This means that instead of relying on frequent replanning to reduce the error in captured ice field state information (i.e., the obstacle footprints $\mathcal{W}_{\text{obs}}$), we use a prediction model of the

ice field to better inform planning behavior. The motion of ice floes is influenced both by the actions of the ship and by environmental disturbances such as wind and ocean currents [10]. This suggests that a useful ice prediction model should receive as input information that encodes how the ship will navigate the environment. Predictions generated by this model could then be used to compute the cost representation of the predicted environment given a particular sequence of ship actions or path segments. Thus, a naive extension to our lattice planning stage is to compute the total collision cost based on the predicted costmap for each candidate plan found during the graph search. However, it is unclear whether computing a cost from a predicted ice field state is useful in practice. Instead, it may be more practical to have a model that computes both the predicted cost and the predicted ice field state. Understanding the relationship between the state of the ice field, the action of the ship, and the associated navigation cost may be a suitable problem for established reinforcement learning (RL) techniques. This would require a sufficiently high-fidelity model of the environment (e.g., a physics simulator) that can be efficiently queried to perform a large number of training iterations. Although, achieving behavior that is both reliable and interpretable remains a significant challenge in RL algorithms.

More broadly, improving the cost function that captures the ship-ice interaction cost is a major direction for future work. Higher-fidelity models for ship-ice collisions are described in [23, 25, 24], which account for additional important features that affect the collision physics, including the angles of the ship's hull and the shapes of the ice edge [23]. However, such models do not translate well with our current costmap-based approach and, therefore, would require novel solutions for designing a suitable cost function for efficient planning. Even more challenging would be to account for the cost associated with the ship breaking through the ice, which typically occurs in head-on collisions with larger ice floes [57]. Another important improvement would be to add constraints or a penalty term for the peak impact force experienced during ship-ice collisions, as it has a greater effect on structural safety than overall energy [57]. Finally, as we observed in the real-world experiments, the sensors that capture the ice data are prone to producing noisy outputs, affecting downstream processing including noisy costmaps. This suggests the use of a probabilistic representation of the ice field where measurements are used to iteratively update beliefs on the shape, position, and orientation of the ice floes. Such a probabilistic approach would be particularly useful in accounting for the varying uncertainties inherent in ship-mounted sensors used to capture the ice field, where errors increase with the distance of ice floes from the ship [90]. This would also allow us to incorporate a notion of *risk* into our objective, where we can use the estimates and the associated uncertainty of the ice data to penalize high-risk scenarios.

References

[1] A Aboulazm and D Muggeridge. Ship model testing in simulated ice flow covered waters. In *SNAME American Towing Tank Conference*, page D011S003R006. SNAME, 1989. 12

[2] Vural Aksakalli, Dindar Oz, Ali Fuat Alkaya, and V Aydogdu. Optimal naval path planning in ice-covered waters. *International Journal of Maritime Engineering*, 159(A1), 2017. 6

[3] Shadi Alawneh, Roelof Dragt, Dennis Peters, Claude Daley, and Stephen Bruneau. Hyper-real-time ice simulation and modeling using gpgpu. *IEEE Transactions on Computers*, 64(12):3475–3487, 2015. 18, 35

[4] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11:1–36, 2019. 32

[5] Ibrahim Ari, Vural Aksakalli, Volkan Aydogdu, and Serdar Kum. Optimal ship navigation with safety distance and realistic turn constraints. *European Journal of Operational Research*, 229(3):707–717, 2013. 1, 6

[6] Kristoffer Bergman, Oskar Ljungqvist, and Daniel Axehill. Improved optimization of motion primitives for motion planning in state lattices. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2307–2314. IEEE, 2019. 6

[7] Kristoffer Bergman, Oskar Ljungqvist, Torkel Glad, and Daniel Axehill. An optimization-based receding horizon trajectory planning algorithm. *IFAC-PapersOnLine*, 53(2):15550–15557, 2020. 6

[8] Kristoffer Bergman, Oskar Ljungqvist, Jonas Linder, and Daniel Axehill. An optimization-based motion planner for autonomous maneuvering of marine vessels

in complex environments. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 5283–5290. IEEE, 2020. 2, 6, 7, 27

[9] Glenn Bitar, Morten Breivik, and Anastasios M Lekkas. Energy-optimized path planning for autonomous ferries. *IFAC-PapersOnLine*, 51(29):389–394, 2018. 6

[10] Glenn Bitar, Andreas B Martinsen, Anastasios M Lekkas, and Morten Breivik. Two-stage optimized trajectory planning for asvs under polygonal obstacle constraints: Theory and experiments. *IEEE Access*, 8:199953–199969, 2020. 2, 6, 7, 22, 25, 27, 56

[11] Victor Blomqvist. Pymunk: A easy-to-use pythonic rigid body 2d physics library. (version 6.7.0), 2024. 35

[12] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 2

[13] Victor Bolbot, Andrei Sandru, Ture Saarniniemi, Otto Puolakka, Pentti Kujala, and Osiris A Valdez Banda. Small unmanned surface vessels—a review and critical analysis of relations to safety and safety assurance of larger autonomous ships. *Journal of Marine Science and Engineering*, 11(12):2387, 2023. 2, 6

[14] Alexander Botros and Stephen L Smith. Computing a minimal set of t-spanning motion primitives for lattice planners. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2328–2335. IEEE, 2019. 40, 52

[15] Alexander Botros and Stephen L Smith. Multi-start n-dimensional lattice planning with optimal motion primitives. *arXiv preprint arXiv:2107.11467*, 2021. 15

[16] Alexander Botros and Stephen L Smith. Spatio-temporal lattice planning using optimal motion primitives. *IEEE Transactions on Intelligent Transportation Systems*, 2023. 26

[17] Xuân-Nam Bui, J-D Boissonnat, Philippe Soueres, and J-P Laumond. Shortest path synthesis for Dubins non-holonomic robot. In *IEEE International Conference on Robotics and Automation*, pages 2–7, 1994. 24

[18] Jean-Baptiste Caillau, Sofya Maslovskaya, Thomas Mensch, Timothée Moulinier, and Jean-Baptiste Pomet. Zermelo-Markov-Dubins problem and extensions in marine navigation. In *IEEE Conference on Decision and Control*, pages 517–522, 2019. 15

[19] Hao-Tien Lewis Chiang and Lydia Tapia. Colreg-rrt: An rrt-based colregs-compliant motion planner for surface vehicle navigation. *IEEE Robotics and Automation Letters*, 3(3):2024–2031, 2018. 6

[20] Aldo Chircop. Sustainable arctic shipping: Are current international rules for polar shipping sufficient? *The Journal of Ocean Technology*, 11(3), 2016. 1

[21] Minjoo Choi, Hyun Chung, Hajime Yamaguchi, and Keisuke Nagakawa. Arctic sea route path planning based on an uncertain ice prediction model. *Cold Regions Science and Technology*, 109:61–69, 2015. 2, 5

[22] DB Colbourne. Scaling pack ice and iceberg loads on moored ship shapes. *Oceanic engineering international*, 4(1):39–45, 2000. 5, 21

[23] Claude Daley. Energy based ice collision forces. In *Proceedings of the 15th International Conference on Port and Ocean Engineering under Arctic Conditions (POAC)*, volume 2, pages 674–686, 1999. 18, 19, 56

[24] Claude Daley, Shadi Alawneh, Dennis Peters, and Bruce Colbourne. Gpu-event-mechanics evaluation of ice impact load statistics. In *OTC Arctic Technology Conference*. OnePetro, 2014. 2, 18, 20, 34, 35, 36, 37, 56

[25] Claude Daley, Shadi Alawneh, Dennis Peters, Bruce Quinton, and Bruce Colbourne. Gpu modeling of ship operations in pack ice. In *SNAME International Conference and Exhibition on Performance of Ships and Structures in Ice*, page D021S005R001. SNAME, 2012. 2, 18, 35, 36, 37, 38, 56

[26] Rodrigue de Schaetzen, Alexander Botros, Robert Gash, Kevin Murrant, and Stephen L. Smith. Real-time navigation for autonomous surface vehicles in ice-covered waters. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1069–1075, 2023. iii, 19, 54

[27] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. Practical search techniques in path planning for autonomous driving. *Ann Arbor*, 1001(48105):18–80, 2008. 22, 25, 27

[28] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. Path planning for autonomous vehicles in unknown semi-structured environments. *The international journal of robotics research*, 29(5):485–501, 2010. 26

[29] Lester E Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3):497–516, 1957. 15

[30] Thor I Fossen. *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2nd edition, 2021. 2, 8, 15, 34, 36, 44

[31] Thor I. Fossen and Tor Arne Perez. Marine systems simulator (mss). https://github.com/cybergalactic/MSS, 2004. 34, 37

[32] Thor I Fossen, Svein I Sagatun, and Asgeir J Sørensen. Identification of dynamically positioned ships. *Control Engineering Practice*, 4(3):369–376, 1996. 34, 37

[33] Maria Fox, Michael Meredith, J Alexander Brearley, Dan Jones, and Derek Long. Long-range route-planning for autonomous vehicles in the polar oceans. *arXiv preprint arXiv:2111.00293*, 2021. 5

[34] Robert M Gash, Kevin A Murrant, Jason W Mills, and David EL Millan. Machine vision techniques for situational awareness and path planning in model test basin ice-covered waters. In *Global Oceans 2020: Singapore–US Gulf Coast*, pages 1–8. IEEE, 2020. 6, 39, 46

[35] Government of Canada. Ice navigation in Canadian waters, May 2019. 1, 2, 12, 15, 20, 43

[36] Government of Canada. Ice glossary, Nov 2020. 2, 17

[37] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of field robotics*, 37(3):362–386, 2020. 2

[38] Dianyuan Han. Comparison of commonly used image interpolation methods. In *Conference of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)*, pages 1556–1559. Atlantis Press, 2013. 27

[39] Carsten Ørts Hansen, Peter Grønsedt, Christian Lindstrøm Graversen, and Christian Hendriksen. *Arctic shipping: commercial opportunities and challenges*. CBS Maritime, 2016. 1

[40] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. 15

[41] Kenta Hasegawa, Shotaro Uto, Haruhito Shimoda, Daisuke Wako, and Takatoshi Matsuzawa. Numerical and experimental investigations of managed ice loads acting on fixed conical structure. In *ISOPE International Ocean and Polar Engineering Conference*, pages ISOPE–I. ISOPE, 2018. 39

[42] Agnieszka Herman. Numerical modeling of force and contact networks in fragmented sea ice. *Annals of glaciology*, 54(62):114–120, 2013. 35

[43] Tsung-Hsuan Hsieh, Shengzheng Wang, Huijia Gong, Wei Liu, and Ning Xu. Sea ice warning visualization and path planning for ice navigation based on radar image recognition. *Journal of Marine Science and Technology*, 29(3):280–290, 2021. 6

[44] Luofeng Huang, Bojan Igrec, and Giles Thomas. New tools to generate realistic ice floe fields for computational models. *Journal of Offshore Mechanics and Arctic Engineering*, 144(4):044503, 2022. 37

[45] Luofeng Huang, Jukka Tuhkuri, Bojan Igrec, Minghao Li, Dimitris Stagonas, Alessandro Toffoli, Philip Cardiff, and Giles Thomas. Ship resistance when operating in floating ice floes: A combined cfd&dem approach. *Marine Structures*, 74:102817, 2020. 2, 35, 37

[46] Malte Humpert and Andreas Raspotnik. The future of arctic shipping along the transpolar sea route. *Arctic Yearbook*, 2012(1):281–307, 2012. 1

[47] Vyacheslav C Khon, II Mokhov, Mojib Latif, Vladimir A Semenov, and Wonsun Park. Perspectives of northern sea route and northwest passage in the twenty-first century. *Climatic change*, 100:757–768, 2010. 1

[48] Hanguen Kim, Donghoon Kim, Jae-Uk Shin, Hyongjin Kim, and Hyun Myung. Angular rate-constrained path planning algorithm for unmanned surface vehicles. *Ocean Engineering*, 84:37–44, 2014. 15

[49] Hyunwook Kim and Claude Daley. Evaluation of ice loads on offshore structure using gpu-event-mechanics. In *ISOPE International Ocean and Polar Engineering Conference*, pages ISOPE–I. ISOPE, 2019. 18

[50] Jeong-Hwan Kim and Yooil Kim. Numerical simulation on the ice-induced fatigue damage of ship structural members in broken ice fields. *Marine Structures*, 66:83–105, 2019. 35

[51] Jeong-Hwan Kim, Yooil Kim, Hyun-Soo Kim, and Seong-Yeob Jeong. Numerical simulation of ice impacts on ship hulls in broken ice fields. *Ocean Engineering*, 182:211–221, 2019. 1, 2, 35

[52] Ville Kotovirta, Risto Jalonen, Lars Axell, Kaj Riska, and Robin Berglund. A system for route optimization in ice-covered waters. *Cold Regions Science and Technology*, 55(1):52–62, 2009. 2, 5

[53] Yoshiaki Kuwata, Michael T Wolf, Dimitri Zarzhitsky, and Terrance L Huntsberger. Safe maritime autonomous navigation with colregs, using velocity obstacles. *IEEE Journal of Oceanic Engineering*, 39(1):110–119, 2013. 6

[54] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006. 7, 8

[55] Ville Lehtola, Jakub Montewka, Floris Goerlandt, Robert Guinness, and Mikko Lensu. Finding safe and efficient shipping routes in ice-covered waters: A framework and a model. *Cold regions science and technology*, 165:102795, 2019. 1, 2, 5, 12

[56] Scott Lembcke. Chipmunk2d: A fast and lightweight 2d game physics library. (version 7.0.2), 2006. 35

[57] Fang Li and Luofeng Huang. A review of computational simulation methods for a ship advancing in broken ice. *Journal of Marine Science and Engineering*, 10(2):165, 2022. 2, 10, 35, 56

[58] Zhiyuan Li, Jonas W Ringsberg, and Francisco Rita. A voyage planning tool for ships sailing between europe and asia via the arctic. *Ships and Offshore Structures*, 15(sup1):S10–S19, 2020. 1, 2

[59] Xiaoyang Liang, Peng Jiang, and He Zhu. Path planning for unmanned surface vehicle with Dubins curve based on GA. In *IEEE Chinese Automation Congress*, pages 5149–5154, 2020. 15

[60] Raed Lubbad, Sveinung Løset, Wenjun Lu, Andrei Tsarau, and Marnix van den Berg. An overview of the oden arctic technology research cruise 2015 (oatrc2015) and numerical simulations performed with sams driven by data collected during the cruise. *Cold Regions Science and Technology*, 156:1–22, 2018. 2, 35, 36, 37

[61] Raed Lubbad, Sveinung Løset, Wenjun Lu, Andrei Tsarau, and Marnix van den Berg. Simulator for arctic marine structures (sams). In *International Conference on Offshore Mechanics and Arctic Engineering*, volume 51296, page V008T07A020. American Society of Mechanical Engineers, 2018. 2, 35, 36

[62] Xiang Ma and David A Castanon. Receding horizon planning for Dubins traveling salesman problems. In *IEEE Conference on Decision and Control*, pages 5453–5458, 2006. 24

[63] Andreas B Martinsen, Anastasios M Lekkas, and Sebastien Gros. Autonomous docking using direct optimal control. *IFAC-PapersOnLine*, 52(21):97–102, 2019. 2

[64] Yu Meng, Yangming Wu, Qing Gu, and Li Liu. A decoupled trajectory planning framework based on the integration of lattice searching and convex optimization. *IEEE Access*, 7:130530–130551, 2019. 27

[65] Brian Vincent Mirtich. *Impulse-based dynamic simulation of rigid body systems*. University of California, Berkeley, 1996. 36

[66] Jakub Montewka, Floris Goerlandt, Mikko Lensu, Lauri Kuuliala, and Robert Guinness. Toward a hybrid model of ship performance in ice suitable for route planning purpose. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 233(1):18–34, 2019. 5

[67] Kevin Murrant, Robert Gash, and Jason Mills. Dynamic path following in ice-covered waters with an autonomous surface ship model. In *IEEE OCEANS*, pages 1–4, San Diego–Porto, 2021. 2, 39, 46

[68] Jong-Ho Nam, Inha Park, Ho Jin Lee, Mi Ok Kwon, Kyungsik Choi, and Young-Kyo Seo. Simulation of optimal arctic routes using a numerical sea ice model based on an ice-coupled ocean circulation method. *International Journal of Naval Architecture and Ocean Engineering*, 5(2):210–226, 2013. 5

[69] Society of Naval Architects, Marine Engineers (U.S.). Technical, and Research Committee. Hydrodynamics Subcommittee. *Nomenclature for Treating the Motion of a Submerged Body Through a Fluid: Report of the American Towing Tank Conference*. Technical and research bulletin. Society of Naval Architects and Marine Engineers, 1950. 8, 34

[70] Wayne Pearson, Mohammed Islam, Michael Lau, Robert Gash, and Jason Mills. Discrete event simulation of vessel stationkeeping operations in ice-rich waters. *Simulation*, 99(7):729–753, 2023. 35

[71] Mihail Pivtoraiko, Ross A Knepper, and Alonzo Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3):308–333, 2009. 6, 14, 16

[72] Drew A Rothrock and Alan S Thorndike. Measuring the sea ice floe size distribution. *Journal of Geophysical Research: Oceans*, 89(C4):6477–6486, 1984. 38

[73] Else-Line M Ruud, Marius S Rundhovde, Jarle Sandrib, and Glenn Bitar. A hybrid-state path planner for asv formations with full-scale experiments. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4841–4848. IEEE, 2023. 6, 15

[74] Christopher Ryan, Luofeng Huang, Zhiyuan Li, Jonas W Ringsberg, and Giles Thomas. An arctic ship performance model for sea routes in ice-infested waters. *Applied Ocean Research*, 117:102950, 2021. 1, 2, 5, 12, 46

[75] Maja Sabolic. Autonomous and remotely controlled ships and ice navigation. Master's thesis, Novia University of Applied Sciences, 2022. 1, 2

[76] Mae L Seto. *Marine robot autonomy*. Springer Science & Business Media, 2012. 6

[77] Tixiao Shan, Wei Wang, Brendan Englot, Carlo Ratti, and Daniela Rus. A receding horizon multi-objective planner for autonomous surface vehicles in urban waterways. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 4085–4092. IEEE, 2020. 6

[78] Jonathan D Smith, Samuel Hall, George Coombs, James Byrne, Michael AS Thorne, J Alexander Brearley, Derek Long, Michael Meredith, and Maria Fox. Autonomous passage planning for a polar vessel. *arXiv preprint arXiv:2209.02389*, 2022. 1, 2, 5, 12, 22

[79] Adam Steer, Anthony Worby, and Petra Heil. Observed changes in sea-ice floe size distribution during early summer in the western weddell sea. *Deep Sea Research Part II: Topical Studies in Oceanography*, 55(8-9):933–942, 2008. 35, 37, 38

[80] Biao Su, Karl Gunnar Aarsæther, and David Kristiansen. Numerical study of a moored structure in moving broken ice driven by current and wave. *Journal of Offshore Mechanics and Arctic Engineering*, 141(3):031501, 2019. 2, 35, 36, 37

[81] Jim Thomson, Stephen Ackley, Fanny Girard-Ardhuin, Fabrice Ardhuin, Alex Babanin, Guillaume Boutin, John Brozena, Sukun Cheng, Clarence Collins, Martin Doble, et al. Overview of the arctic sea state and boundary layer physics program. *Journal of Geophysical Research: Oceans*, 123(12):8674–8687, 2018. 2

[82] Trung Tien Tran. Multi-objective route selection for ice-class vessels using reinforcement learning and graph-based approaches. Master's thesis, Memorial University of Newfoundland, 2021. 2, 5

[83] Trung Tien Tran, Thomas Browne, Mashrura Musharraf, and Brian Veitch. Pathfinding and optimization for vessels in ice: A literature review. *Cold Regions Science and Technology*, 211:103876, 2023. 2, 5

[84] Pavel Valtr. Probability that n random points are in convex position. *Discrete & Computational Geometry*, 13(3-4):637–643, 1995. 38

[85] Aleksander Veksler, Tor Arne Johansen, Francesco Borrelli, and Bjørnar Realfsen. Dynamic positioning with model predictive control. *IEEE Transactions on Control Systems Technology*, 24(4):1340–1353, 2016. 2

[86] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106:25–57, 2006. 32

[87] Weixin Wang, Hui Wang, Guozhong Dai, and Hongan Wang. Visualization of large hierarchical data by circle packing. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 517–520, 2006. 37

[88] Biye Yang, Zhe Sun, Guiyong Zhang, Qingkai Wang, Zhi Zong, and Zhijun Li. Numerical estimation of ship resistance in broken ice and investigation on the effect of floe geometry. *Marine Structures*, 75:102867, 2021. 2, 35, 38, 46

[89] Renat Yulmetov, Raed Lubbad, and Sveinung Løset. Planar multi-body model of iceberg free drift and towing in broken ice. *Cold Regions Science and Technology*, 121:154–166, 2016. 2, 35, 37

[90] Chengqian Zhang, Xiaodong Chen, and Shunying Ji. Semantic image segmentation for sea ice parameters recognition using deep convolutional neural networks. *International Journal of Applied Earth Observation and Geoinformation*, 112:102885, 2022. 2, 11, 56

[91] Chi Zhang, Di Zhang, Mingyang Zhang, and Wengang Mao. Data-driven ship energy efficiency analysis and optimization model for route planning in ice-covered arctic waters. *Ocean Engineering*, 186:106071, 2019. 5

[92] Qin Zhang and Roger Skjetne. Image processing for identification of sea-ice floes and the floe size distributions. *IEEE Transactions on geoscience and remote sensing*, 53(5):2913–2924, 2014. 11

[93] Qin Zhang, Roger Skjetne, Ivan Metrikin, and Sveinung Løset. Image processing for ice floe analyses in broken-ice model testing. *Cold Regions Science and Technology*, 111:27–38, 2015. 11, 20

[94] Tongjie Y Zhang and Ching Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984. 39

[95] Xiaojing Zhang, Alexander Liniger, and Francesco Borrelli. Optimization-based collision avoidance. *IEEE Transactions on Control Systems Technology*, 29(3):972–983, 2020. 27

[96] Jia-yuan Zhuang, Yu-min Su, Yu-lei Liao, and Han-bing Sun. Motion planning of usv based on marine rules. *Procedia Engineering*, 15:269–276, 2011. 6

[97] Matt Zucker, Nathan Ratliff, Anca D Dragan, Mihail Pivtoraiko, Matthew Klingensmith, Christopher M Dellin, J Andrew Bagnell, and Siddhartha S Srinivasa. Chomp: Covariant hamiltonian optimization for motion planning. *The International journal of robotics research*, 32(9-10):1164–1193, 2013. 27, 28