Optimal Path Finding in Direction, Location and Time Dependent Environments

by

Irina Sergeyevna Dolinskaya

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2009

Doctoral Committee:

Professor Robert L. Smith, Chair
Emeritus Professor Stephen M. Pollock
Professor Jing Sun
Associate Professor Marina A. Epelman

To my parents, who have given up so much to bring me to this country and to give me an opportunity to achieve this accomplishment.

ACKNOWLEDGEMENTS

My time in graduate school has been a very educational, exciting and challenging journey. I would like to take this opportunity to acknowledge the people who have had a positive impact on my life over the past five years and who helped me to achieve this milestone.

I would like to thank my adviser, Dr. Robert L. Smith, who has been such a wonderful role model, mentor, colleague and friend. Thank you for all of your time, your numerous advice, and your refreshing perspective on things. I have always enjoyed our research meetings and conversations over coffee.

Thank you to my committee members, Dr. Marina Epelman, Dr. Stephen Pollock and Dr. Jing Sun, for finding time in their busy schedules to help me at the various stages of my graduate career. Each of you have been so much more than a thesis committee member. Dr. Epelman, thank you for making Introduction to Integer and Nonlinear Programming such an interesting course, despite the early starting time during a cold Michigan winter. I greatly enjoyed your clear and precise delivery of material. It had a great impact on my personal approach to optimization. Dr. Pollock, thank you so much for your time and wisdom, especially during my job search process. I could not have done it without you! Thank you, Dr. Sun, for your help and perspective on my research problem. Your insightful questions and comments helped me shape the direction of my research.

A number of other outstanding professors at the University of Michigan have been

a great source of knowledge and inspiration to me. I would like to thank Dr. Vlad Babich, Dr. Xiuli Chao, Dr. Amy Cohn, Dr. Elmer Gilbert, Dr. Jessy Grizzle, Dr. Katta Murty, Dr. Romesh Saigal, Dr. Demosthenis Teneketzis and Dr. Mark Van Oyen for sharing with me their expertise on academic, as well as other, aspects of life. A special thank you to our department chair, Dr. Lawrence Seiford, for his support, his advices, and the opportunities that helped me to successfully complete my studies and get a great academic position.

I would also like to thank my colleagues from the Naval Architecture and Marine Engineering Department, who worked closely with me on the ONR MURI vessel routing project. Dr. Robert Beck, Dr. Miltos Kotinis and Dr. Michael Parsons, thank you for your time and patience while educating me on the concepts of wave propagation, vessel navigation, seakeeping and other topics that helped me to understand the intricate details of our problem.

Thank you to all my fellow IOE students for the unforgettable five years of my life. I remember being told that the friends you make in grad school are your life long friends, and I really hope that we will be able to continue our friendship for the rest of our lives. Special thanks to Katrina Appell, Ada Barlatt, Stan Dimitrov, Kate Heynoski, Tim Lortz, Blake Nicholson, Joy Oguntebi, Betzabe Rodriguez, Sarah Root, Esra Sisikoglu, Tara Terry and Arleigh Waring. Be it proofreading this dissertation, giving me a ride to the airport, sitting through my INFORMS practice presentations, or getting a cup of coffee with me and just being a good friend, I want you to know that I greatly appreciate everything you have done for me!

Thank you to the wonderful staff that takes care of so many things to ensure our department runs smoothly. Thank you, Rod Capps, Chris Konrad and Olof Henrik (Mint) Minto, for all of your invaluable technical support. Thank you, Gwen Brown,

Liz Fisher, Matt Irelan and Mary Winter, for taking care of all of the administrative aspects. A very special thank you to Tina Blay and Wanda Dobberstein. Tina, you take care of every single little detail, especially when I feel too overwhelmed or stressed out to remember them myself. And, Wanda, your frequent advice and sympathy have been so crucial to my teaching and ability to handle a hundred students each term.

I would like to thank my parents and my brother for their unconditional love, support and encouragement through all these years. Mom and Dad, you have given up so much to give me this opportunity, and I really hope that one day I can make it up to you. Thank You!

Last, but most certainly not least, I thank my dear husband, Fernando Tavares. While working on your own Ph.D. degree, you have found the time and energy to be there for all my ups, my downs, and everything in between. Only you know how much your support means to me, and you know that I could not have done this without you. A very special thanks for your expertise and immense help with the coding of my algorithms, I have the numerical results thanks to you!

This dissertation is the result of work partially sponsored by the Office of Naval Research through the Multidisciplinary University Research Initiative (MURI) Optimal Vessel Performance in Evolving Nonlinear Wave-Fields under contract N00014-05-1-0537. This support is gratefully acknowledged.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

CHAPTER I

Introduction

1.1 Problem Overview

Over the past few decades, researchers in a wide range of disciplines have been studying optimal path finding problems within a variety of applications. These approaches find an optimal way to traverse a complex medium or network under a diverse set of constraints, and outside influences like weather. Computational geometry and geographical information systems analyze the shortest paths defined by Euclidean distance and other metrics, often with the presence of polygonal obstacles and weighted homogeneous regions. Optimal robot routing problems integrate the system's physical properties and constraints to find fastest or minimum energy-consumption paths over various terrain. Naval vessel path-finding and navigation integrate the vessel's hull structure and forces exerted by waves and wind to minimize travel time to a destination. Each aforementioned application adds complexity to the original optimal path-finding problem, while integrating a number of assumptions in each scenario to make the problem more tractable. These assumptions often simplify the problem such that it is inappropriate for real-life applications. In our research, we relax a number of restrictive assumptions to create an accurate and tractable model suitable for actual implementation.

We study optimal path finding problems in a direction, location and time dependent environment. Since the objective of a problem depends on the actual application, we do not restrict our analysis to a specific objective function whenever possible. Throughout this dissertation we discuss the problems of minimizing travel time, fuel consumption, and various motions, as well as more general objective functions. The main complexity of the problems arises from the dependence of the constraints and cost function on the location of the mobile agent, the direction it is heading, and time. Additionally, we integrate the system-dynamics to further constrain a feasible path by maximum sharpness of the turn that an agent can make to add another dimension of reality to our model.

Our work delivers a more realistic optimal path-finding model while reducing the computational time required to find such a path. This is particularly important since real-time implementation is essential for our applications. In addition, many analytical results derived here provide insights into the structure of the problem, its objective function, and the optimal solution. These insights provide a closed-form solution to a large subset of problems where additional assumptions are applicable. For such problems, we easily construct the analytical solutions instead of implementing more involved, and often approximate, methods presented in literature.

1.2 Motivation: Optimum Vessel Performance in Evolving Nonlinear Wave-Fields Project

Our research was motivated by an optimal vessel-routing project entitled "Optimum Vessel Performance in Evolving Nonlinear Wave-Fields." This five-year project funded by the Office of Naval Research (ONR) Multidisciplinary University Research Initiative (MURI) grant is a collaboration with the Department of Naval Architecture and Marine Engineering at the University of Michigan, the Applied Physics Labora-

tory at the University of Washington, and the Department of Electrical and Computer Engineering at The Ohio State University. In this section, we provide a brief overview of the project and the research tasks of the teams involved. Throughout the dissertation, we continually revisit this project to illustrate the real-life application of the developed methodology and results.

The goal of this project was to develop a system that can, in real-time, control the behavior of a vessel, based on real-time measurements and forecasts of the wave-field surrounding the vessel. Four major groups divided the project into the following parts based on the areas of expertise:

1. **Real-Time Measurement of Ocean Wave-Fields.** The first group of researchers develops and tests a coherent (Doppler) X-band radar for measuring of the ocean wave-field surrounding a moving or stationary vessel in real-time.

2. **Short-Term Forecasts of Evolving Nonlinear Wave-Fields.** The second team uses data collected by the radar to forecast the time-dependent evolution of the wave-field.

3. **Time-Domain Computation of Nonlinear Ship Motions.** Based on the forecast of the evolving wave-field, this group develops a numerical model to predict nonlinear ship motions in the multidirectional wave-field.

4. **Dynamic Real-Time Path Optimization and Vessel Control.** As part of the fourth team, we use the developed motion-prediction model to evaluate the vessel speed, motions and other operability criteria conditional on a path chosen to traverse the forecasted wave-field. We then integrate this information into our optimal path finding algorithm to determine the most favorable path. An adaptive control system developed by our colleagues guides the vessel along the

found optimal path as closely as possible.

This project considers a wide range of problems, and the objective varies depending on the specific application. Wave-field forecast can be used to predict time periods and areas of calm seas to ensure a safe landing onto an aircraft carrier or a successful launch-and-recovery operation in rough weather. Finding a path that minimizes ship motion is important for improving safety and comfort of the passengers on board. Minimizing travel time is crucial in emergency rescue missions and improves efficiency of the ship-to-ship or port-to-ship cargo transfer operations. Alternatively, finding a path that minimizes fuel consumption instead of the vessel's travel time is favorable for some naval transportation problems. Consequently, in our work we predominantly study a very general set of path finding problems, such that either one of the aforementioned applications can be addressed with our models.

To summarize, our objective as part of this project is to develop computationally efficient and numerically robust algorithms to solve path optimization problems in time-varying media. We are given information about the environment surrounding the vessel up to the radar visibility horizon and the dynamic restriction of the vessel: operability constraints such as probability of capsizing and maximum root mean square roll, and minimum turning radius constraining curvature of a feasible path. We incorporate this information to find an optimal path to a specified desired destination. It is important to note that the wave forecasting model developed as part of this project is precise, and the path finding problem is considered to be deterministic if the initial condition (i.e, the observed wave-field) is accurate.

1.3 Literature Overview

Existing literature details a wide variety of optimal path finding problems. While some work analyzes path finding in a location, and possibly, time dependent medium, others look at the scenarios of anisotropic (i.e., direction-dependent) environment. However, no previous research studies a generalized model that includes all aforementioned aspects of the environment into a single analysis. In this section, we present an overview of various areas of studies and applications that look at the optimal path finding problems as they relate to our work. The subsequent chapters specify different components of our model in more details and provide a more in-depth literature review related to the chapter topic.

Geometric shortest path finding is a fundamental problem extensively studied in computational geometry. Mitchell's survey [45] gives a comprehensive overview of all current work conducted in this field. Most computational geometry research is restricted to finding an optimal path defined by Euclidean distance or other metrics, such as L_1-metric (the *Manhattan distance*) [43] and *C-oriented* paths [71]. Asymmetric direction-dependence is occasionally considered in literature [12, 61], however the introduced anisotropy makes a strong assumption of the distance function convexity which we relax in our analysis. The path-finding problems in a location-dependent environment examine a presence of polygonal obstacles [3, 31, 37, 42, 44] and uniform-weighted regions [11, 47, 67]. On the other hand, all the problems studied in the field of computational geometry are predominantly static, and time-dependence is not considered in these settings. It is important to note that Geographic Information Systems (GIS) is one of the primary application areas for the computational geometry, and a number of papers published at GIS journals

[14, 17, 66, 73] also discuss shortest path finding problems.

Optimal path finding research extends to other applications, such as robot, vessel, airplane and unmanned aerial vehicle routing. In each of these areas, researchers create the models specific to said application; unfortunately their analysis and results cannot be easily transferred to other problems. For example, the problem of computing an optimal path for a mobile robot considers friction and gravity forces for various regions of terrain, and then uses this direction and location dependent cost function to find a path that minimizes the total energy consumption of the robot [34, 62, 63, 68]. Since surface contour does not change over time, this set of problems only considers path finding in a static environment.

Optimal vessel routing evaluates how waves and wind affect vessel speed and dynamics in finding an optimal path. For example, Philpott *et al.* [58] apply mathematical programming methods to create a yacht velocity prediction program that computes the vessel speed for a specified range of wind speeds and yacht headings. The resulting velocity prediction data is used in stochastic dynamic programming models to find the yacht fastest path for uncertain weather [2, 56, 57].

A significant amount of work assumes that the vessel speed function can be written analytically, either generally or specifically. This assumption allows researchers to invoke various methodologies from calculus of variation and optimal control theory to characterize an optimal path [24, 25, 39, 54, 55]. However, researchers typically use a simplified form of the speed function in order to make the analysis more manageable. Our colleagues working on the Optimum Vessel Performance in Evolving Nonlinear Wave-Fields project are developing more accurate and involved models to evaluate vessel dynamics and wave evolution. From our experience of working on this project, it is clear that analytical functions cannot accurately describe the vessel movement

through the waves, thus obliging us to look for alternative methods to solve the problem.

Airline industry researchers analyze how weather affects airplane path planning and air traffic management. For example, Nilim and his colleagues model the weather as Markov chains where storms have a certain probability of becoming the obstacles, thus preventing the airplanes from passing through those regions. Then, a path finding model identifies a path minimizing the expected travel time and dynamically reevaluates the path as more accurate information about the storms becomes available [50, 49]. In their work, Nilim *et al.* assume that the airplanes have a constant velocity, consequently reducing the problem to a shortest path finding problem among stochastic obstacles.

Unmanned Aerial Vehicles (UAVs) have become widely employed in civilian and military applications over the past few years. The problem of optimal path finding for mini UAVs subjected to wind is similar in nature to the vessel routing problems, and has been extensively studied in recent years. The direction dependence of the speed function is introduced as a uniform wind vector field, which is added to a constant isotropic 'wind-free' speed of the airplane [40, 41]. It is important to note that the resulting speed function has very distinct structure, and more specifically, the property of a convex polar plot. We study fastest-path finding problems for the generalized speed functions. Path curvature restrictions are discussed in the UAV path planning problems, however the minimum turning radius is assumed to be constant. In our work, we observe that direction-dependent speed often implies the direction-dependent nature of the minimum turning radius, and we address such problems.

1.4 Dissertation Outline

This dissertation is organized as follows. In Chapter II we discuss optimal path finding in an anisotropic, time and space homogeneous environment. We find a closed form solution for the problems with obstacle-free domain and present a step-by-step algorithm that finds the optimal paths. We employ our findings and adapt a *visibility graph search* method of computational geometry to an anisotropic environment. Consequently, we deliver another algorithm that finds an optimal obstacle-avoiding path in a direction-dependent medium.

Chapter III extends our analysis of path finding in an anisotropic, time and space homogeneous environment to a set of problems where path curvature is constrained by a direction-dependent minimum turning radius function. We invoke techniques from optimal control theory to demonstrate the problem's controllability, prove existence of an optimal path, and derive a necessary condition for optimality. Further analysis characterizes an optimal path and presents an algorithm that facilitates the implementation of the presented results.

The assumption of time and space homogeneity is relaxed in Chapter IV, where we develop a dynamic programming model to find an optimal path in a location, direction and time dependent environment. The results from preceding chapters are integrated into the model to improve its accuracy, efficiency and run-time. The path finding model addresses limited information availability, control-feasibility and computational demands of a time-dependent environment. The application of the developed path finding model to the Optimum Vessel Performance in Evolving Nonlinear Wave-Fields project is also presented in this chapter. The dissertation concludes with Chapter V summarizing the results, contributions and future directions of our

work.

CHAPTER II

Optimal Path Finding in an Anisotropic Time and Space Homogeneous Environment

2.1 Introduction

In this chapter, we address fastest-path finding problems for anisotropic (i.e., direction-dependent) speed functions, which occur, for example, in sailing, robotics, and aircraft navigation. We assume that we know the points of origin and destination, and that time and space homogenous speed is given to us as a function of heading. Our objective is to find a path that minimizes the total travel time from the origin point to the destination. Problems of optimal path finding in an obstacle-free domain and in the presence of polygonal obstacles are discussed in this chapter.

The difficulty of optimal-path finding in anisotropic medium comes from the fact that our travel-time function is asymmetric, that is, the time it takes to travel along a straight line path from a to b, does not necessarily equal the time required to traverse the reversed path ba. Therefore, our cost function is not a metric, which prevents us from using more traditional and established approaches to solving optimal-path finding problems. Furthermore, the anisotropic cost, in general, violates the triangle inequality, which is another key property exploited in the Euclidean shortest-path finding problems. Consequently, it is not guaranteed that one of the 'taut-string' paths has to be an optimal obstacle-avoiding path. Thus, the traditional approach

of searching among a finite number of taut-string paths might not deliver an optimal solution.

One of the most important applications for fastest-path finding algorithms for direction-dependent speed functions is in the area of navigation of autonomous vehicles (AVs). Autonomous vehicles play a crucial role in assisting with a wide range of military and civilian tasks, and their utilization is rapidly increasing as AVs designs and capabilities continue to improve. While able to accomplish the same missions as people thus reducing the risk to human lives, they are also adept in performing sets of tasks they are uniquely capable for. A recent report of the US Department of Defence [70] affirms that "unmanned systems will continue to have a central role in meeting our country's diverse security needs, especially in the Global War on Terrorism." In the report of the National Research Council of the National Academies [48], it is argued that progress in the technologies of computing, robotics, and navigation enables as well as limits the advances in AV capabilities. Therefore, computationally efficient optimal-path finding is an important component to the improvement of current autonomous systems.

In the absence of humans, unmanned systems heavily rely on autonomous navigation systems or autopilots. While some vehicles are remotely operated by people, an increasing number of unmanned systems require only high-level commands from the human, and then use an onboard computer system to perform the task autonomously. The autonomous nature of unmanned systems gives rise to the need for computationally efficient onboard optimal path finding programs that can run in real-time. As the unmanned aerial, ground, and maritime vehicles experience forces created by winds, uneven terrains and waves, their speed functions exhibit the anisotropic property, the main focus of this chapter. For example, mini-Unmanned Aerial Vehicles (UAVs)

that are widely used in military surveillance often face directional wind fields, which create significant anisotropic effects on its speed. We provide analytical solutions to the task of determining a fastest path for these and other direction-dependent domains. The analytic character of these solutions not only provides insight into the properties of an optimal policy but also offers the opportunity for exceptionally fast computation of these paths for on-line implementation.

It is important to note that the analysis and results presented in this chapter apply to a wide range of optimal path finding problems, and not only to the problems of minimum travel time. Despite the fact that our discussion here is limited to the direction-dependent speed functions, it can be easily extended to other anisotropic cost functions, such as fuel and energy consumption or agent's motions.

2.1.1 Related Work

Optimal path planning problems have been studied for a very long time. However, the majority of the to date work concentrates on determining the Euclidean shortest paths (see Mitchell's extensive survey [45]). Even though a number of extensions to optimal path planning have been considered (e.g., traversing through polygonal constantly-weighted regions [47, 67]), most work is restricted to isotropic metrics, where the cost function is assumed to be independent of the traveling direction. Some shortest path finding problems discussed in literature [43, 71] introduce the direction dependency by restricting the feasible paths to a fixed set of orientations, however the resulting cost function preserves its metric properties.

Optimal path finding problems in the anisotropic media have been discussed for some specific applications, however the solution approach and results are often customized to the application at hand. Furthermore, presence of obstacles is not commonly addressed in the published studies. For example, Rowe [62] and Rowe and

Ross [63] study optimal path finding for a mobile agent (e.g., robot or vehicle) across hilly terrains, where a simple and specific physical model of friction and gravity forces is used to compute the anisotropic cost function for the agent.

In the area of optimal yacht sailing, Philpott, Sullivan and Jackson [58] created a mathematical programming model that evaluates the vessel speed for a specified range of wind speeds and yacht heading angles. The resulting velocity prediction data is used to find the yacht fastest path by applying the dynamic programming algorithms [2, 56, 57]. Alternatively, Sellen [64] studies the optimal sailing routing problems for a more abstract scenario, and presents results similar to ours by heuristically arguing that an optimal path in an obstacle-free domain consists of at most two line segments. He also introduces a set of polygonal obstacles and extends his discussion to this restricted domain. However, Sellen's analysis is limited to problems with very specific speed functions represented by piecewise-linear reciprocal functions (i.e., for a direction-dependent speed function denoted by $V(\theta)$, the function $\frac{1}{V(\theta)}$ is assumed to be piecewise-linear). Unlike in the aforementioned work, we make absolutely no assumptions on the structure of the speed function, and find closed form solutions for *any* time and space homogeneous medium.

Some researchers employ the calculus of variations and optimal control theory for optimal vessel routing problems. Faulkner [24, 25], and Papadakis and Perakis [54], utilize Euler's equations to characterize an optimal path; while Kimball and Story [33] establish an analogy between traveling light ray and an optimal path seeking sailboat, and extend the use of optical notions such as Fermat's principle, Huygens' principle and Hamilton's optics to sailing strategies. These optimal-path finding methods reduce to solving systems of differential equations, which can present a difficult and challenging task.

Reif and Sun [61] investigate a problem of time-optimum movement planning through a set of polygonal regions, where anisotropy is introduced as a uniform flow assigned to each region. The actual velocity of an object is defined to be the sum of a flow vector and a chosen control velocity. While the resulting speed function does display the direction-dependent property, its structure is very specific, and Reif and Sun's analysis does not translate to more general problems addressed in this chapter.

In the most recent work on anisotropic movement, Cheng *et al.* [11] generalize the problem studied by Reif and Sun, and look at the shortest path finding in anisotropic regions where the direction-dependency of the speed is not restricted to the effect of the uniform flow. Nevertheless, Cheng *et al.* still limit their research to the speed function with a very specific structure, referred to as a 'convex distance function' (first discussed by Chew and Drysdale [12]). Their convex distance function is equivalent to our case of a convex *linear path attainable region*, however the results presented in our work subsequently relax the convexity assumption and deliver a closed form fastest path among obstacles for a general anisotropic speed function. In addition, we provide rigorous proofs previously omitted in the published work that discussed the convex distance functions.

2.1.2 Overview of the Results

This chapter presents an analytical form solution to the fastest-path finding problem for any given anisotropic speed function. We demonstrate that an optimal path in a general obstacle-free, time and space homogeneous medium is piecewise-linear with at most two line segments (i.e., one waypoint). Consequently, we merge these results with the visibility graph search methods developed for Euclidean shortest path problems [3, 37], to develop an obstacle-avoiding fastest-path finding algorithm for an anisotropic speed function. Our powerful results provide computationally

fast techniques for finding a closed form solution to the very large class of applied problems discussed earlier.

While our main results make no assumptions about the structure of the speed function, we first consider a special case of the problem where the speed polar plot (or the *linear path attainable region*) encloses a convex region. This restricted scenario provides important insight and intuition to the structure of an optimal path for the more general case. Consequently, we relax the convexity assumption to consider a case for a very general speed function. One of our main results is presented in Theorem 2.10, which characterizes a fastest path for an arbitrary speed function in an obstacle-free domain. Algorithm 1 describes the step-by-step procedure to construct such an optimal path. (We note that a fastest path between two points is not necessarily unique, and therefore refrain from using a definite article 'the', and in general refer to it as 'a fastest path'.) In addition to characterizing a fastest path, we also compute a bound on the improvement in travel time were one to choose to follow an optimal path as opposed to traversing the simpler linear path between the two points. This bound is an important tool for evaluating tradeoffs, as well as for proving our key theorem.

We employ our findings for fastest path in an obstacle-free domain to the problems that consider the presence of polygonal obstacles. For the speed functions corresponding to convex linear path attainable regions, the straight line path is a fastest path in $\mathbb{R}^2$, and the triangle inequality holds true in an obstacle-free domain. Consequently, fastest-path finding in a polygonal domain can be restricted to a modified visibility graph, similarly to Euclidian shortest-path finding problems. The triangle inequality might not hold true for a general speed function. In that case, an augmented speed function corresponding to the convex hull of the original speed

polar plot is used to find a lower bound on the minimum travel time for our problem. We use the results for an optimal path in the obstacle-free domain to construct an obstacle-avoiding path that achieves this lower bound, thus implying its optimality.

The rest of the chapter is organized as follows. Subsection 2.1.3 provides the key notation used throughout the chapter and gives a more rigorous statement of the problem. Section 2.2 develops and presents fastest paths for an anisotropic speed function in an obstacle-free domain. This section includes the analysis for a convex linear path attainable region (Subsection 2.2.1); the construction of a bound on the optimal travel time for the general speed function (Subsection 2.2.3); and the later employment of this bound to prove Theorem 2.10 that characterizes a fastest path in a general anisotropic medium (Subsection 2.2.4). In Subsections 2.2.1 through 2.2.4, we assume that a speed function takes on only positive values, where as in Subsection 2.2.5 we discuss the problem of feasibility and fastest paths for the case where speed can be zero for some headings, such as in the cases of stalling or infeasible headings. Subsection 2.2.6 concludes the section with the description of Algorithm 1 that facilitates the implementation of the presented results.

The following Section 2.3 extends our analysis and results to the obstacle-avoiding fastest-path finding problems in anisotropic domain. Similarly to our discussion of the obstacle-free domain, we first find an optimal path for the case of a convex linear path attainable region (Subsection 2.3.1) and then relax the convexity assumption to find a path for an arbitrary speed function (Subsection 2.3.2). Algorithms 2 and 3 describe the fastest-path finding procedures corresponding to each of the cases.

Section 2.4 concludes this chapter with the examples of applications of our results for the vessel and unmanned aerial vehicle routing problems. Finally, Subsection 2.4.2 summarizes the findings and contribution of the presented work.

2.1.3 Notation and Problem Statement

In this section, we introduce the notation and a precise description of the fastest-path finding problem that we analyze in this chapter.

The problem of interest is to find a fastest path from one given point to another for a direction-dependent speed function. We consider two separate scenarios of the problem: (i) an obstacle-free domain where all the feasible paths must lie in $\mathbb{R}^2$; and (ii) a domain containing a set of polygonal obstacles that must be avoided. We are given a direction-dependent speed function, which characterizes the movement within the domain. The speed is assumed to only depend on the heading direction, implying a time and space homogeneous domain (with the exception of obstacles). Next, we introduce the notation to be used throughout the chapter.

Let $\mathcal{P}$ denote a set of open polygonal obstacles, such that their closures do not intersect, or in other words, the distance between any two obstacles is assumed to be greater than zero. Note that since each obstacle is assumed to be an open set, the movement along its edges is permitted. All the feasible paths, including a starting points s and a target point t, are assumed to lie in the *free space*, denoted by $\mathcal{F}$, which we define as the compliment of the obstacles, or $\mathcal{F} := \mathbb{R}^2 \backslash \mathcal{P}$. For consistency, we use this notation and terminology for both aforementioned scenarios, including the obstacle-free case where we set $\mathcal{P} = \emptyset$.

We define P_{st} to be the set of all continuous and rectifiable feasible paths from the start point s to the target point t. That is, $P_{st} = \{p : [0,1] \to \mathcal{F}$ such that $p(0) = s,\ p(1) = t,\ p$ is continuous and rectifiable$\}$. Then, for any $p \in P_{st}$, let $t(p)$ denote the travel time required to traverse the path p.

Let $V(\theta)$ for $0 \leq \theta \leq 2\pi$ denote the maximum attainable speed for a given heading θ. Unless otherwise specified, we assume that $V(\theta) > 0$ for all $\theta \in [0, 2\pi]$.

Allowing the speed function to take on a value of zero for some headings might result in an infeasible problem. Consequently, this case requires special attention and is discussed separately in Section 2.2.5. It is worth noting, that time and space homogeneous nature of our problem, allows us to assume, without loss of optimality, that one always travels at the *maximum* attainable speed, since voluntary speed reduction would never result in a faster path.

We define $L_\delta(x)$ to be the *linear path attainable region* (LPAR) for a given point $x \in \mathbb{R}^2$ and time $\delta > 0$. That is, $L_\delta(x)$ is the set of all points that can be reached in a fixed time period, $\delta > 0$, from point x along a straight line path. In other words, $L_\delta(x) = \{y \in \mathbb{R}^2 : \|y-x\| \leq \delta V(\theta_{y-x})\}$, where θ_{y-x} and $\|y-x\|$ denote the angle and length of a vector $y - x$, respectively. Note that $V(\theta)$ uniquely defines $L_\delta(x)$ for a given x and δ, and vise versa. In the presence of obstacles (i.e., $\mathcal{P} \neq \emptyset$) it is assumed that $x \notin \mathcal{P}$, and δ is too small to reach any obstacles from x. An alternative way to define $L_\delta(x)$, is to introduce an *elementary* LPAR, L, uniquely defined by $V(\theta)$ as $L = \{y \in \mathbb{R}^2 : \|y\| \leq V(\theta_y)\}$. Then from time and space homogeneity we have $L_\delta(x) = x + \delta L$. Note that L is equivalent to a region enclosed by a polar graph of the speed function $V(\theta)$, and is often referred to as the 'speed polar plot' in some literature.

Let $A_\delta(x)$ be the *attainable region* (AR). That is, $A_\delta(x)$ is the set of all points that can be reached in a fixed time period, $\delta > 0$, from point $x \in \mathbb{R}^2$. We can give a more precise definition of $A_\delta(x)$ as follows, $A_\delta(x) = \{y : \exists p \in P_{xy}$ such that $t(p) \leq \delta\}$. Note that in the definition of $A_\delta(x)$, we do not restrict a path to be the straight line path, that is, $A_\delta(x)$ represents the set of all points that can be reach in time δ following *any* rectifiable path from point x. Similarly to the definition of LPAR, we assume that $x \notin \mathcal{P}$, and δ is too small to reach any obstacles from x, if $\mathcal{P} \neq \emptyset$.

Finally, we let the function $\tau(x, y) : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^+$ denote the travel time from point x to point y following the straight line path connecting these two points. We assume that $\tau(x, y)$ is only defined if the straight line segment xy does not intersect the set of obstacles $\mathcal{P}$. Then $\tau(x, y) = \min\{\delta : y \in L_\delta(x),\ \delta > 0\}$. The value of $\tau(x, y)$ can be also computed explicitly using the speed function $V(\theta)$ as $\tau(x, y) = \frac{\|y-x\|}{V(\theta_{y-x})}$. Note that $\tau(x, y)$ is not well defined if $V(\theta_{y-x}) = 0$, in which case we set $\tau(x, y) = \infty$.

Now, we can give the formal statement of our problem.

Problem statement: For a given speed function $V(\theta) : [0, 2\pi] \to \mathbb{R}^+$, a starting point $s \in \mathcal{F}$, and a target point $t \in \mathcal{F}$, find a fastest path from s to t that lies in $\mathcal{F}$. That is, our objective is to find $p^* \in P_{st}$ such that $t(p^*) \leq t(p)$ for all $p \in P_{st}$.

2.2 Fastest-Path Finding for an Anisotropic Speed Function in an Obstacle-Free Domain

In this section we study the fastest-path finding problems in an obstacle-free domain, that is, $\mathcal{P} = \emptyset$ and $\mathcal{F} = \mathbb{R}^2$ for the entire Section 2.2.

2.2.1 Fastest Path for a Convex Linear Path Attainable Region

We first analyze a problem restricted to the convex linear path attainable region (LPAR), as this special case gives an intuitive and insightful analysis of the problem, and it then can be extended to a general case. Therefore, throughout Section 2.2.1, we assume that $L_\delta(x)$ is convex for all x and δ; in other words, the convex combination of any two points in the set $L_\delta(x)$ is contained by that set (see Figure 2.1). Note that from the time and space homogeneity of the speed function $V(\theta)$, we know that if $L_\delta(x)$ is convex for some specific x and δ, then it is convex for all $x \in \mathbb{R}^2$ and all positive δ.

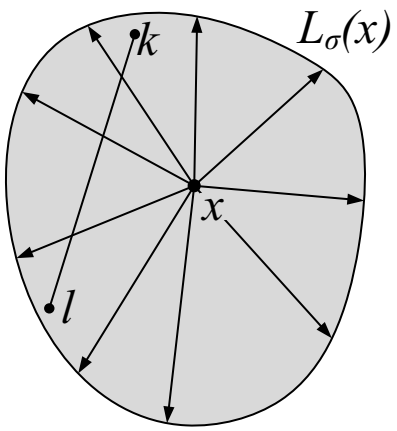

Figure 2.1: An example of a convex linear path attainable region $L_\delta(x)$.

Let $m(x)$ denote the smallest non-negative scalar such that $L_{m(x)}(0,0)$ contains point x. We can also write $m(x) := \inf\{r : \frac{x}{r} \in L_1(0,0), r > 0\}$. Observe that since $L_1(0,0)$ is a closed set, the infimum is achieved and the definition can we rewritten as $m(x) := \min\{r : \frac{x}{r} \in L_1(0,0), r > 0\}$ as long as $x \neq (0,0)$. Also, note that since $L_1(0,0) = \frac{1}{m(x)} L_{m(x)}(0,0)$ is a convex set in $\mathbb{R}^2$ and $(0,0)$ is its interior point, we conclude that $m(x)$ is the Minkowski functional. We then know from [38] that the Minkowski functional $m(.)$ satisfies the inequality $m(x_1 + x_2) \leq m(x_1) + m(x_2)$ for all $x_1, x_2 \in \mathbb{R}^2$. A couple of algebraic manipulations lead to the fact that $m(x)$ reduces to the straight line travel time function $\tau(.)$, that is, $\tau(x,y) = m(y-x)$ for all $x, y \in \mathbb{R}^2$. We now show that the equivalent inequality holds true for the travel time function τ.

Lemma 2.1. *For any $x, y, z \in \mathbb{R}^2$, we have $\tau(x,y) \leq \tau(x,z) + \tau(z,y)$. That is, traveling time along the straight line path from x to y is never greater than the time it takes to travel along straight lines from x to z and then from z to y. (See Figure 2.2)*

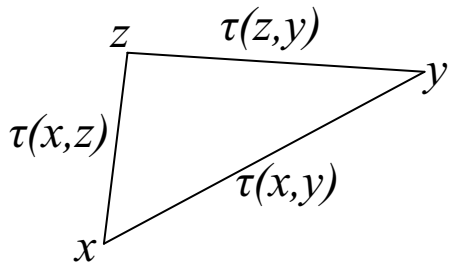

Figure 2.2: Illustration of the inequality from Lemma 2.1, $\tau(x, y) \leq \tau(x, z) + \tau(z, y)$.

Proof.

$$\begin{aligned}
\tau(x, y) &= m(y - x) \\
&= m(y - x + z - z) \\
&= m((z - x) + (y - z)) \\
&\leq m(z - x) + m(y - z) = \tau(x, z) + \tau(z, y).
\end{aligned}$$

□

We now can use the inequality from Lemma 2.1 to show that the straight line path between two points is a fastest path for a convex LPAR, $L_\delta(x)$. Recall that $p \in P_{st}$ is an arbitrary continuous and rectifiable path from s to t, and that $t(p)$ denotes the travel time along the path p. Then we can state the following lemma.

Lemma 2.2. *For a convex $L_\delta(x)$ and an arbitrary continuous and rectifiable path $p \in P_{st}$, we have $\tau(s, t) \leq t(p)$. In other words, in the case of a convex LPAR, the travel time along the straight line path is never greater than that of any other path.*

Proof. Since the length of any continuous and rectifiable path p, in the limit, equals to the length of a piecewise-linear approximation, we can iteratively apply the inequality from Lemma 2.1 to obtain the desired result.

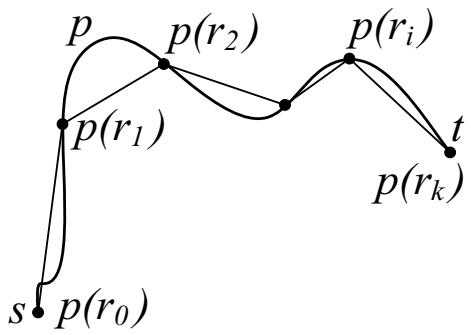

Figure 2.3: Polygonal path approximation.

To compute $t(p)$ we apply polygonal approximation to the path p. Choose an arbitrary partition Π of interval $[0,1]$, i.e., let $\Pi = (r_0, r_1, r_2, ..., r_k)$ such that $0 = r_0 < r_1 < r_2 < ... < r_{k-1} < r_k = 1$. Let mesh $|\Pi|$ be the maximum length $r_i - r_{i-1}$ of a subinterval of Π, that is, $|\Pi| = \max_{1 \le i \le k}\{r_i - r_{i-1}\}$. Then Π defines a polygonal approximation to p, i.e., the polygonal arc from $p(0)$ to $p(1)$ having successive vertices $p(r_0)$, $p(r_1)$, ..., $p(r_k)$ (see Figure 2.3).

Then, the traveling time along the polygonal approximation of the path can be written as $\eta(p, \Pi) = \sum_{i=1}^{k} \tau(p(r_{i-1}), p(r_i))$. However, as we let $|\Pi|$ approach zero, thus increasing the number of vertices, the polygonal approximation in the limit is equal to path p; then so are their travel times (this follows from the assumption that path p is rectifiable). Given this,

$$t(p) = \lim_{|\Pi| \to 0} \eta(p, \Pi). \tag{2.1}$$

Note that for any arbitrary partition Π, $\eta(p, \Pi) = \tau(p(r_0), p(r_1)) + \tau(p(r_1), p(r_2)) + ... + \tau(p(r_{k-1}), p(r_k))$. After iteratively applying inequality from Lemma 2.1 we obtain $\eta(p, \Pi) \ge \tau(s, t)$. Substituting this into equation (2.1) results in $\tau(s, t) \le t(p)$. □

Lemma 2.2 above provides the fastest path between two points for a convex LPAR. Furthermore, the following theorem adds that convexity of an LPAR is also a neces-

sary condition for the straight line path to be optimal.

Theorem 2.3. *A fastest path in $\mathbb{R}^2$ from an arbitrary start point $s \in \mathbb{R}^2$ to any other point in $\mathbb{R}^2$ is a path along the straight line connecting the two points* ***if and only if*** *the linear path attainable region $L_\delta(x)$ is a convex set for all $x \in \mathbb{R}^2$.*

Proof. Lemma 2.2 concludes that a fastest path from an arbitrary start point $s \in \mathbb{R}^2$ to any other point in $\mathbb{R}^2$ is a path along the straight line connecting the two points **if** the linear path attainable region $L_\delta(x)$ is a convex set for all $x \in \mathbb{R}^2$.

Now, we prove the **only if** statement of the theorem by contradiction.

Select an arbitrary start point $s \in \mathbb{R}^2$ and assume that $L_\delta(s)$ is not convex. Then, there exist $x_1, x_2 \in L_\delta(s)$ and $\lambda \in [0, 1]$ such that $\lambda x_1 + (1 - \lambda)x_2 \notin L_\delta(s)$, and we set point $y = \lambda x_1 + (1 - \lambda)x_2$.

Since $x_1, x_2 \in L_\delta(s)$, we have $\tau(s, x_1) \leq \delta$ and $\tau(s, x_2) \leq \delta$. Then, consider the following path p: from point s, we first travel following the vector $\lambda(x_1 - s)$ and then continue on following the vector $(1 - \lambda)(x_2 - s)$ (Figure 2.4). Our path p starts at point s and ends at point $s + \lambda(x_1 - s) + (1 - \lambda)(x_2 - s) = \lambda x_1 + (1 - \lambda)x_2 = y$. Note that time and space homogeneity give us that traveling time for this path, $t(p) = \lambda\tau(s, x_1) + (1 - \lambda)\tau(s, x_2) \leq \lambda \cdot \delta + (1 - \lambda) \cdot \delta = \delta$. However, $\tau(s, y) > \delta$ since $y \notin L_\delta(s)$. We reach a contradiction that the straight line path from s to any point in $\mathbb{R}^2$ is not necessarily the fastest path if the attainable region is not convex. Thus, $L_\delta(s)$ has to be convex. □

It is important to acknowledge that earlier work on 'convex distance functions' [9, 11, 12] have stated some results similar to Lemma 2.1 and Theorem 2.3. However, none of the found literature provides a rigorous proof in its entirety, compelling us to include our proofs developed independently of the cited literature.

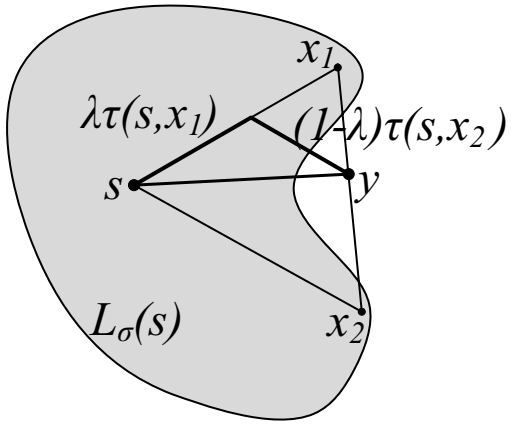

Figure 2.4: Theorem 2.3 counter example for a non-convex linear path attainable region.

Next, we analyze optimal path finding for a general LPAR, which may or may not be convex.

2.2.2 Properties of an Attainable Region and the Corresponding Linear Path Attainable Region

From this point on, we relax the convexity assumption for the linear path attainable region, $L_\delta(x)$, and analyze the problem for a general time and space homogeneous speed function. In this section, we provide a series of lemmas, theorems and propositions stating supporting properties of LPARs and the corresponding attainable regions, ARs. Lemmas represented here are the building blocks for our main results presented in the following sections.

Proposition 2.4. $L_\delta(x) = A_\delta(x)$ ***if and only if*** $L_\delta(x)$ *is convex.*

Proof. First, we prove the **if** direction of the proposition by contradiction.

Assume $L_\delta(x) \neq A_\delta(x)$. From the definitions of $L_\delta(x)$ and $A_\delta(x)$, we know that $L_\delta(x) \subseteq A_\delta(x)$. Then, $L_\delta(x) \subset A_\delta(x)$, that is, $\exists y \in A_\delta(x)$, s.t. $y \notin L_\delta(x)$. Hence, there exists a non-linear path p from point x to point y, such that traveling time along this path is less than traveling time along a straight line path from x to y. However,

Theorem 2.3 states that for a convex linear path attainable region, a straight line path is the fastest path between any two points in $\mathbb{R}^2$. Thus, we reach a contradiction and conclude that if $L_\delta(x)$ is convex, then $L_\delta(x) = A_\delta(x)$.

Next, we prove the **only if** direction of the proposition by contradiction.

Assume that $L_\delta(x) = A_\delta(x)$ but $L_\delta(x)$ is not convex. From Theorem 2.3 we know that if $L_\delta(x)$ is not convex, then $\exists x, y \in \mathbb{R}^2$ such that the straight line path from x to y is not a fastest path. Let δ_{xy} be the minimum travel time from x to y. Then $y \in A_{\delta_{xy}}(x)$ but $y \notin L_{\delta_{xy}}(x)$ since traveling time along the straight line segment from x to y will be greater than δ_{xy}. This contradicts our assumption that $L_\delta(x) = A_\delta(x)$. Hence, our assumption that $L_\delta(x)$ is not convex was incorrect. □

Comparison of Two Distinct LPARs and Their Corresponding ARs.

Consider two arbitrary maximum attainable speed functions $V^1(\theta)$ and $V^2(\theta)$ defined on $\theta \in [0, 2\pi]$. Recall that each speed function uniquely defines the linear path attainable region for a given time interval, and vise versa. Thus, let $L^1_\delta(x)$ and $L^2_\delta(x)$ be the linear path attainable regions corresponding to the speed functions $V^1(\theta)$ and $V^2(\theta)$, respectively. Then we can make the following observations about $L^1_\delta(x)$ and $L^2_\delta(x)$.

Lemma 2.5. *$L^1_\delta(x) \subseteq L^2_\delta(x)$ **if and only if** $V^1(\theta) \leq V^2(\theta)$ for all θ.*

Proof. We first show that if $V^1(\theta) \leq V^2(\theta)\ \forall\theta$ then $L^1_\delta(x) \subseteq L^2_\delta(x)$. Select an arbitrary point $y \in L^1_\delta(x)$. Then from the definition of $L^1_\delta(x)$, we have $\|y - x\| \leq \delta V^1(\theta_{y-x})$. Since $V^1(\theta) \leq V^2(\theta)\ \forall\theta$, we know that $\|y - x\| \leq \delta V^2(\theta_{y-x})$ as well. Hence, $y \in L^2_\delta(x)$. Since point $y \in L^1_\delta(x)$ was chosen arbitrarily, we can conclude that $L^1_\delta(x) \subseteq L^2_\delta(x)$.

Next, we prove the other direction of the lemma by contradiction. We need to show

that if $L^1_\delta(x) \subseteq L^2_\delta(x)$ then $V^1(\theta) \leq V^2(\theta)\ \forall\theta$. Assume that $V^1(\theta) \not\leq V^2(\theta)$ for some θ. Then, $\exists\theta^* \in [0, 2\pi]$ such $V^2(\theta^*) < V^1(\theta^*)$. Select $y \in L^1_\delta(x)$ such that $\theta^* = \theta_{y-x}$ and $\|y - x\| = \delta V^1(\theta_{y-x})$. Then $\delta V^2(\theta^*) < \delta V^1(\theta^*) = \delta V^1(\theta_{y-x}) = \|y - x\|$ and $y \notin L^2_\delta(x)$. We reach a contradiction, since $L^1_\delta(x) \subseteq L^2_\delta(x)$. Hence, our assumption that $V^1(\theta) \not\leq V^2(\theta)$ for some θ was incorrect. □

Lemma 2.6. *Let $A^1_\delta(x)$ and $A^2_\delta(x)$ be the attainable regions corresponding to linear path attainable regions $L^1_\delta(x)$ and $L^2_\delta(x)$, respectively. Then, $L^1_\delta(x) \subseteq L^2_\delta(x)$ implies $A^1_\delta(x) \subseteq A^2_\delta(x)$.*

Proof. From Lemma 2.5, we know that if $L^1_\delta(x) \subseteq L^2_\delta(x) \Rightarrow V^1(\theta) \leq V^2(\theta)\ \forall\theta$. Thus, for any heading direction speed $V^2(\theta)$ is always at least as great as $V^1(\theta)$. Select an arbitrary $y \in A^1_\delta(x)$. From the definition of attainable region $A^1_\delta(x)$, there exists a continuous path $p : [0, 1] \to \mathbb{R}^2$ from point x to point y, such that if a mobile agent's maximum speed function is $V^1(\theta)$, the travel time from x to y is no greater than δ, i.e., $t_{V^1}(p) \leq \delta$. Now, consider following this path p with the maximum speed given by function $V^2(\theta)$. Since $V^1(\theta) \leq V^2(\theta)\ \forall\theta$, we know that travel time along path p with speed corresponding to function $V^2(\theta)$, $t_{V^2}(p)$, will be at most $t_{V^1}(p)$. Hence, $t_{V^2}(p) \leq t_{V^1}(p) \leq \delta$ and thus, point y also belongs to set $A^2_\delta(x)$. Since $y \in A^1_\delta(x)$ was chosen arbitrarily, we can conclude that $A^1_\delta(x) \subseteq A^2_\delta(x)$. □

Attainable Region Corresponding to a Given Linear Path Attainable Region.

The problem discussed in our work assumes that a maximum attainable speed function $V(\theta)$ is given for all $\theta \in [0, 2\pi]$. Since the speed function uniquely defines the linear path attainable region for a given x and δ, $L_\delta(x)$, one can always use the definition of the LPAR to find $L_\delta(x)$ corresponding to a given function $V(\theta)$. Theorem 2.3 establishes the fact that a straight line is not necessarily the fastest

path for a non-convex $L_\delta(x)$, and from Proposition 2.4 we know that $L_\delta(x) \neq A_\delta(x)$ if $L_\delta(x)$ is not convex. Thus, finding the attainable region, $A_\delta(x)$, corresponding to a given speed function is not always a straight forward task. In this section, we establish how one can find the attainable region corresponding to a given $L_\delta(x)$.

Lemma 2.7. *The convex combination of any two points in $L_\delta(x)$ is contained in $A_\delta(x)$, i.e., $\forall x_1, x_2 \in L_\delta(x)$ and $\forall \lambda \in [0,1]$, $\lambda x_1 + (1-\lambda)x_2 \in A_\delta(x)$.*

Proof. Select arbitrary $x_1, x_2 \in L_\delta(x)$ and $\lambda \in [0,1]$. Note that $\lambda x_1 + (1-\lambda)x_2$ may not lie in $L_\delta(x)$, since set $L_\delta(x)$ might not be a convex set. Let $y = \lambda x_1 + (1-\lambda)x_2$. Since $x_1, x_2 \in L_\delta(x)$, $\tau(x, x_1) \leq \delta$ and $\tau(x, x_2) \leq \delta$.

Now consider the following path p: from point x, we travel following the vector $\lambda(x_1 - x)$ and then, continue on following the vector $(1-\lambda)(x_2 - x)$ (This path is the same path as the one constructed in the proof of Theorem 2.3 which can be seen in Figure 2.4.) Then, our path p starts at point x and ends at point $x + \lambda(x_1 - x) + (1-\lambda)(x_2 - x) = \lambda x_1 + (1-\lambda)x_2 = y$. Using time and space homogeneity, we can find the travel time for this path: $t(p) = \lambda\tau(x, x_1) + (1-\lambda)\tau(x, x_2) \leq \lambda \cdot \delta + (1-\lambda) \cdot \delta = \delta$. Consequently, $y \in A_\delta(x)$. Since x_1, x_2 and λ were chosen arbitrarily, we can conclude that the set of all convex combinations of points from $L_\delta(x)$ lies in $A_\delta(x)$. □

Theorem 2.8. *Attainable region, $A_\delta(x)$, is the convex hull of the corresponding linear path attainable region, $L_\delta(x)$, i.e., $A_\delta(x) = conv(L_\delta(x))$.*

Proof. The statement $conv(L_\delta(x)) \subseteq A_\delta(x)$ follows directly from Lemma 2.7. It is worth noting, that path p constructed in the lemma's proof is not necessarily a fastest path from x to y, it is just a path that reaches point y in time less than or equal to δ.

Next, we show that $A_\delta(x) \subseteq conv(L_\delta(x))$. Consider a new linear path attainable region $L'_\delta(x) = conv(L_\delta(x))$. Since our linear path attainable region $L'_\delta(x)$ is convex, from Proposition 2.4 we know that the corresponding attainable region $A'_\delta(x) = L'_\delta(x)$. Since $L_\delta(x) \subseteq L'_\delta(x)$, from Lemma 2.6 it follows that $A_\delta(x) \subseteq A'_\delta(x)$. And since $A'_\delta(x) = conv(L_\delta(x)) \Rightarrow A_\delta(x) \subseteq conv(L_\delta(x))$. Hence, $A_\delta(x) = conv(L_\delta(x))$.

□

2.2.3 Bound on the Optimal Travel Time

From Theorem 2.3, we know that sometimes a straight line path is not necessarily a fastest path for a given speed function $V(\theta)$. In particular, a straight line is the fastest path for a convex linear path attainable region, but not necessarily so for a non-convex region. Here, we calculate a bound on the shortest travel time error if the straight line path is implemented for a non-convex LPAR. A lower bound on the minimum travel time is not only important for assessing the penalty for deviating from the optimal path by following a straight line, but the bound also plays a significant role in the proof of our key result: by showing that in some cases the travel time for our proposed path is equal to the lower bound, we prove its optimality.

Consider a non-convex linear path attainable region, $L^1_\delta(x)$, corresponding to some speed function $V^1(\theta)$. Then, we can calculate a bound on a decrease in the travel time from point x to point y by following an optimal path instead of the straight line path, without actually knowing the optimal path. Consider a new linear path attainable region defined as the convex hull of the original LPAR, that is, $L^2_\delta(x) = conv(L^1_\delta(x))$. And let $V^2(\theta)$ be the maximum attainable speed function associated with the new LPAR. From Theorem 2.3, we know that for $L^2_\delta(x)$, the fastest path from x to y is along the straight line segment connecting these two points, l_{xy}, with the total

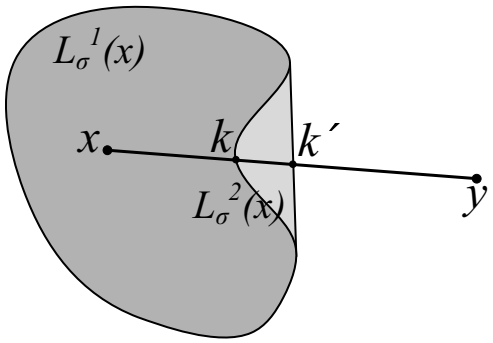

Figure 2.5: Computing a bound on the decrease in travel time for a non-convex linear path attainable region, $L^1_\delta(x)$.

travel time $\tau_2(x,y) = \frac{\|y-x\|}{V^2(\theta_{y-x})}$. Since $L^1_\delta(x) \subset L^2_\delta(x)$, from Lemma 2.5 we know that $V^1(\theta) \leq V^2(\theta)$ for all θ. Then, the smallest travel time from x to y for the linear path attainable region $L^1_\delta(x)$, denoted by $t^*_{L^1_\delta(x)}(x,y)$, is at least as much $\tau_2(x,y)$, i.e., $t^*_{L^1_\delta(x)}(x,y) \geq \tau_2(x,y)$.

Define k to be the point of intersection of the line connecting points x and y, and the boundary of the linear path attainable region $L^1_\delta(x)$, i.e., $k := l_{xy} \cap bd(L^1_\delta(x))$. Similarly, we define $k' := l_{xy} \cap bd(L^2_\delta(x))$ (see Figure 2.5). Note that the sets $L^1_\delta(x)$ and $L^2_\delta(x)$ are closed and therefore contain their boundaries. Also note that the travel time along the straight line path from x to y corresponding to the linear path attainable region $L^1_\delta(x)$ is $\tau_1(x,y) = \delta\frac{\|y-x\|}{\|k-x\|}$. Set $\beta := \frac{\|k-x\|}{\|k'-x\|} \leq 1$. Then, we have the following bounds on $t^*_{L^1_\delta(x)}(x,y)$.

$$\begin{aligned}
\tau_2(x,y) \leq \;& t^*_{L^1_\delta(x)}(x,y) && \leq \tau_1(x,y) \\
\delta\frac{\|y-x\|}{\|k'-x\|} \leq \;& t^*_{L^1_\delta(x)}(x,y) && \leq \delta\frac{\|y-x\|}{\|k-x\|} \\
\beta\delta\frac{\|y-x\|}{\|k-x\|} \leq \;& t^*_{L^1_\delta(x)}(x,y) && \leq \delta\frac{\|y-x\|}{\|k-x\|} \\
\beta\tau_1(x,y) \leq \;& t^*_{L^1_\delta(x)}(x,y) && \leq \tau_1(x,y)
\end{aligned} \tag{2.2}$$

From inequalities (2.2), we deliver the following proposition.

Proposition 2.9. *The optimal travel time for a non-convex LPAR is at most β times shorter than following a straight line path from x to y, where $\beta := \frac{\|k-x\|}{\|k'-x\|}$. That is, the traveling time would at most decrease by $100(1-\beta)$ percent, if one were to follow an optimal path instead of traveling along the straight line.*

This lower bound is next used to show that a proposed path is, in fact, optimal.

2.2.4 Fastest Path for an Arbitrary Linear Path Attainable Region

In the earlier Section 2.2.1 we solved an instance of our fastest-path finding problem for a convex linear path attainable region (LPAR). In this section, we describe a closed form solution to our fastest path problem in $\mathbb{R}^2$ for any speed function, even when corresponding $L_\delta(x)$ fails to be convex. The following theorem is one of the key results of this chapter.

Theorem 2.10. *Consider a linear path attainable region $L_\delta(x)$. For two arbitrarily given points $x, y \in \mathbb{R}^2$, let k denote the intersection point of the line connecting x and y, l_{xy}, and the boundary of the set $L_\delta(x)$, i.e., $k := l_{xy} \cap bd(L_\delta(x))$. Similarly, let $k' := l_{xy} \cap bd(conv(L_\delta(x)))$. Then, the fastest path from x to y is described by one of the following two scenarios.*

1. *If $k = k'$, the fastest path from x to y is the straight line segment connecting these two points (Figure 2.6).*

2. *If $k \neq k'$, the fastest path from x to y consists of two line segments: the straight line segment from point x to point $z = x + \alpha\lambda^*(x_1 - x)$ and the second line segment from point z to point y, where $\alpha = \frac{\|y-x\|}{\|k'-x\|}$ and $x_1, x_2 \in L_\delta(x)$ s.t. $\exists \lambda^* \in [0,1] : k' = \lambda^* x_1 + (1-\lambda^*)x_2$. (See Figure 2.7, and note that $(y-z) \| (x_2 - x)$).*

Proof. 1. Consider the case where $k = k'$. From inequalities (2.2), we have $\beta\tau(x,y) \leq$

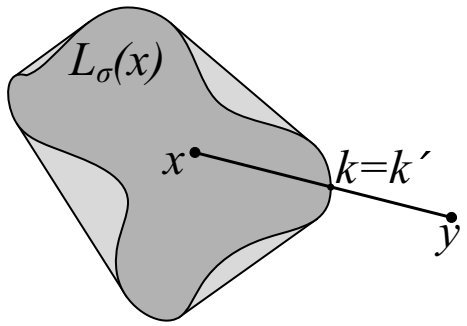

Figure 2.6: Illustration of Theorem 2.10 scenario 1: $k = k'$.

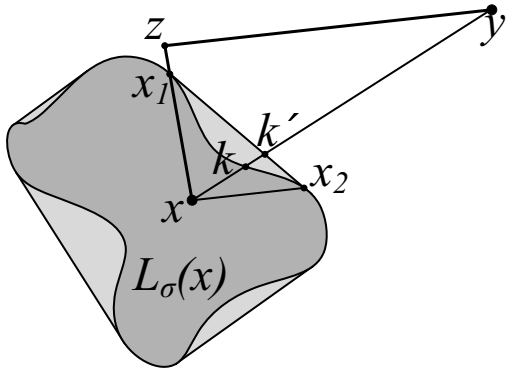

Figure 2.7: Illustration of Theorem 2.10 scenario 2: $k \neq k'$.

$t^*_{L_\delta(x)}(x, y) \leq \tau(x, y)$, where $\beta := \frac{\|k-x\|}{\|k'-x\|}$ and $t^*_{L_\delta(x)}(x, y)$ is the minimum travel time from x to y. Since $k = k'$, we have $\beta = 1$, and $\tau(x, y) \leq t^*_{L_\delta(x)}(x, y) \leq \tau(x, y) \Rightarrow t^*_{L_\delta(x)}(x, y) = \tau(x, y)$. This means that the travel time from x to y along the straight line path equals the minimum travel time, and hence, straight line path is a fastest path from x to y.

2. Now, we consider the case where $k \neq k'$. From definition of k', we have that $k' \in conv(L_\delta(x))$. Then, $\exists \lambda^* \in [0, 1]$ and $\exists x_1, x_2 \in L_\delta(x)$, such that $\lambda^* x_1 + (1 - \lambda^*)x_2 = k'$. Note, that since $x_1, x_2 \in L_\delta(x)$, we know that $\tau(x, x_1) \leq \delta$ and $\tau(x, x_2) \leq \delta$.

   From inequalities (2.2), we have $t^*_{L_\delta(x)}(x, y) \geq \delta \frac{\|y-x\|}{\|k'-x\|} = \delta\alpha$, where $t^*_{L_\delta(x)}(x, y)$ is the minimum travel time from x to y. Now, consider the following path p: from

point x we follow vector $\alpha\lambda^*(x_1 - x)$, and then, continue on following vector $\alpha(1-\lambda^*)(x_2-x)$. Note, that the first part of the path is equivalent to following a straight line segment from point x to point $x + \alpha\lambda^*(x_1 - x) = z$. And the second part of the path ends at point $x + \alpha\lambda^*(x_1 - x) + \alpha(1 - \lambda^*)(x_2 - x) = x + \alpha((\lambda^*)(x_1 - x) + (1 - \lambda^*)(x_2 - x)) = x + \alpha(k' - x) = y$. Hence, the proposed path p is the same path as in the statement of the theorem. This proves the existence of the path described in the theorem.

Next, we want to find the travel time along this path p, $t(p)$. From the space and time homogeneity property, we have $t(p) = \alpha\lambda^*\tau(x, x_1) + \alpha(1 - \lambda^*)\tau(x, x_2) \leq \alpha\lambda^* \cdot \delta + \alpha(1-\lambda^*) \cdot \delta = \alpha\delta$. Since travel time for path p is less than or equal to the lower bound on the minimum travel time from x to y (i.e., $t(p) \leq t^*_{L_\delta(x)}(x, y)$), $t(p)$ must be equal to the minimum travel time from x to y. Hence, our path p is, in fact, a fastest path from x to y.

□

It is worth noting that in the case when $k \neq k'$ (corresponding to scenario 2 of Theorem 2.10) the fastest path constructed in the theorem is not uniquely optimal. It is only one of the infinitely many feasible paths with the same minimum travel time. Note that any zigzag path from x to y restricted to the traveling directions of the vectors $x_1 - x$ and $x_2 - x$ would correspond to the same minimum travel time. Furthermore, the straight line path in the case of $k = k'$ might also not be uniquely optimal. Depending on the structure of the speed function, it is possible that a piecewise-linear path would have the same optimal travel time as the straight line.

2.2.5 Problem Feasibility and Fastest-Path Finding for a Non-negative Speed Function

All the analysis and results presented above assume that $V(\theta) > 0$ for all $\theta \in [0, 2\pi]$. However in practice, the speed function $V(\theta)$ can take on the value of zero for some headings, i.e., leading to a 'stall'. For example, a vehicle traveling across some hilly terrain might encounter impermissible headings due to overturn danger or power limitations [63]. On another hand, a sailing boat can not travel in head sea corresponding to a zero speed for that heading [57]. In this section, we discuss how allowing $V(\theta)$ to take on zero values for some headings can change the results presented in the previous sections and in particular, its possible effects on problem feasibility. Consequently, we will allow $V(\theta) \geq 0, \forall \theta \in [0, 2\pi]$ for the discussion in this section. To avoid the trivial case, we assume that there always exists some θ such that $V(\theta) > 0$.

Feasibility and Optimal Path Finding for a Convex LPAR.

Similar to a positive speed function case, convexity of a linear path attainable region (LPAR) is a useful property that simplifies the optimal path finding task. Therefore, we first analyze the case where LPAR, $L_\delta(s)$, is convex. In the following section, we look at a more general case where $L_\delta(s)$ does not have to be convex. Lemma 2.11 below helps establish the existence of a feasible path from s to t.

Lemma 2.11. *If $V(\theta) = 0$ for some $\theta \in [0, 2\pi]$, and the corresponding linear path attainable region, $L_\delta(s)$, is convex, then there exists a line passing through the starting point s such that none of the points belonging to one of the half-planes created by this line can be reached. See Figure 2.8.*

Proof. If $V(\theta) = 0$ for some θ, then s has to be a boundary point of the convex set

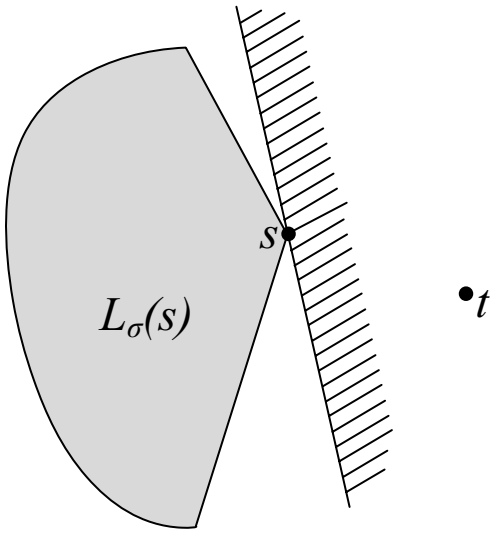

Figure 2.8: Example of a convex LPAR where $V(\theta_{st}) = 0$; there is no feasible path from s to t.

$L_\delta(s)$. Therefore, there exists a supporting line passing through s such that $L_\delta(s)$ lies on one side of this line. Consequently, there is no linear combination of feasible headings that would deliver us to any point belonging to the other half-space. □

Next, Theorem 2.12 describes an optimal path from s to t for a convex LPAR.

Theorem 2.12. *Assume that LPAR, $L_\delta(s)$, corresponding to some speed function $V(\theta) \geq 0$, is convex. Then,*

1. *if $V(\theta_{st}) = 0$, a feasible path from s to t does not exist; and*

2. *if $V(\theta_{st}) > 0$, a fastest path from s to t is along the straight line path st.*

Proof. 1. Proof of this statement follows from Lemma 2.11. We can construct a supporting line to $L_\delta(s)$ at point s, that separates the LPAR and point t. Concluding, that no feasible path from s to t exists (see Figure 2.8).

2. To prove the second statement, we select an arbitrary $\epsilon > 0$, such that $\epsilon < \min_\theta\{V(\theta) : V(\theta) > 0\}$. Then, define a new speed function $V'(\theta)$ as follows (see

Figure 2.9).

$$V'(\theta) = \begin{cases} V(\theta), & \text{if } V(\theta) > 0 \\ \epsilon, & \text{if } V(\theta) = 0 \end{cases} \tag{2.3}$$

By construction, $V(\theta) \leq V'(\theta)$ for all $\theta \in [0, 2\pi]$. Then, from Lemma 2.5 and Lemma 2.6 we know that a fastest path corresponding to the speed function $V(\theta)$ cannot be faster than an optimal path corresponding to $V'(\theta)$. Since $V'(\theta) > 0$ for all θ, we can apply Theorem 2.10 to find a fastest path from s to t corresponding to that speed function. Note that from $\epsilon < \min_\theta\{V(\theta) : V(\theta) > 0\}$, we know that the intersection point of the line st with the boundaries of LPAR and the intersection point of line st with LPAR's convex hull are equal to each other, corresponding to scenario 1 of Theorem 2.10. Therefore, an optimal path for the speed function $V'(\theta)$ is a straight line path st. Since $V'(\theta_{st}) = V(\theta_{st})$, the straight line path is also feasible for the original speed function, and it has the same travel time. Hence, st is an optimal path for the original speed function $V(\theta)$.

□

Feasibility and Optimal Path Finding for an Arbitrary LPAR.

We now relax the convexity assumption for a linear path attainable region and analyze optimal paths for a general $L_\delta(s)$. Note that results presented below apply to a convex as well as a non-convex LPAR cases. However, if one knows that the linear path attainable region is convex, application of Theorem 2.12 would be more straight forward.

Recall that θ_{st} denotes the heading angle of the vector $t - s$. It is apparent that if $V(\theta_{st}) > 0$, then the optimal path finding problem is feasible. We are interested

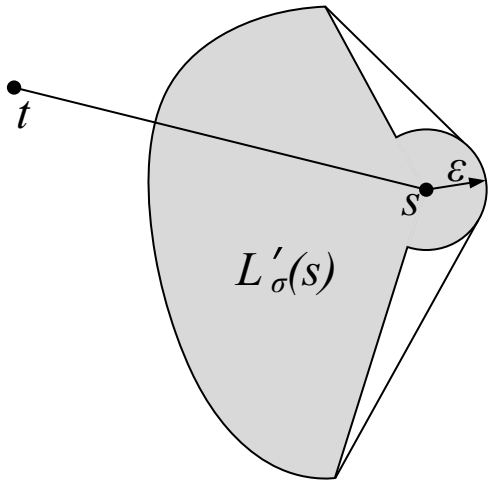

Figure 2.9: Linear path attainable region corresponding to the speed function $V'(\theta)$.

in describing necessary and sufficient conditions for the problem to be infeasible. Assuming $V(\theta_{st}) = 0$, we can define $\underline{\theta}$ and $\bar{\theta}$ as given below.

$$\underline{\theta} = \inf\{\theta^* : V(\theta) = 0, \forall\theta \in [\theta^*, \theta_{st}]\} \tag{2.4}$$

$$\bar{\theta} = \sup\{\theta^* : V(\theta) = 0, \forall\theta \in [\theta_{st}, \theta^*]\} \tag{2.5}$$

Note that *infimum* and *supremum* in equations (2.4) and (2.5) might not be actually achieved. Also note that in defining $\underline{\theta}$ and $\bar{\theta}$ we extend the domain of the speed function to $[-\pi, 3\pi]$, by observing that $V(\theta) = V(\theta + 2\pi), \forall\theta$. This extension is necessary to guaranty the continuity of the interval at the boundary points $\theta = 0$ and $\theta = 2\pi$. Now, we are ready to state our problem feasibility theorem.

Theorem 2.13. *A feasible path from s to t does not exist* ***if and only if*** $V(\theta_{st}) = 0$ *and* $\bar{\theta} - \underline{\theta} \geq \pi$.

Proof. To prove the **if** statement of the theorem, we observe that if $V(\theta) = 0, \forall\theta \in (\underline{\theta}, \bar{\theta})$, $\bar{\theta} - \underline{\theta} \geq \pi$ and $\theta_{st} \in [\underline{\theta}, \bar{\theta}]$, then no linear combination of feasible headings would deliver you from point s to point t. Figure 2.8 provides a visual example.

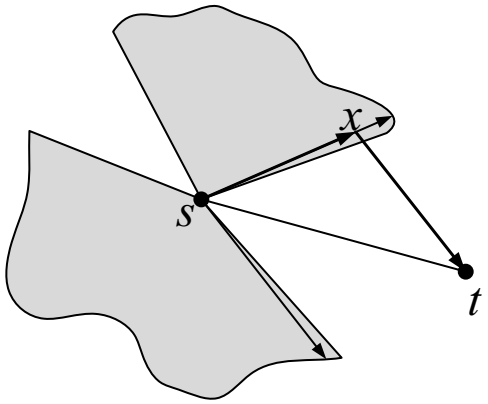

Figure 2.10: Existence of a feasible path from s to t: path sxt.

Next, we prove the **only if** direction of the theorem by contradiction. Assume that there does not exist a feasible path from s to t, but either $V(\theta_{st}) \neq 0$ or $\bar{\theta} - \underline{\theta} < \pi$. Recall that $\underline{\theta}$ and $\bar{\theta}$ are not defined if $V(\theta_{st}) \neq 0$, thus it is not possible for both conditions to be violated simultaneously. Clearly, if $V(\theta_{st}) \neq 0 \;\; \Rightarrow V(\theta_{st}) > 0$, which would mean that the straight line path from s to t is feasible. On the other hand, if $\bar{\theta} - \underline{\theta} < \pi$, then $\exists \epsilon_1, \epsilon_2 \geq 0$ such that $V(\underline{\theta} - \epsilon_1) > 0, V(\bar{\theta} + \epsilon_2) > 0$ and $(\bar{\theta} + \epsilon_2) - (\underline{\theta} - \epsilon_1) < \pi$. Therefore, $\exists \alpha \in [0, 1]$ such that $\theta_{st} = \alpha(\underline{\theta} - \epsilon_1) + (1 - \alpha)(\bar{\theta} + \epsilon_2)$; and hence we can construct a feasible path from s to t by first traveling in the direction $\underline{\theta} - \epsilon_1$ and then turning to the direction $\bar{\theta} + \epsilon_2$. See Figure 2.10. We reached a contradiction which proves that the original assumption of nonexistence of a feasible path was incorrect. □

Next, Theorem 2.14 delivers an optimal path from s to t for a general linear path attainable region.

Theorem 2.14. *Consider $V(\theta) \geq 0$ for all $\theta \in [0, 2\pi]$, and let $L_\delta(s)$ be the corresponding linear path attainable region. Then,*

1. *if $V(\theta_{st}) = 0$ and $\bar{\theta} - \underline{\theta} \geq \pi$, a feasible path from s to t does not exist; and*

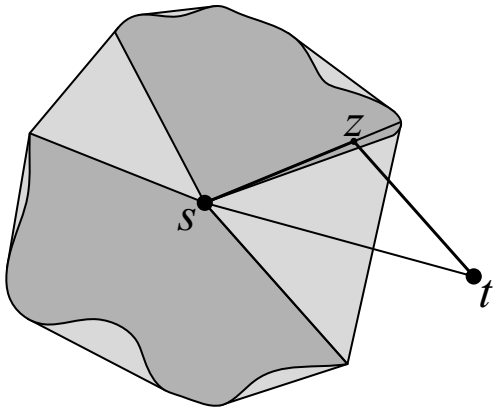

Figure 2.11: An optimal path from s to t, szt.

2. *if* $V(\theta_{st}) > 0$ *or* $\bar{\theta} - \underline{\theta} < \pi$, *then a fastest path from* s *to* t *is characterized the same way as in Theorem 2.10, where* $x = s$, $y = t$ *and if* $V(\theta_{st}) = 0$ *we set* $k = s$. *(See Figure 2.11.)*

Proof. 1. Proof of the first statement follows directly from Theorem 2.13.

2. *[Sketch]* The proof of this statement is analogous to the proof of Theorem 2.12 part 2. The optimality of a path characterized in Theorem 2.10 has been only proven for the positive speed function, where s is an interior point of the corresponding LPAR. Thus, we first define the new speed function $V'(\theta)$ as given by equation (2.3), then we apply Theorem 2.10 to the new speed function, and finally, we show that the found path is also feasible and has the same travel time for the original speed function $V(\theta)$, making it an optimal path for our problem.

□

2.2.6 Fastest-Path Finding Algorithm

Sections 2.2.1-2.2.5 characterize an optimal path between any two points in $\mathbb{R}^2$ for an arbitrary speed function $V(\theta)$. In this section, we discuss the implementation

of the presented results and provide an algorithm that can be implemented by a computer program (e.g., on-board autonomous navigation system) to find a fastest path from a given start point $s \in \mathbb{R}^2$ to a given target point $t \in \mathbb{R}^2$. The presented algorithm checks the feasibility of the problem as discussed in Section 2.2.5 and then implements the results of Theorem 2.10 in the case of a feasible problem.

Since, in practice, the direction-dependent speed is usually evaluated by a computer program, we assume that the values of a speed function $V(\theta)$ is given for a discrete set of equally spaced heading angles, θ, which we denote by the set of polar coordinates $S = \{(\theta_0, V(\theta_0), ..., (\theta_n, V(\theta_n)\}$. (The speed values for the intermediate heading angles are assumed to be equal to the linear interpolation within a polar coordinate system, see Figure 2.12 for example.) Note that in the case when an analytical function of $V(\theta)$ is available, we still have to discretize the speed function in order to be able to implement the fastest-path finding procedure by a computer.

Our first step is to check the feasibility of the problem. Theorem 2.13 states the necessary and sufficient condition for a problem to be infeasible. If those conditions are not satisfied, we know that an optimal path exist and we can proceed to finding such a path.

The first step in finding a fastest path is to construct a convex hull of the linear path attainable region. Construction of a convex hull of a finite number of points in $\mathbb{R}^2$ is a well studied problem, and its details are omitted. However, we recommend the use of Graham's Scan algorithm [16, 28] to accomplish this task. The advantage of this algorithm is that it uses a technique called "rotational sweep," processing vertices in the order of polar angles they form with a reference vertex. The polar nature of our LPAR makes Graham's Scan a favorable choice as it forgoes the sorting procedure required for other algorithms.

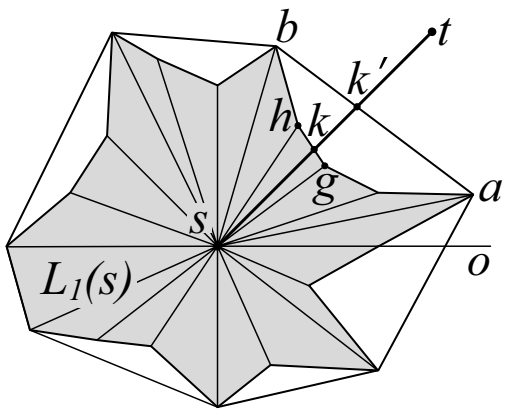

Figure 2.12: $L_1(s)$ and its convex hull: $\angle ost = \theta_{st}$, $\angle osg = \theta_L$, $\angle osh = \theta_U$, $\angle osa = \theta'_L$, and $\angle osb = \theta'_U$.

After the construction of a convex hull, we obtain a subset $S' \subseteq S$ corresponding to the extreme (corner) points of the resulting convex hull. Furthermore, the convex combination of two consecutive points in S' is part of the boundary of $conv(L_1(s))$. (Just like, all convex combinations of pairs of consecutive points in S is the boundary of $L_1(s)$.) Let l_{st} denote the straight line passing through points s and t, and θ_{st} the heading angle of the vector $t - s$. Then, to apply Theorem 2.10, we need to find the point of intersection of l_{st} with the boundary of $L_1(s)$, denoted by k, and the point of intersection of l_{st} with the boundary of a convex hull of $L_1(s)$, denoted by k'. To do so, we find between which two headings in sets S and S' our θ_{st} falls. We label such headings as θ_L and θ_U, and θ'_L and θ'_U, respectively (See Figure 2.12, L and U stand for the lower and upper headings).

We know that k lies on the line segment connecting points $(\theta_L, V(\theta_L))$ and $(\theta_U, V(\theta_U))$, and k' lies on the line segment connecting points $(\theta'_L, V(\theta'_L))$ and $(\theta'_U, V(\theta'_U))$. Based on that, we can use the found $\theta_L, \theta_U, \theta'_L$ and θ'_U to determine whether $k = k'$ without actually finding the points k and k'. Note that if points $(\theta_L, V(\theta_L)), (\theta_U, V(\theta_U)), (\theta'_L, V(\theta'_L))$ and $(\theta'_U, V(\theta'_U))$ (corresponding to points g, h, a and b on Figure 2.12) are all collinear,

then k must equal k', and $k \neq k'$ otherwise. If $k = k'$, we conclude that line segment st is the fastest path as proven in scenario 1 of Theorem 2.10. If $k \neq k'$, our problem reduces to scenario 2 of the theorem, and we need to compute the values of α and λ^*, as defined in Theorem 2.10. After some algebraic manipulations omitted here, we find that

$$\alpha\lambda^* = \frac{\|t - s\| \sin(\theta'_U - \theta_{st})}{V(\theta'_L) \sin(\theta'_U - \theta'_L)}. \tag{2.6}$$

Then, we know that a fastest path is piecewise-linear with a single waypoint $z = s + \alpha\lambda^*(\cos(\theta'_U), \sin(\theta'_U))$.

The following algorithm outlines a step-by-step procedure of finding the fastest path from s to t.

Algorithm 1 Fastest Path from s to t in an Obstacle-Free Domain.

Step 1. Find $\underline{\theta}$ and $\bar{\theta}$ using equations (2.4) and (2.5).
If $V(\theta_{st}) = 0$ and $\bar{\theta} - \underline{\theta} \geq \pi$, STOP. The problem is infeasible.
Else, go to step 2.

Step 2. Find the convex hull of the linear path attainable region $L_1(s)$.

Step 3. Find the heading angle θ_{st} and compute the values of $\theta_L, \theta_U, \theta'_L$ and θ'_U.

Step 4. If points $(\theta_L, V(\theta_L)), (\theta_U, V(\theta_U)), (\theta'_L, V(\theta'_L))$ and $(\theta'_U, V(\theta'_U))$ are collinear (i.e., if $k = k'$), STOP. Straight line path st is an optimal path.
Else (i.e., if $k \neq k'$), go to step 5.

Step 5. Compute $\alpha\lambda^*$ using equation (2.6).
Set $z = s + \alpha\lambda^*(\cos(\theta'_U), \sin(\theta'_U)) \in \mathbb{R}^2$. A fastest path from s to t is the two consecutive straight line segments sz and then zt.

2.3 Obstacle-Avoiding Fastest-Path Finding for an Anisotropic Speed Function

In this section we discuss obstacle-avoiding fastest-path finding by relaxing the assumption made in Section 2.2 that $\mathcal{P} = \emptyset$. Throughout this section, $\mathcal{P}$ is a nonempty set of polygonal obstacles that are not permitted to be intersected by any feasible path.

2.3.1 Fastest Path for a Convex Linear Path Attainable Region

Similarly to our analysis of path finding problems in an obstacle-free domain, we first restrict our attention to problems with speed functions corresponding to the convex linear path attainable regions (LPARs). The analysis of this special case demonstrates an interesting insight into the structure of the solution for the problems with arbitrary speed functions. In the following subsection, we relax the convexity assumption and show how the results presented here are extended to the unrestricted time and space homogenous anisotropic speed functions.

The visibility graph search method, used to solve Euclidean shortest-path finding problems, exploits the triangle inequality property of the distance function and restricts the optimal path search to the set of 'taut strings' connecting the points of origin and destination. Similar properties can be established for the fastest-path finding problem at hand. Our Theorem 2.10 states that in the case of a convex LPAR a fastest path between any two points in an obstacle-free anisotropic domain is the connecting straight line segment. Consequently, the triangle inequality, restated in Corollary 2.15 for completeness, also holds true for our anisotropic cost function (travel time). We use this property to develop a fastest-path finding algorithm analogous to the one used for Euclidean shortest path problems.

Corollary 2.15. *If a speed function $V(\theta)$ corresponds to a convex linear path attainable region, then the travel-time function $\tau(.)$ has the 'triangle inequality' property, that is, $\tau(a,b) \leq \tau(a,c) + \tau(c,b)$, $\forall a,b,c \in \mathcal{F} = \mathbb{R}^2 \backslash \mathcal{P}$, as long as neither one of the linear paths are obstructed by obstacles.*

Proof. Follows directly from Lemma 2.1. □

The triangle inequality stated in Corollary 2.15 provides the grounds for a direct

extension of the earlier mentioned visibility graph search method to our anisotropic problem. In the case of minimizing Euclidean distance, "an easy geometric argument shows that in general the shortest path between two points must be a polygonal chain whose vertices are vertices of obstacles" [3]. A similar observation is true for our anisotropic medium, which validates the search of a modified visibility graph as an appropriate solution approach for our problem.

Theorem 2.16. *If a linear path attainable region $L_\delta(x)$ is convex, then there exists a fastest path from s to t in $\mathcal{F}$, which is piecewise-linear with all its waypoints (vertices) corresponding to the vertices of obstacles in $\mathcal{P}$.*

Proof. Corollary 2.15 and the polygonal structure of the obstacles imply that any continuous path $p \in \mathcal{F}$ from s to t can be replaced by a piecewise-linear path from s to t such that the travel time of the piecewise-linear path is not greater than that of the initial path p. Therefore, there exists a piecewise-linear path which is a fastest path from s to t. (See Figure 2.13 for a visual illustration.)

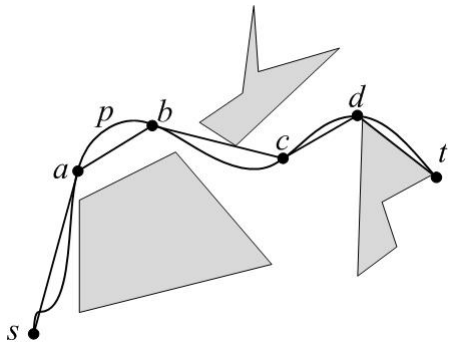

Figure 2.13: For a convex LPAR, the travel time along the piecewise-linear path *sabcdt* is not greater than along the curve p.

Next, we show that there exists an optimal piecewise-linear path such that all its vertices correspond to the obstacle vertices. Consider a piecewise-linear path with some vertex a not equal to a vertex of any obstacle in $\mathcal{P}$. Then, there exist two points

b and c lying on each of the two line segments of the polygonal path joined by vertex a, such that the line segment bc does not intersect $\mathcal{P}$. We construct a new path by replacing the bac part of the path with a straight line segment bc. From the triangle inequality of Corollary 2.15 we know that the travel time for the resulting path is not greater than the travel time for the original path. It follows that there exists a fastest path which is piecewise-linear and its vertices correspond to the vertices of obstacles in $\mathcal{P}$. (See Figure 2.14.) □

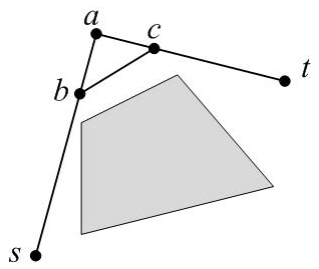

Figure 2.14: For a convex LPAR, the travel time along the piecewise-linear path *sbct* is not greater than along the path *sat*.

Theorem 2.16 implies that when a given speed function corresponds to a convex linear path attainable region, a fastest-path search can be restricted to a directed visibility graph with the edge cost defined to be the travel time along the straight line connecting its nodes. Henceforth, we adapt the shortest path visibility graph approach to develop the algorithm below, which finds an obstacle-avoiding fastest path for an anisotropic speed function corresponding to a convex LPAR.

Published work discussing Euclidean shortest path problems notes that in some cases the visibility graph has quadratic size (i.e., the construction time of a graph with n vertices is $O(n^2)$), and is not the most efficient approach for such problems [45]. An alternative method, referred to as *continuous Dijkstra*, involves simulating the effect of a 'wavefront' propagating out from the source s and constructs the linear-

Algorithm 2 Obstacle-Avoiding Fastest Path for a Speed Function $V(\theta)$ Corresponding to a Convex LPAR.

Step 1. Construct a visibility graph $\mathcal{VG}_V$ as follows.

- The set of $\mathcal{VG}_V$ vertices is composed of all the vertices of the obstacles in $\mathcal{P}$, as well as points s and t.
- The set of $\mathcal{VG}_V$ edges consists of all the straight line edges interconnecting these vertices such that they do not intersect any of the obstacles in $\mathcal{P}$.
- The cost associated with an edge (i, j) is equal to the travel time $\tau(i, j) = \frac{||j-i||}{V(\theta_{j-i})}$. (Note that unlike the case of Euclidean metric, our visibility graph has to be directed since the cost of an arc (i, j) does not generally equal to the cost of an arc (j, i).)

Figure 2.15 provides an example of constructing a visibility graph by illustrating all the visibility graph nodes and edges.

Step 2. Apply Dijkstra's algorithm [19] to find an optimal path in the constructed network $\mathcal{VG}_V$ from node s to node t. The resulting path is an obstacle-avoiding fastest path.

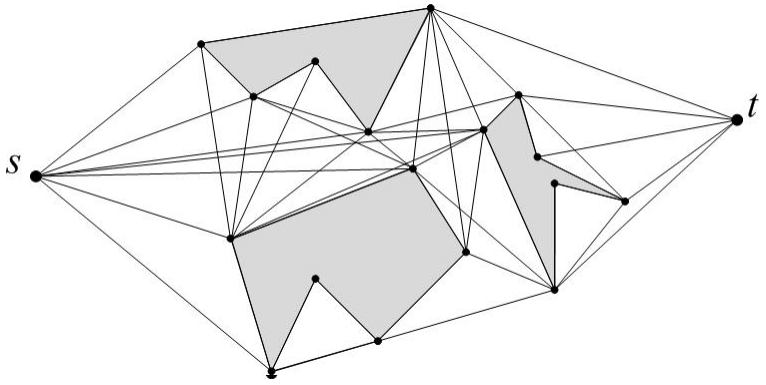

Figure 2.15: Construction of a visibility graph.

size shortest path map directly [46, 47]. While this method was originally developed for the Euclidean shortest-path problems, applications to other scenarios have been also considered (e.g., L_1 shortest paths in the plane [43], and construction of Voronoi diagrams for convex distance functions [12]). We thus note that Algorithm 2 is not the only possible method to address our problem, and the extension of a continuous Dijkstra algorithm to an anisotropic medium with a convex LPAR is also a plausible approach.

2.3.2 Fastest Path for an Arbitrary Anisotropic Speed Function

Subsection 2.3.1 discusses a direct extension of the shortest-path visibility graph approach to the obstacle-avoiding fastest-path finding problems with convex LPAR. However, the proposed algorithm does not apply to a general speed function $V(\theta)$ in the case when the correspond $L_\delta(x)$ is not convex. In Theorem 2.3 we have shown that a straight line path between a pair of points is not necessarily optimal for an arbitrary direction-dependent speed function. Therefore, in general, the triangle inequality does not hold true for the travel time function $\tau(.)$, and we cannot restrict our fastest-path search to the set of taut strings connecting s and t. In this subsection, we relax the convexity assumption of an LPAR, and analyze fastest-path finding problems for a general anisotropic speed function.

Consider an arbitrary speed function $V(\theta)$ and the corresponding linear path attainable region $L_\delta(x)$ which may or may not be convex. We introduce an augmented speed function $V'(\theta)$, such that, its corresponding LPAR, $L'_\delta(x)$, is the convex hull of $L_\delta(x)$, i.e., $L'_\delta(x) := conv(L_\delta(x))$ (see Figure 2.16). Note that the set $L'_\delta(x)$ and the speed function $V'(\theta)$ are unique, due to the uniqueness of a convex hull. By definition, a linear path attainable region $L'_\delta(x)$ is convex. Therefore, by constructing the visibility graph $\mathcal{VG}_{V'}$ as described in Algorithm 2, we can find an obstacle-avoiding fastest path from s to t corresponding to the new speed function $V'(\theta)$. We let $p_{V'}$ represent this optimal path and $t_{V'}(p_{V'})$ denote the travel time along the path $p_{V'}$ while traveling with speed $V'(\theta)$. Then, Proposition 2.17 below states that the minimum travel time from s to t corresponding to the original speed function $V(\theta)$ has to be greater than or equal to $t_{V'}(p_{V'})$.

Proposition 2.17. *The traveling time along a fastest path corresponding to an ar-*

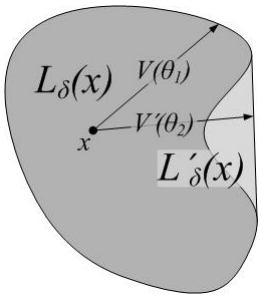

Figure 2.16: Example of $L_\delta(x)$ and $L'_\delta(x) := conv(L_\delta(x))$.

bitrary speed function $V(\theta)$ has to be greater than or equal to the travel time along a fastest path corresponding to the speed function $V'(\theta)$, where $L'_\delta(x) = conv(L_\delta(x))$.

Proof. [by contradiction] Let p_V denote a fastest path from s to t corresponding to a speed function $V(\theta)$, and $t_V(p_V)$ be the path travel time at speed $V(\theta)$. Assume that $t_V(p_V) < t_{V'}(p_{V'})$. From Lemma 2.5 we know that since $L_\delta(x) \subseteq L'_\delta(x)$, then $V(\theta) \leq V'(\theta), \forall\theta$. Consequently, traveling along the path p_V with the speed described by function $V'(\theta)$ (denoted by $t_{V'}(p_V)$), constrains the travel time to be less than or equal to $t_V(p_V)$. Hence, we find a feasible path corresponding to the speed function $V'(\theta)$ with the travel time less than or equal to $t_V(p_V)$. Then, the travel time along an optimal path corresponding to the speed function $V'(\theta)$ will be less than or equal to the travel time along this feasible path. That is $t_{V'}(p_{V'}) \leq t_{V'}(p_V) \leq t_V(p_V)$, implying that assumption $t_V(p_V) < t_{V'}(p_{V'})$ is contradictory, and thus proving the proposition. □

Proposition 2.17 concludes that if we let p_V denote an obstacle-avoiding fastest path for speed $V(\theta)$, then $t_V(p_V) \geq t_{V'}(p_{V'})$. This provides a lower bound on the minimum travel time for our original problem, which is an important component to characterizing an optimal path. We use this bound to demonstrate that the travel time for a path proposed below is equal to its lower bound, consequently proving the

path's optimality.

Since path $p_{V'}$ lies in the visibility graph $\mathcal{VG}_{V'}$, it is piecewise-linear with the waypoints corresponding to the vertices of $\mathcal{P}$, and points s and t. Consequently, the total travel time of the path can be written as the sum of travel times along each individual link. Recall that the travel time for each linear link (i, j) of the path $p_{V'}$ is equal to $\tau'(i, j) = \frac{\|j-i\|}{V'(\theta_{j-i})}$ (see Figure 2.17). From Theorem 2.10, our obstacle-free analysis describes a fastest path from i to j for an arbitrary speed function $V(\theta)$ with the optimal travel time equal to $\tau'(i, j)$. Applying the theorem to each linear link of the path $p_{V'}$ and then combining them together results in a path corresponding to the original speed function $V(\theta)$ with the travel time equal to the lower bound $t_{V'}(p_{V'})$.

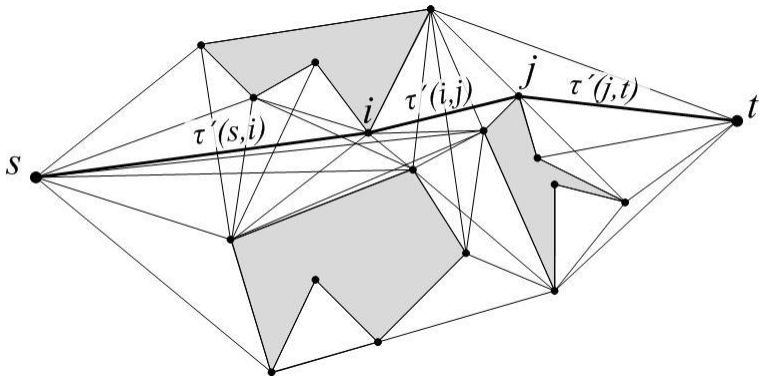

Figure 2.17: Fastest path corresponding to the speed function $V'(\theta)$ is shown in bold. Its travel time $t_{V'}(p_{V'}) = \tau'(s, i) + \tau'(i, j) + \tau'(j, t)$.

Recall that an optimal path described in the second scenario of Theorem 2.10 is not unique; it is just one of infinitely many paths with the same minimum travel time. In fact, as we attempt to implement the one-waypoint path along each linear link of the path $p_{V'}$ we might intersect the obstacle space $\mathcal{P}$. However, due to time and space homogeneity of the cost function, i.e., $V(\theta)$, we can construct a zigzag path with the same travel time by alternating the traveling directions between headings

corresponding to vectors $x_1 - x$ and $x_2 - x$ as many times as needed. Our problem statement assumes that the distance between any two obstacles is always greater than zero. Therefore, we can always construct a zigzag path close enough to the line xy, such that it does not intersect with the neighboring obstacles. (See Figure 2.18.)

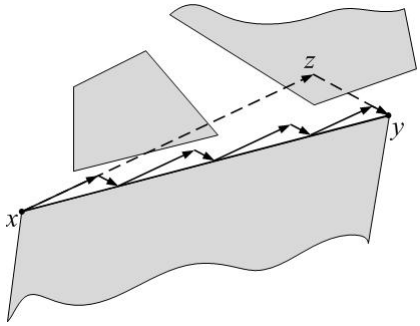

Figure 2.18: A feasible zigzag path from x to y with the total travel time equal to $\tau'(x, y)$.

We now introduce an algorithm for finding an obstacle-avoiding fastest path for an arbitrary speed function.

Algorithm 3 Obstacle-Avoiding Fastest Path for an Arbitrary Speed Function $V(\theta)$.

Step 1. Find $V'(\theta)$ for $\theta \in [0, 2\pi]$ such that $L'_\delta(x) = conv(L_\delta(x))$.

Step 2. Use Algorithm 2 to find an optimal path corresponding to the speed function $V'(\theta)$. Let $p_{V'}$ denote the determined path, and let $(k_0, k_1, k_2, ..., k_n)$ be the sequence of vertices path $p_{V'}$ is traversing. Note that $k_0 = s$ and $k_n = t$. Then the corresponding travel time along the path $p_{V'}$, denoted by $t_{V'}(p_{V'})$, can we written as

$$t_{V'}(p_{V'}) = \Sigma_{i=1}^{n} \tau'(k_{i-1}, k_i) \tag{2.7}$$

Step 3. For each pair of consecutive points in $(k_0, k_1, ..., k_n)$, apply Algorithm 1 to find a fastest path between the two points corresponding to the speed function $V(\theta)$. If a given one waypoint path is infeasible due to the presence of obstacles, increase the number of waypoints in a zigzag path as discussed above.

Step 4. Combine together the optimal paths found in Step 3. The resulting path has a travel time equal to $t_{V'}(p_{V'})$ and is therefore a fastest obstacle-avoiding path for an arbitrary speed function $V(\theta)$. (See Figure 2.19.)

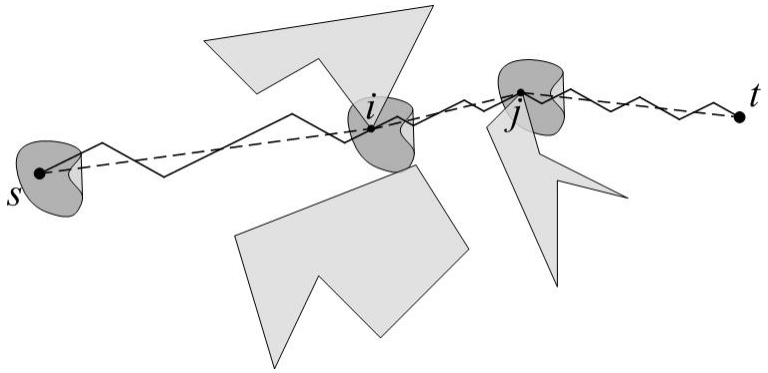

Figure 2.19: Example of a fastest path for speed function $V'(\theta)$ (dashed line), and an optimal path for speed $V(\theta)$ (solid line).

2.4 Applications and Conclusion

2.4.1 Applications

A fastest-path finding problem for the direction-dependent speed functions arises in a wide range of applications. For example, the speed of a sail boat depends on the traveling heading angle it makes with wind, and a vehicle speed varies as the agent traverses up and down a hill. Airplanes have to deal with an anisotropic speed due to wind, while motor boats have similar effects caused by waves. The two areas of application that we analyze in more details are optimal short-range routing of vessels in a seaway, and the optimal path finding for Tactical Mini-Unmanned Aerial Vehicles (TACMAVs), also referred to as mini Unmanned Aerial Vehicles (mini UAVs) or miniature drones.

Optimal Short-Range Routing of Vessels in a Seaway.

Any vessel traveling at a seaway encounters waves which add drag and affect the vessel's performance. In our collaboration with colleagues working on Optimal Vessel Performance in Evolving Nonlinear Wave-Fields project [20], we evaluate the added drag by computing the time average wave force acting on the vessel in the longitudinal direction. Then, by superimposing the added drag on the steady drag

experienced by the moving ship in calm waters, we compute the maximum mean attainable speed for each given sea state (which describes the distribution of the waves) and the heading angles in the range from 0° to 180°. Figure 2.20, borrowed from [20], illustrates an example of the linear path attainable region for the S-175 containership at Sea State no.7. Here, heading is measured as the angle a vessel makes with the dominant wave direction, which is assumed to be in the southerly direction.

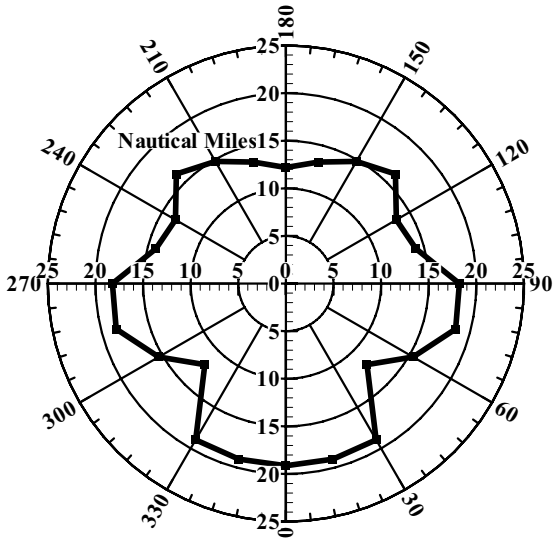

Figure 2.20: "An example of linear path attainable regions for the S-175 corresponding to voluntary speed loss at Sea State no.7" [20].

For the given LPAR, we can use Theorem 2.10 to find a fastest path; Algorithm 1 describes the step-by-step procedure to construct such an optimal path. As an example, we consider two scenarios. In first case, let the target point t_1 lie directly east from the starting point s. This example corresponds to the scenario 1 of Theorem 2.10, since the straight line st_1 intersects the boundary of the linear path attainable region $L_\delta(s)$ and the boundary of its convex hull at the same point. Hence, we can conclude that the straight line path st_1 is a fastest path from s to t_1, illustrated in

Figure 2.21.

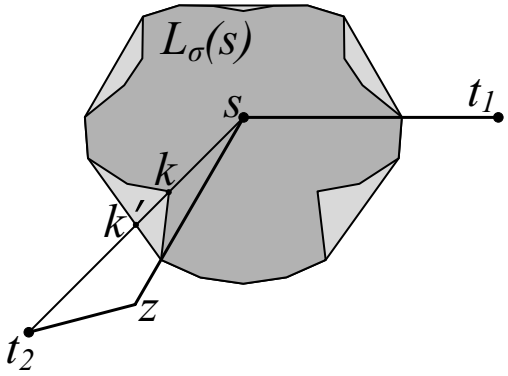

Figure 2.21: Illustration of the fastest paths from point s to points t_1 and t_2, paths st_1 and szt_2, respectively.

In the second example, let the target point t_2 lie south-west from the starting point s. Then, the intersection points of the line st_2 with the boundary of $L_\delta(s)$ and the boundary of $L_\delta(s)$'s convex hull are not the same (i.e., $k \neq k'$), corresponding to the scenario 2 of Theorem 2.10. From the theorem we can conclude that a fastest path from s to t_2 is piecewise-linear with one waypoint. Thus, to reach the point t_2 as fast as possible, the vessel should first travel SSE, or 30° clockwise from the south direction, and then complete the travel heading 75° clockwise from south. This corresponds to the path szt_2, illustrated on the Figure 2.21.

In addition to finding a fastest path from s to t_2, we can use equations (2.2) to calculate how much improvement in travel time a vessel observes as it follows the optimal path szt_2 instead of following a straight line path st_2. By dividing the length of sk by the length of sk', we find that $\beta = 0.688$, which implies that by following an optimal path we can decrease our travel time at most by approximately 31.2%. This kind of information is particulary useful in evaluating the tradeoffs between following an optimal path as opposed to following a straight line.

In some applications seaway regions might be restricted for vessel's use due to

severe weather, presence of land, other vessels, or imposed regulations. For such problems we approximate the restricted regions with polygonal obstacles and apply Algorithm 3 to find an optimal obstacle-avoiding path to the destination.

Optimal Path Finding for Tactical Mini-Unmanned Aerial Vehicles (TACMAVs).

Unmanned Aerial Vehicles (UAVs) are extensively used in a wide range of military and civilian applications. While UAVs significantly vary in size and utilization, we direct our attention to mini-UAVs that are primarily used for surveillance and intelligence gathering. These mini-drones are called Tactical Mini-Unmanned Aerial Vehicles (TACMAVs) and can be seen in Figure 2.22 borrowed from [4]. TACMAV system includes a control unit and communication equipment that allows a person on the ground to choose points of interest for the miniature drone to visit. Two cameras mounted on a TACMAV capture the live video of the targets as the unmanned aerial vehicle flies over them. An onboard autopilot program navigates the mini-UAV between the points of interest.

With the wing span varying between 21 and 26 inches, and a total weight of only 0.8 lbs. to 1.6 lbs. [4, 70], TACMAVs are very susceptible to wind, which has a significant nonlinear effect on the guidance algorithms, and therefore must be accounted for in optimal path planning [52]. Since the TACMAV battery can only endure a flight time of 70 to 90 minutes, a fastest-path finding algorithm is essential for the best use of the mini-drone.

The TACMAN limited range of flight supports the assumption that the wind distribution is stationary during its flight. This provides time and space homogeneity assumed in this chapter. Calculating the wind effects on the mini-drone speed results in a direction-dependent speed function, $V(\theta)$. Once the speed function is known, we construct the corresponding linear path attainable region, apply Theorem 2.10,

Figure 2.22: Tactical Mini-Unmanned Aerial Vehicle (TACMAV) from Applied Research Associates, Inc. [4]

and then use Algorithm 1 to find a fastest path.

In some applications the agent's speed is maintained constant by utilizing a greater amount of fuel or battery charge. Our solution approach easily extends to such problems. We redefine the linear path attainable region to represent the set of points one can reach consuming a single unit of fuel or energy. *Then, by using the algorithms presented in this chapter we can find a path minimizing fuel consumption instead of traveling time.* Furthermore, a set of obstacles can be used to model UAVs restrictions such as enemy detection regions or the presence of buildings and other physical barriers.

2.4.2 Conclusion

In this chapter, we find the solution to a fastest-path finding problem for a direction-dependent time and space homogeneous speed function. We demonstrate that in an obstacle-free domain an optimal path is piecewise-linear with at most two line segments, regardless of the underline structure of the speed function. This analytical character of our results provides a computationally fast method for finding an optimal path, making it suitable for online applications. We also provide a tight

bound on the improvement in travel time by following an optimal path as opposed to traversing a simpler straight line path. Algorithm 1 presented in the chapter facilitates a simple implementation of these results.

We also use these results to address the obstacle-avoiding fastest-path problems in the anisotropic media. We use the properties of speed functions with the convex polar plots to adapt the visibility graph search method, traditionally used for Euclidean shortest-path problems, to find a solution for this type of problems. Algorithm 2 outlines the fastest-path finding procedure for solving the problems with the convex speed polar plots. We then address the case of an arbitrary speed function, which may not correspond to a convex liner path attainable region. For the general scenario, we introduce an augmented speed function such that its polar plot is the convex hull of the original speed plot. Then, to find a lower bound on the minimum travel time for our original problem we apply Algorithm 2 to the augmented speed function. By applying a fastest piecewise-linear path between the nodes of the visibility graph, we constructe a path with the travel time equal to its lower bound, thus establishing its optimality. Algorithm 3 gives the detailed steps to finding an optimal obstacle-avoiding path for a general time and space homogeneous speed function.

CHAPTER III

Optimal Path with Bounded Curvature in an Anisotropic Medium

3.1 Introduction

Chapter II showed that an optimal path in an anisotropic time and space homogeneous environment has a piecewise-linear structure. Unfortunately, the instantaneous heading change required to follow a piecewise-linear path is infeasible for most applications, at least on a small scale, including the navigation of aerial, surface and naval vehicles. For these problems, the control system of an agent constrains the set of feasible paths and these restrictions must be integrated into in the optimal path finding process. In this chapter, we introduce a direction-dependent minimum turning radius function that constrains the curvature of a feasible path to find an optimal path with bounded curvature in an anisotropic time and space homogeneous environment.

The problem's objective is to find a fastest path that starts at the starting point and heading angle, ends at a destination point with a predetermined final heading, and has a curvature bounded by a specified minimum turning radius function. Most published work that discusses fastest-path finding problems with bounded curvature assumes constant speed and minimum turning radius. We analyze the problems in the anisotropic media where both the agent's speed and minimum tuning radius

are described by the direction-dependent functions. The anisotropic nature of this problem implies the asymmetry of a travel time function as discussed in Chapter II. Additionally, the non-constant turning radius results in complex sharpest turn curves, as opposed to a circle, which is an essential part of an optimal path for the isotropic problems. These facts make the task of extending the problem of optimal path finding with minimum curvature to the anisotropic case a significant challenge.

3.1.1 Related Work

The problem of finding a fastest path with bounded curvature was first introduced by L. E. Dubins in 1957 [22]. Since then, it has been referred to as the Dubins car problem. Unlike our problem of interest, Dubins assumes that the vehicle speed and the minimum turning radius are constant and not direction-dependent. His intricate set of geometric statements and propositions show that "an R-geodesic is necessarily a continuous differentiable curve which consists of not more than three pieces, each of which is either a straight line segment or an arc of a circle of radius R" [22]. A simplified proof is later developed by two independent research teams, Boissonnat *et al.* [7] and Sussmann and Tang [69], and it is based on techniques of modern control theory and the minimum principle of Pontryagin [59]. Fortune's analysis of the problem [27] establishes the controllability of Dubins car by proving that a feasible path exists for any starting and final settings of the problem.

The optimal control theory based approach to Dubins car problem has generated a great interest in the field of robotics, and numerous variations and extensions of the problem are discussed in the literature. Reeds and Shepp [60] study optimal paths with bounded curvature for a car that can move forward and backwards, resulting in paths with cusps. Boissonnat *et al.* [6] propose a dynamic extension of Dubins car problem by constraining the angular acceleration of the agent instead

of its angular velocity. Some researchers consider problems with a more complex mobile robot configuration such as trailer-truck systems navigation [65]. Chitsaz and LaValle [13], on the other hand, extend Dubins car to having altitude, leading to problems studying a time-optimal trajectory for airplanes. Despite a wide variety of the aforementioned extensions, the assumption of constant speed and minimum turning radius (or angular acceleration in the case of [6]) restricts the analysis to the isotropic case.

Some researchers address the direction-dependent problems while considering specific applications. Unmanned aerial vehicle (UAV) routing is the predominant area of such applications, where researchers conduct an analysis of the effects of directional wind on the UAV optimal paths. In the majority of published work, the aircraft velocity and wind velocity are assumed to be constant, and the actual speed of the vehicle is equal to the sum of the two vectors [52, 40]. This assumption imposes a very specific structure on the actual direction-dependent speed function and the minimum turning radius of an agent; we make no such restrictions in our forthcoming analysis.

McNeely *et al.* [41] consider the problem of constructing a minimum-time trajectory for UAVs in the presence of a time-dependent wind vector field. Analogous to work discussed above, the realized speed of a vehicle is computed by adding the wind vector field to the constant velocity of an agent. To solve their problem, McNeely and her colleagues transform space by the wind vector, implement Dubins car solution in the new space and iteratively apply Newton's method to converge to the correct destination (boundary condition). The authors prove the existence and uniqueness of the optimal solution, along with the convergence of the algorithm. While we consider a time homogeneous environment, our work presented in this chapter explicitly

characterizes an optimal path and facilitates a straightforward construction of such a path.

3.1.2 Overview of the Results

In the current chapter, we deliver an analytical characterization of a fastest path with bounded curvature in an anisotropic medium. We restrict our analysis to the set of direction-dependent speed functions corresponding to a convex linear path attainable region (i.e., convex speed polar plot) and an arbitrary direction-dependent minimum turning radius. We employ the methodology from optimal control theory (Pontryagin's Principle) and derive a necessary condition for path optimality. The subsequent analysis invokes the properties derived in Chapter II and further investigates the structure of an optimal path, resulting in a closed form characterization of such path.

We show that the structure of an optimal path in an anisotropic medium is similar to the solution of a Dubins car problem and consists of either three sharpest turn arcs with an alternating direction, or a sharpest turn curve followed by a straight line segment and concluded by a second sharpest turn arc. It is important to note that while the characterization of the optimal paths for two classes of problems is similar, the sharpest turn curves of our problem have a very general, and often complex, structure. An algorithm presented at the end of the chapter provides a detailed set of steps that delivers an optimal path in an anisotropic medium.

The rest of the chapter is arranged as follows. Section 3.1.3 provides the precise problem statement and includes a list of technical assumptions necessary to ensure the rigorous analysis throughout the chapter. Section 3.2 discusses the application of techniques from optimal control theory in order to establish the problem's controllability (reduction to Dubins car problem), prove existence of an optimal path

(Filippov's Theorem) and derive the necessary condition for optimality (Pontryagin's Principle). Section 3.3 assumes a convex linear path attainable region and employs the optimality condition and further analysis of the problem structure to deliver an optimal path. This section also contains Algorithm 4, which facilitates the implementation of our results and delivers an explicit path finding procedure. Finally, Section 3.4 concludes this chapter with the summary of our results and the discussion of future work related to this problem.

3.1.3 Problem Statement

Given a starting point $s = (x_s, y_s) \in \mathbb{R}^2$ and a target point $t = (x_t, y_t) \in \mathbb{R}^2$, find a fastest path such that a mobile agent departs the starting point at time $t_0 = 0$ at a heading angle $\theta_s \in [0, 2\pi]$ and arrives at the target point with a heading direction $\theta_t \in [0, 2\pi]$. The path curvature is restricted by a minimum turning radius function $R(\theta) : [0, 2\pi] \to \mathbb{R}^+$ dependent on the vehicle traveling direction. Anisotropic speed function $V(\theta) : [0, 2\pi] \to \mathbb{R}^+$ denotes the maximum attainable speed the vehicle can achieve for each heading angle.

System assumptions.

We address a general fastest-path finding problem in a direction-dependent environment and do not restrict our analysis to any particular application or specific speed and minimum turning radius functions. To ensure the completeness and rigor of the presented analysis, we next state the assumptions necessary for our discussion. However, it is important to note that most of the listed assumptions are relatively unrestrictive and valid for all practical applications.

Assumption 3.1. *An agent's speed is always equal to its maximum attainable speed* $V(\theta)$.

The speed of a mobile agent is a set parameter for any given instance of the problem. Intuitively, it appears that in a time-homogeneous medium it is never advantageous to voluntarily decrease the speed, questioning the need for this assumption. However, it might be beneficial to slow down if a lower speed permits sharper turns and allows paths with shorter distance. In that scenario, the minimum turning radius function R would explicitly depend on the vehicle speed, in addition to the heading angle. If a mobile agent had an option to change its speed along a path, we could add the speed control variable to the system decision space. The additional controller significantly increases the complexity of a model, and we do not consider such problems in the presented analysis. However, the relaxation of Assumption 3.1 is an interesting extension to be considered in our future work.

Assumption 3.2. *The minimum turning radius function can only take on positive values, i.e.,* $R(\theta) > 0, \forall\theta$.

Assumption 3.3. *The speed function can only take on positive values, i.e.,* $V(\theta) > 0, \forall\theta$.

Assumption 3.4. *The minimum turning radius function* $R(\theta)$ *is bounded, that is,* $\exists R_{\max}$, *such that* $R(\theta) \leq R_{\max},\ \forall\theta \in [0, 2\pi]$.

Assumption 3.5. *The speed function* $V(\theta)$ *is bounded, that is,* $\exists V_{\max}$, *such that* $V(\theta) \leq V_{\max}, \forall\theta \in [0, 2\pi]$.

Assumption 3.6. *The minimum turning radius,* $R(\theta)$, *is a* $\mathcal{C}^1$ *function, that is,* $\frac{dR(\theta)}{d\theta}$ *is continuous.*

Assumption 3.7. *The speed function,* $V(\theta)$, *is a* $\mathcal{C}^1$ *function, that is,* $\frac{dV(\theta)}{d\theta}$ *is continuous.*

3.2 Optimal Control Modeling and Analysis of the Problem

In this section, we apply the optimal control theory techniques to demonstrate the problem's controllability, to prove the existence of an optimal path, and to derive the system's necessary condition for optimality. For completeness, we dedicate Subsection 3.2.1 to describing Pontryagin's Principle of Optimality, which is one of the main techniques used in our analysis. In Subsection 3.2.2, we describe the detailed application of Pontryagin's Principle and other optimal control methodology to our problem.

3.2.1 Overview of Pontryagin's Minimum Principle

The content of this section is based on *The Mathematical Theory of Optimal Processes* by Pontryagin *et al.* [59]. In their work, Pontryagin and his colleagues introduce the key theorem as the *maximum* principle. However, most of the control literature following the publication of Pontryagin's book defines the Hamiltonian to be the negative of the 'Pontryagin's Hamiltonian', consequently transforming the principle from 'maximum' to 'minimum'. In the following discussion, we choose to adapt Pontryagin's original claim to the *minimum* principle in order to conform to a larger part of published work in the field of optimal control.

Optimal Control Problem

Consider a control process where $x(t) = (x_1(t), ..., x_n(t)) \in \mathbb{R}^n$ for some $t \in [t_0, t_f]$ is the vector characterizing the process (state of the system) at time t, and $u(t) = (u_1(t), ..., u_r(t)) \in U$ is the control vector function (given for $t_0 \leq t \leq t_f$) which determines the course of the process. The range of the control function is given by a *control region* $U \subseteq \mathbb{R}^r$, and $x_i(t_0) = x_i^0,\ i = 1, ..., n$ denote the initial state of the system. We analyze a control process described by a system of ordinary differential

equations:

$$\dot{x}_i(t) = \frac{dx_i(t)}{dt} = f_i(x_1(t), ..., x_n(t), u_1(t), ..., u_r(t)),\ i = 1, 2, ..., n. \tag{3.1}$$

Note that the system (3.1) is *autonomous*, i.e., its right-hand side does not depend explicitly on the time t. Pontryagin *et al.* also discuss the non-autonomous systems in their book; however, they are not relevant to our problem.

Our objective is to find a piecewise continuous $u(t)$ that minimizes the functional J, as defined below, such that $x(t_0) = x^0$ and $x(t_f) = x^f$.

$$J = \int_{t_0}^{t_f} f_0(x_1(t), ..., x_n(t), u_1(t), ..., u_r(t))dt. \tag{3.2}$$

Here, $f_0(x(t), u(t))$ is a given function, and t_f is not fixed (i.e., it is a free parameter).

It is important to note that the discussion and results presented in this section are limited to the scenarios when the following assumptions hold true.

1. The control region U is assumed to be a closed set.

2. The control function $u(t)$ is piecewise continuous. Furthermore, we assume that $u(t)$ is either right- or left-continuous at each point of discontinuity, leading to the fact that every control $u(t)$ is bounded even if U is not.

3. The state coordinates $(x_1(t), ..., x_n(t))$ are continuous and piecewise differentiable.

4. The functions $f_i(x, u)$ are assumed to be continuous in $x_1, ..., x_n, u_1, ..., u_r$, and continuously differentiable with respect to $x_1, ..., x_n$.

Adjoin a new coordinate $x_0(t)$ to the state coordinates $x_1(t), ..., x_n(t)$, and let $x_0(t)$ vary according to the law

$$\dot{x}_0(t) = \frac{dx_0(t)}{dt} = f_0(x(t), u(t)), \tag{3.3}$$

where right-hand side does not depend on $x_0(t)$. Then, $x_0(t_0) = 0$ and $x_0(t_f) = J$. For convenience, introduce a new notation $\mathbf{x}(t) = (x_0(t), x(t)) = (x_0(t), x_1(t), ..., x_n(t))$, not to be confused with $x(t) = (x_1(t), ..., x_n(t))$.

The Minimum Principle

In addition to the fundamental system of equations:

$$\dot{x}_i(t) = f_i(x(t), u(t)),\ i = 0, 1, 2, ..., n, \tag{3.4}$$

consider another system of equations in the auxiliary (adjoint) variables $\psi_0, \psi_1, ..., \psi_n$ defined by:

$$\dot{\psi}_i(t) = -\sum_{\alpha=0}^{n} \frac{\partial f_\alpha(x(t), u(t))}{\partial x_i} \psi_\alpha(t),\ i = 0, 1, ..., n. \tag{3.5}$$

It is important to note that if we choose an admissible control $u(t),\ t_0 \leq t \leq t_f$, and have the corresponding state trajectory $\mathbf{x}(t)$ of the system (3.4) with the initial condition $\mathbf{x}(t_0) = \mathbf{x}^0$, then for any initial conditions $\boldsymbol{\psi}(t_0)$, system (3.5) has a unique solution $\boldsymbol{\psi}(t) = (\psi_0(t), \psi_1(t), ..., \psi_n(t)),\ t_0 \leq t \leq t_f$.

Also note that the vector function $\boldsymbol{\psi}(t)$ is continuous and has everywhere continuous derivative with respect to t, except at a finite number of points corresponding to the points of discontinuity of $u(t)$.

Now, we combine $\mathbf{x}$ and $\boldsymbol{\psi}$ to define the 'Hamiltonian' as

$$H(\boldsymbol{\psi}, x, u) = \langle \boldsymbol{\psi}, \mathbf{f}(x, u) \rangle = \sum_{\alpha=0}^{n} \psi_\alpha f_\alpha(x, u). \tag{3.6}$$

Then the above systems (3.4) and (3.5) can be written as

$$\dot{x}_i = \frac{\partial H(\boldsymbol{\psi}, x, u)}{\partial \psi_i},\ i = 0, 1, ..., n \tag{3.7}$$

$$\dot{\psi}_i = -\frac{\partial H(\boldsymbol{\psi}, x, u)}{\partial x_i},\ i = 0, 1, ..., n \tag{3.8}$$

Treating $\boldsymbol{\psi}(t)$ and $x(t)$ as fixed parameters at each time t, the function H becomes a function of the parameter $u \in U$. Let us define the minimum of H as

$$M(\boldsymbol{\psi}, x) = \min_{u \in U} H(\boldsymbol{\psi}, x, u). \tag{3.9}$$

Now, we are ready to state Pontryagin's Minimum Principle that provides a *necessary* condition for control function $u(t)$ to be optimal.

Theorem 3.8 (adapted from [59]). *Let $u(t),\ t_0 \leq t \leq t_1$ be an admissible control and $\mathbf{x}(t)$ the corresponding trajectory that begins at the point $\mathbf{x}^0$ at the time t_0. In order that $u(t)$ and $\mathbf{x}(t)$ be optimal, it is necessary that there exist a nonzero continuous vector function $\boldsymbol{\psi}(t) = (\psi_0(t), \psi_1(t), ..., \psi_n(t))$ corresponding to $u(t)$ and $\mathbf{x}(t)$ (i.e., the equations (3.7) and (3.8) are satisfied), such that*

1. *$\forall t,\ t_0 \leq t \leq t_1$, the function $H(\boldsymbol{\psi}(t), x(t), u)$ of the variable $u \in U$ attains its minimum at the point $u = u(t)$. That is,*

$$H(\boldsymbol{\psi}(t), x(t), u(t)) = M(\boldsymbol{\psi}(t), x(t)); \tag{3.10}$$

2. *$\forall t,\ t_0 \leq t \leq t_1,\ M(\boldsymbol{\psi}(t), x(t)) = 0$ and $\psi_0(t) \equiv \psi_0(t_0) \geq 0$.*

In another words, Pontryagin and his colleagues show that the optimal control function must minimize $H(\boldsymbol{\psi}(t), x(t), u(t))$, and furthermore, the minimum M must be equal to zero (since t_f is a free parameter), and the function $\psi_0(t)$ must be a non-negative constant for all $t \in [t_0, t_1]$.

3.2.2 Application of Pontryagin's Minimum Principle

In this section we apply Pontryagin's minimum principle and other methods of the optimal control theory to the problem introduced in Section 3.1.3.

Control Model

Let $\alpha(t)$ denote the polar angle of the tangent to the path, or the heading angle at time t. We define the state of our system to be $(x(t), y(t), \alpha(t))$ for $t \in [0, T]$, where $(x(t), y(t))$ is the position of a vehicle in $\mathbb{R}^2$ at time t (see Figure 3.1). We set the system steering controller $u(t)$ for $t \in [0, T]$ to represent the change in the vehicle heading at time t.

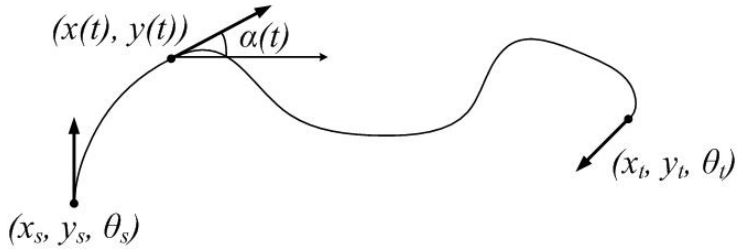

Figure 3.1: The state of the system is $(x(t), y(t), \alpha(t))$, where $(x(t), y(t))$ is the position of a vehicle in $\mathbb{R}^2$ and $\alpha(t)$ is its heading angle at time t.

The objective of our problem is to minimize the total travel time denoted by J_u, which is defined as:

$$J_u = \int_0^T f_0(x(t), y(t), \alpha(t), u(t))dt = \int_0^T \frac{\sqrt{\dot{x}(t)^2 + \dot{y}(t)^2}}{V(\alpha(t))}dt \tag{3.11}$$

To apply Pontryagin's minimum principle, we let $x_0(t)$ denote the travel cost accumulated by time t. Then the objective of our problem can be written as:

$$x_0(T) = J_u = \int_0^T \frac{\sqrt{\dot{x}(t)^2 + \dot{y}(t)^2}}{V(\alpha(t))}dt \tag{3.12}$$

The differential system describing the dynamics of our system is

$$\dot{x}(t) = f_1(x(t), y(t), \alpha(t), u(t)) = V(\alpha(t))\cos(\alpha(t)), \tag{3.13}$$

$$\dot{y}(t) = f_2(x(t), y(t), \alpha(t), u(t)) = V(\alpha(t))\sin(\alpha(t)), \tag{3.14}$$

$$\dot{\alpha}(t) = f_3(x(t), y(t), \alpha(t), u(t)) = \frac{V(\alpha(t))}{R(\alpha(t))}u(t), \tag{3.15}$$

$$\dot{x}_0(t) = f_0(x(t), y(t), \alpha(t), u(t)) = \frac{\sqrt{\dot{x}(t)^2 + \dot{y}(t)^2}}{V(\alpha(t))} = 1, \tag{3.16}$$

with the boundary conditions:

$$(x(0), y(0), \alpha(0)) = (x_s, y_s, \theta_s), \tag{3.17}$$

$$(x(T), y(T), \alpha(T)) = (x_t, y_t, \theta_t), \tag{3.18}$$

$$x_0(0) = 0. \tag{3.19}$$

Then, we can simplify equation (3.11) as

$$J_u = \int_0^T \frac{\sqrt{\dot{x}(t)^2 + \dot{y}(t)^2}}{V(\alpha(t))} dt = \int_0^T dt = T \tag{3.20}$$

Note that while the minimum turning radius $R(\alpha(t))$ depends on the heading along the path, the control region U (i.e., all admissible controllers $u(t) \in U$) cannot be dependent on the state of the system. To accommodate this restriction, we set $U = [-1, 1]$ and scale the path curvature by the residual of a minimum turning radius, $\frac{1}{R(\alpha(t))}$, as can be seen in Equation (3.15). (Assumption 3.2 guarantees that we never divide by zero.)

System Controllability

The first step of finding an optimal path is to prove that the problem is *controllable*, that there exists a feasible path for all possible starting and target states of the system. Dubins car problem of finding a fastest path with bounded curvature in the case of a constant speed has been shown to be controllable for a constant minimum turning radius, $R_D(\theta) = r$ for all $\theta \in [0, 2\pi]$ (see [27, 35]). Assumption 3.4 states that the minimum turning radius function is bounded above which ensures that there exists a minimum turning radius value large enough to be feasible for all headings, i.e., let $R_{\max} = \sup_\theta R(\theta)$. Then, a problem of fastest-path finding with curvature restricted by a minimum turning radius equal to $R_{\max}$ for all headings is reduced to Dubins car problem which we know to be controllable. Consequently, we can always

find a feasible path from an arbitrary point (x_s, y_s) and the starting heading θ_s to any destination point and heading angle (x_t, y_t, θ_t), thus implying the controllability of our problem.

Existence of an Optimal Path

To prove the existence of an optimal path, we apply a variation of Filippov's theorem [26] for minimum-time problems as stated by Souères and Boissonnat in [65]. For completeness, we state the theorem here with a slight modification to match our notation.

Let M be an open subset of $\mathbb{R}^3$ and U a subset of $\mathbb{R}$. Then, when $f := (f_1, f_2, f_3)$ is a linear function of the control parameter u of the form:

$$f(x, y, \alpha, u) = g_1(x, y, \alpha) + g_2(x, y, \alpha)u, \tag{3.21}$$

we have the following version of Filippov's theorem.

Theorem 3.9 (adapted from [65]). *Let (x_s, y_s, θ_s) and (x_t, y_t, θ_t) be two points in M. Under the following hypotheses there exists an optimal trajectory solution of (3.21) linking (x_s, y_s, θ_s) and (x_t, y_t, θ_t).*

H_1 : *The g_1 and g_2 are locally lipschitzian functions of x, y and α.*

H_2 : *The control set U is a compact convex subset of $\mathbb{R}$.*

H_3 : *There exists an admissible trajectory from (x_s, y_s, θ_s) to (x_t, y_t, θ_t).*

H_4 : *The system is complete, in the sense that given any point $(x_s, y_s, \theta_s) \in M$, and any admissible control law $u(t)$ defined for $t \in [0, T]$, there exists a corresponding trajectory $(x(t, u), y(t, u), \alpha(t, u))$ defined on the whole time interval $[0, T]$ and verifying $(x(0), y(0), \alpha(0)) = (x_s, y_s, \theta_s)$.*

To demonstrate that Theorem 3.9 applies to our problem, first rewrite f as a linear function of the control parameter u from equations (3.13)-(3.15).

$$f(x, y, \alpha, u) = \begin{pmatrix} V(\alpha)\cos(\alpha) \\ V(\alpha)\sin(\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \frac{V(\alpha)}{R(\alpha)} \end{pmatrix} u. \tag{3.22}$$

The four hypotheses of the theorem hold true:

H_1 : Recall that a function is called locally Lipschitz continuous if for any point on its domain there exists a neighborhood such that function restricted to this neighborhood is Lipschitz continuous. From that definition we can observe that any $\mathcal{C}^1$ (continuously differentiable) function is locally lipschitzian, as continuous function on a locally compact space is locally bounded.

From Assumptions 3.6 and 3.7, we know that $V(\alpha)\cos(\alpha)$ and $V(\alpha)\sin(\alpha)$ are $\mathcal{C}^1$ functions and consequently, are locally lipschitzian. The function $\frac{V(\alpha)}{R(\alpha)}$ is also locally Lipschitz continuous since its derivative, $\frac{dV(\alpha)/d\alpha}{R(\alpha)} - \frac{V(\alpha)dR(\alpha)/d\alpha}{R^2(\alpha)}$, is continuous. Thus, we can conclude the first hypothesis of Theorem 3.9.

H_2 : The control region of our problem is $U = [-1, 1]$, which is a compact subset of $\mathbb{R}$.

H_3 : System controllability demonstrated earlier in this section implies that there exists an admissible trajectory from (x_s, y_s, θ_s) to (x_t, y_t, θ_t).

H_4 : Since the domain of a path is not restricted by obstacles or other constraints, any admissible control would result in a feasible path, implying the completeness of the system.

As shown above, all four hypotheses of Theorem 3.9 hold true, therefore we can

conclude the existence of an optimal path for any starting and target states of the system.

It is important to note that Filippov's theorem proves the existence of an optimal path in a larger class of controls which are only assumed to be measurable. However, Pontryagin's principle also applies to this larger class (see Chapter II of [59]), and our further analysis proves that the optimal controls do belong to the smaller class of piecewise continuous functions. Consequently, we justify the discussion restricted to that class of controls, which improves the clarity and applicability of our work.

Necessary Conditions for Optimality: Pontryagin's Minimum Principle

Let $(\psi_0(t), \psi_1(t), \psi_2(t), \psi_3(t))$ be the adjoint variables corresponding to $(x_0(t), x(t), y(t), \alpha(t))$. We find the Hamiltonian $H(\psi_0(t), \psi_1(t), \psi_2(t), \psi_3(t), x_0(t), x(t), y(t), \alpha(t), u(t))$ as defined in equation (3.6). To simplify the notation we let $H(.)$ denote the Hamiltonian, implying the same set of dependent variables as above.

(3.23)
$$H(.) = \psi_0(t) + \psi_1(t)V(\alpha(t))\cos(\alpha(t)) + \psi_2(t)V(\alpha(t))\sin(\alpha(t)) + \psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))}u(t),$$

and the adjoint system found using equation (3.8) is

$$\dot{\psi_0}(t) = -\frac{\partial H(.)}{\partial x_0} = 0 \tag{3.24}$$

$$\dot{\psi_1}(t) = -\frac{\partial H(.)}{\partial x} = 0 \tag{3.25}$$

$$\dot{\psi_2}(t) = -\frac{\partial H(.)}{\partial y} = 0 \tag{3.26}$$

$$\begin{aligned} \dot{\psi_3}(t) &= -\frac{\partial H(.)}{\partial \alpha} = \\ &= \psi_1(t)V(\alpha(t))\sin(\alpha(t)) - \psi_1(t)V'(\alpha(t))\cos(\alpha(t)) - \\ &\quad -\psi_2(t)V(\alpha(t))\cos(\alpha(t)) - \psi_2(t)V'(\alpha(t))\sin(\alpha(t)) + \\ &\quad +\psi_3(t)\frac{V(\alpha(t))R'(\alpha(t))}{R(\alpha(t))^2}u(t) - \psi_3(t)\frac{V'(\alpha(t))}{R(\alpha(t))}u(t), \end{aligned} \tag{3.27}$$

where $V'(\alpha(t)) = \frac{dV(\alpha(t))}{d\alpha}$ and $R'(\alpha(t)) = \frac{dR(\alpha(t))}{d\alpha}$.

Then ψ_0, ψ_1 and ψ_2 are constants on $[0, T]$. To simplify the notation, let $\psi_1 = \lambda\cos(\phi)$ and $\psi_2 = \lambda\sin(\phi)$, where $\lambda = \sqrt{\psi_1^2 + \psi_2^2} \geq 0$ and $\phi < 2\pi$ such that $\tan(\phi) = \frac{\psi_2}{\psi_1}$ (see Figure 3.2 for an intuitive interpretation).

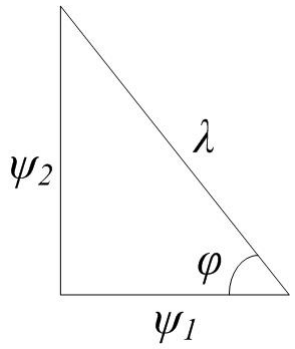

Figure 3.2: Relabeling ψ_1 and ψ_2 as $\lambda\cos(\phi)$ and $\lambda\sin(\phi)$, respectively.

We can then rewrite Hamiltonian and $\dot{\psi_3}$ as follows.

$$H(.) = \psi_0 + \lambda V(\alpha(t))\cos(\alpha(t) - \phi) + \psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))}u(t) \tag{3.28}$$

$$\begin{aligned}\dot{\psi_3}(t) = {} & \lambda V(\alpha(t))\sin(\alpha(t) - \phi) - \lambda V'(\alpha(t))\cos(\alpha(t) - \phi) + \\ & + \psi_3(t)\frac{V(\alpha(t))R'(\alpha(t))}{R(\alpha(t))^2}u(t) - \psi_3(t)\frac{V'(\alpha(t))}{R(\alpha(t))}u(t).\end{aligned} \tag{3.29}$$

From Pontryagin's Minimum Principle (Theorem 3.8) we know that if $u^*(t)$ is an optimal control function, then for all $t \in [0, T]$ we have

$$u^*(t) = \arg\min_{u\in[-1,1]}\{\psi_0 + \lambda V(\alpha(t))\cos(\alpha(t) - \phi) + \psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))}u\} \tag{3.30}$$

where $\psi_3(t)$ is defined by equation (3.29).

Furthermore, we know that

$$\psi_0 + \lambda V(\alpha(t))\cos(\alpha(t) - \phi) + \psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))}u^*(t) = 0, \forall t \tag{3.31}$$

and

$$\psi_0 \geq 0. \tag{3.32}$$

From Condition 1 of Theorem 3.8 we know that for an optimal control $u^*(t)$, along any $\mathcal{C}^2$ piece of the optimal path, we have

$$\psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))}u(t) \leq 0. \tag{3.33}$$

Assumptions 3.3 and 3.4 simplify Equation (3.33) as follows:

$$\psi_3(t)u(t) \leq 0. \tag{3.34}$$

Furthermore, along any $\mathcal{C}^2$ piece of an optimal path either one of the following two cases holds [7]:

1. $\frac{\partial H}{\partial u} \equiv 0, \forall t$, which implies $\psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))} \equiv 0$. Since $V(\theta) > 0, \forall\theta$ (Assumption 3.3) and $R(\theta)$ is bounded (Assumption 3.4), we know $\psi_3(t) = 0, \forall t$. Then $\dot{\psi}_3(t) \equiv 0$, implying $\lambda V(\alpha(t))\sin(\alpha(t)-\phi) - \lambda V'(\alpha(t))\cos(\alpha(t)-\phi) = 0$ (see equation (3.29)).

   If $\lambda \neq 0$, from calculus (e.g., [72] page 676) $V'(\alpha(t)) = V(\alpha(t))\cot(\varphi(\alpha(t)))$, where $\varphi(\alpha(t))$ is an angle between the tangent and radial lines of the $V(\alpha(t))$ polar plot. See Figure 3.3. Then we can solve the equation $\dot{\psi}_3(t) = 0$ as follows.

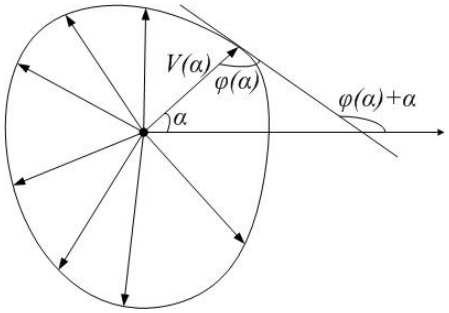

Figure 3.3: $\varphi(\alpha)$ denotes an angle between the tangent and radial lines of the $V(\alpha)$ polar plot.

$$
\begin{aligned}
\lambda V(\alpha(t))\sin(\alpha(t)-\phi) - \lambda V'(\alpha(t))\cos(\alpha(t)-\phi) &= 0 \quad | \div \lambda V(\alpha(t)) \\
\sin(\alpha(t)-\phi) - \cot(\varphi(\alpha(t)))\cos(\alpha(t)-\phi) &= 0 \quad | \cdot \sin(\varphi(\alpha(t))) \\
\sin(\varphi(\alpha(t)))\sin(\alpha(t)-\phi) - \cos(\varphi(\alpha(t)))\cos(\alpha(t)-\phi) &= 0 \\
\cos(\varphi(\alpha(t)) + \alpha(t) - \phi) &= 0 \\
\varphi(\alpha(t)) + \alpha(t) - \phi &= \pm\frac{\pi}{2} \\
\varphi(\alpha(t)) + \alpha(t) &= \phi \pm \frac{\pi}{2}. \qquad (3.35)
\end{aligned}
$$

Note that $\varphi(\alpha(t)) + \alpha(t)$ is the slope of the line tangent to the speed polar plot. Setting this angle to constant ϕ corresponds to a straight line path with a fixed heading angle equal to $\alpha(t)$.

If $\lambda = 0$, then $\psi_1 = \psi_2 = 0$, and equation (3.28) implies that $\psi_0 = 0$. However, Pontryagin Principle does not permit the vector $(\psi_0, \psi_1, \psi_2, \psi_3)$ to be zero. So λ cannot equal to zero for this scenario.

2. Otherwise, $\frac{\partial H}{\partial u} \neq 0, \forall t$, implying $\psi_3(t)\frac{V(\alpha(t))}{R(\alpha(t))} \neq 0 \; \forall t$, then equation (3.30) states that $u^* = \pm 1$ corresponding to the sharpest possible turn.

Thus, we can state the following proposition based on the application of Pontryagin's Principle.

Proposition 3.10. *Any optimal path is the concatenation of the arcs with minimum turning radius $R(\theta)$ and the straight line segments all parallel to the fixed directions defined by equation (3.35).*

3.3 Further Analysis of an Optimal Path Structure

In the previous section, we demonstrated the controllability of our problem, proved existence of an optimal path and derived the necessary condition for optimality (Proposition 3.10). Pontryagin's Principle provides very important infor-

mation about the structure of an optimal path; however, the derived results are only *necessary* conditions for optimality and define a very large set of potential optimal paths. In this section, we present further analysis of an optimal path structure and characterize a more specific, and significantly smaller, set of optimal path candidates.

In Section 3.3.1, we introduce terminology and notation to be used through out the chapter. Section 3.3.2 lists general observations and properties that follow directly from the definitions and control model of the problem. These statements are used extensively in the upcoming proofs and provide a more intuitive understanding of our analysis. Finally, Section 3.3.3 describes the detailed analysis and the resulting characterization of an optimal path when the polar plot of an agent's speed is convex. Section 3.3.4 presents an algorithm that facilitates the construction of an optimal path.

3.3.1 Terminology and Notation

- D_{st} - the displacement vector from the starting point (x_s, y_s) to the destination point (x_t, y_t), that is $D_{st} = (x_t, y_t) - (x_s, y_s)$ (see Figure 3.4).
- $\alpha(D_{st})$ - angle of the displacement vector D_{st} (see Figure 3.4).

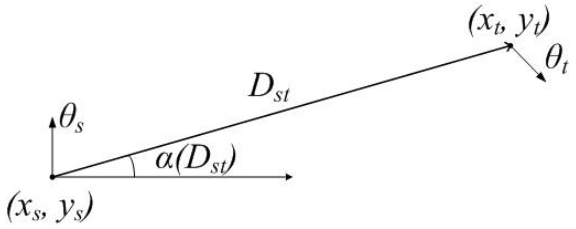

Figure 3.4: The displacement vector $D_{st} = (x_t, y_t) - (x_s, y_s)$.

- *Right-hand sharpest turn curve* - a continuous curve in $\mathbb{R}^2$ corresponding to a clockwise turn with the minimum turning radius, i.e., $u(t) = -1$ in Equation (3.15) (see Figure 3.5).

- *Left-hand sharpest turn curve* - a continuous curve in $\mathbb{R}^2$ corresponding to a counterclockwise turn with the minimum turning radius, i.e., $u(t) = 1$ in Equation (3.15) (see Figure 3.5).

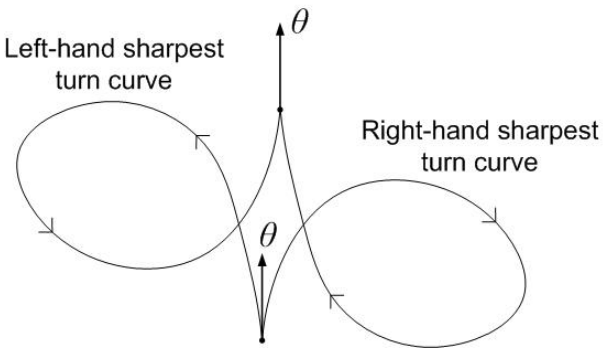

Figure 3.5: Right-hand and left-hand sharpest turn curves.

- $\mathcal{C}_R(\theta_1, \theta_2)$ - a continuous segment (arc) of the right-hand sharpest turn curve that starts at the heading angle θ_1, ends at the angle θ_2, and spans an interval of headings smaller than 2π. The curve is defined for $\theta_1, \theta_2 \in [0, 2\pi]$. Consequently, if $\theta_2 > \theta_1$ we assume that the curve ends at the angle $\theta_2 - 2\pi$. See Figure 3.6.

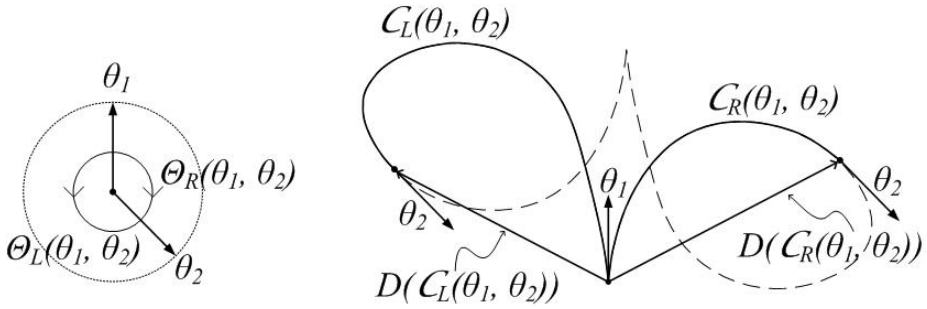

Figure 3.6: Polar plot of $\Theta_R(\theta_1, \theta_2)$ and $\Theta_L(\theta_1, \theta_2)$, and the corresponding curves $\mathcal{C}_R(\theta_1, \theta_2)$ and $\mathcal{C}_L(\theta_1, \theta_2)$.

- $\Theta_R(\theta_1, \theta_2)$ - a set of heading angles spanned by the right-hand sharpest turn curve $\mathcal{C}_R(\theta_1, \theta_2)$, see Figure 3.6. Since the heading angle cannot be changed instantaneously, it is natural for the set of spanned angles to be a continuous

interval of heading angles. However for consistency, we assume that $\Theta_R(\theta_1, \theta_2) \subseteq [0, 2\pi]$. Consequently,

$$\Theta_R(\theta_1, \theta_2) := \begin{cases} [\theta_2, \theta_1], & \text{if } \theta_1 > \theta_2 \\ [0, \theta_1] \cup [\theta_2, 2\pi], & \text{if } \theta_2 > \theta_1 \\ \emptyset & \text{if } \theta_1 = \theta_2 \end{cases} . \tag{3.36}$$

- $\mathcal{C}_L(\theta_1, \theta_2)$ - a continuous segment (arc) of the left-hand sharpest turn curve that starts at the heading angle θ_1, ends at the angle θ_2, and spans an interval of headings smaller than 2π. The curve is defined for $\theta_1, \theta_2 \in [0, 2\pi]$. Consequently, if $\theta_1 > \theta_2$ we assume that the curve ends at the angle $\theta_2 + 2\pi$. See Figure 3.6.

- $\Theta_L(\theta_1, \theta_2)$ - a set of heading angles spanned by the left-hand sharpest turn curve $\mathcal{C}_L(\theta_1, \theta_2)$, see Figure 3.6. Since the heading angle cannot be changed instantaneously, it is natural for the set of spanned angles to be a continuous interval of heading angles. However for consistency, we assume that $\Theta_L(\theta_1, \theta_2) \subseteq [0, 2\pi]$. Consequently,

$$\Theta_L(\theta_1, \theta_2) := \begin{cases} [\theta_1, \theta_2], & \text{if } \theta_1 < \theta_2 \\ [0, \theta_2] \cup [\theta_1, 2\pi], & \text{if } \theta_2 < \theta_1 \\ \emptyset & \text{if } \theta_1 = \theta_2 \end{cases} . \tag{3.37}$$

- $\mathcal{C}.(\theta_1, \theta_2)$ - a sharpest turn curve representing either $\mathcal{C}_R(\theta_1, \theta_2)$ or $\mathcal{C}_L(\theta_1, \theta_2)$. To simplify the notation, we sometimes omit the subscripts R and L and write $\mathcal{C}.(\theta_1, \theta_2)$ when the actual direction of the curve has no significant value to the statement. Consequently, it is assumed that any statement made regarding $\mathcal{C}.(\theta_1, \theta_2)$ is true for both $\mathcal{C}_R(\theta_1, \theta_2)$ and $\mathcal{C}_L(\theta_1, \theta_2)$.

- $\Theta.(\theta_1, \theta_2)$ - a set of heading angles spanned by the curve $\mathcal{C}.(\theta_1, \theta_2)$, where the omitted subscript denoted by '.' can be replaced by either R or L. It is implied

that in any single statement the direction of the curve $\mathcal{C}.(\theta_1, \theta_2)$ and the set $\Theta.(\theta_1, \theta_2)$ match.

- $\mathcal{C}_{.,2\pi}(\theta_1)$ - a sharpest turn curve that makes a complete 2π turn and starts (as well as ends) at the heading angle θ_1, see Figure 3.7. We often refer to the curve as a 2π-*curve*.

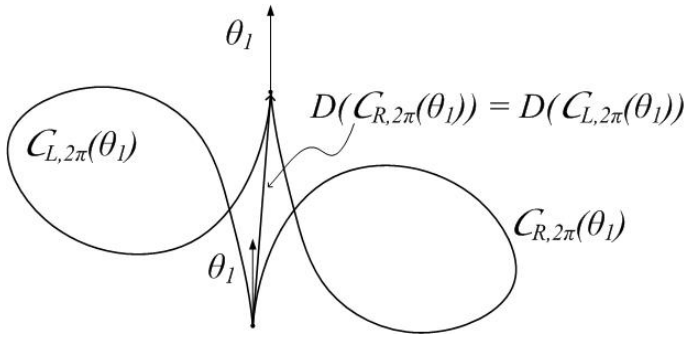

Figure 3.7: Definition of $\mathcal{C}_{R,2\pi}(\theta_1)$, $\mathcal{C}_{L,2\pi}(\theta_1)$ and the corresponding displacement vector.

- $\tau(.)$ - a travel time function that returns the total travel time along a path specified as an input. For example, $\tau(\mathcal{C}_R(\theta_1, \theta_2))$ denotes the travel time along curve $\mathcal{C}_R(\theta_1, \theta_2)$.

- $D(.)$ - a displacement vector (from the start point to the end point) for a path given as an input. For example, $D(\mathcal{C}_R(\theta_1, \theta_2))$ denotes the displacement vector corresponding to a curve $\mathcal{C}_R(\theta_1, \theta_2)$, see Figure 3.6.

- $\alpha(.)$ - a heading angle of a vector specified as an input. See $\alpha(D_{st})$ and Figure 3.4 for example.

- $\|\Theta.(\theta_1, \theta_2)\|$ - denotes the size of the set $\Theta.(\theta_1, \theta_2)$, which is equal to the sum of the lengths of the angle intervals belonging to the set.

- $(x_R, y_R) := (x_s, y_s) + D(\mathcal{C}_R(\theta_s, \theta_t))$ - the end point of a right-hand sharpest turn curve $\mathcal{C}_R(\theta_s, \theta_t)$ that starts at point (x_s, y_s), see Figure 3.8.

- $(x_L, y_L) := (x_s, y_s) + D(\mathcal{C}_L(\theta_s, \theta_t))$ - the end point of a left-hand sharpest turn curve $\mathcal{C}_L(\theta_s, \theta_t)$ that starts at point (x_s, y_s), see Figure 3.8.

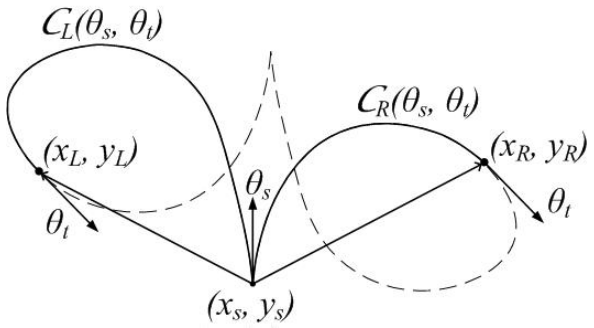

Figure 3.8: Definition of (x_R, y_R) and (x_L, y_L).

- *(Time) reversed curve of some curve* - a curve that spans the same set of headings but in the reversed order. For example, curve $\mathcal{C}_L(\theta_2, \theta_1)$ is the (time) reversed curve of (or corresponding to) $\mathcal{C}_R(\theta_1, \theta_2)$, see Figure 3.11.

- *Time reversed path* - a path corresponding to a specified heading angle sequence traversed in the reversed order. This is a generalization of a *time reversed curve*.

3.3.2 Some General Observations and Properties

In this section, we provide general observations and properties of the sharpest turn curves as defined above. In addition to being a foundation for our further analysis of an optimal path, these statements provide an intuitive understanding of the problem structure. Listed properties are the direct derivations from the problem statement and the control model established in equations (3.13)-(3.16). Therefore, we choose to omit their proofs since they are straightforward exercises from calculus and differential equations.

Observation 3.11. *For any $\theta_1, \theta_2 \in [0, 2\pi]$, the minimum turning radius function $R(\theta)$ uniquely characterizes the right-hand and left-hand sharpest turn curves*

$\mathcal{C}_R(\theta_1, \theta_2)$ *and* $\mathcal{C}_L(\theta_1, \theta_2)$.

Observation 3.12. *Speed function* $V(\theta)$ *describes how fast a vehicle moves along a sharpest turn curve (or any other path); that is,* $V(\theta)$ *uniquely characterizes the functional* $\tau(.)$.

Property 3.13. *The slope of a displacement vector for an arbitrary curve must belong to the set of heading angles spanned by that curve, if the size of the spanned angles set is less than or equal to* π. *That is, for some* θ_1 *and* θ_2, $\alpha(D(\mathcal{C}.(\theta_1, \theta_2))) \in \Theta.(\theta_1, \theta_2)$ *if* $||\Theta.(\theta_1, \theta_2)|| \leq \pi$. *(See curve* $\mathcal{C}_R(\theta_1, \theta_2)$ *and the corresponding displacement vector* $D(\mathcal{C}_R(\theta_1, \theta_2))$ *in Figure 3.6 for a visual illustration.)*

Property 3.14. *Consider an arbitrary curve* $\mathcal{C}.(\theta_1, \theta_2)$ *with* $||\Theta.(\theta_1, \theta_2)|| \leq \pi$ *and the following two lines:* l_1 *passing through the start point of the curve with the slope* θ_1, *and* l_2 *passing through the end point of the curve with slope* θ_2. *Then, the curve* $\mathcal{C}.(\theta_1, \theta_2)$ *does not intersect* l_1 *and* l_2 *except for its start and end points, respectively.*

Furthermore, if $\Theta.(\theta_1, \theta_2) < \pi$, *the curve lies in the triangle region bounded by lines* l_1, l_2 *and* $D(\mathcal{C}.(\theta_1, \theta_2))$, *see Figure 3.9.*

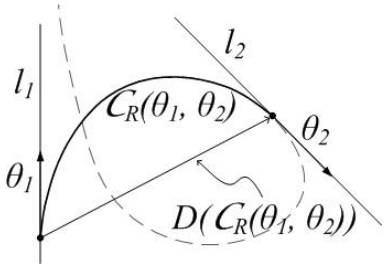

Figure 3.9: Illustration of Property 3.14.

Property 3.15. *A sharpest turn curve* $\mathcal{C}.(\theta_1, \theta_2)$ *with* $||\Theta.(\theta_1, \theta_2)|| > \pi$ *can have at most one point of intersection with itself. Furthermore, its displacement vector* $D(\mathcal{C}.(\theta_1, \theta_2)) \notin \Theta.(\theta_1, \theta_2)$ *only if the curve has an intersection point with itself. And*

$D(\mathcal{C}.(\theta_1, \theta_2)) \in \Theta.(\theta_1, \theta_2)$ *if the curve* $\mathcal{C}.(\theta_1, \theta_2)$ *does not have a point of intersection, see Figure 3.10.*

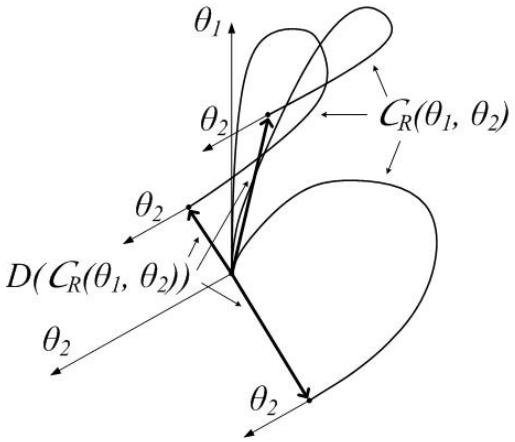

Figure 3.10: Illustration of the possible sharpest turn curves as described in Property 3.15.

Property 3.16. *The travel times and the displacement vectors for a pair of time reversed curves are equal to each other. That is,* $\tau(\mathcal{C}_R(\theta_1, \theta_2)) = \tau(\mathcal{C}_L(\theta_2, \theta_1))$ *and* $D(\mathcal{C}_R(\theta_1, \theta_2)) = D(\mathcal{C}_L(\theta_2, \theta_1))$ *for arbitrary* $\theta_1, \theta_2 \in [0, 2\pi]$*, see Figure 3.11.*

More generally, the travel times and displacement vectors are the same for a pair of paths that are time reversed of each other.

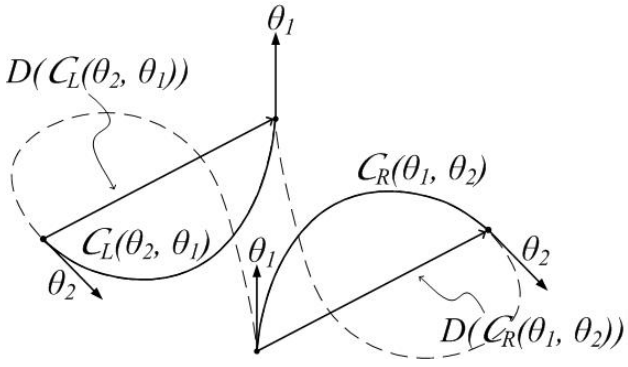

Figure 3.11: Illustration of $\mathcal{C}_R(\theta_1, \theta_2)$ and $\mathcal{C}_L(\theta_2, \theta_1)$ properties.

Property 3.17. *The travel time and displacement vector are the same for a 2π-curve regardless of the starting (and ending) heading angle. That is, $\tau(\mathcal{C}_{.,2\pi}(\theta_1)) = \tau(\mathcal{C}_{.,2\pi}(\theta_2))$ and $D(\mathcal{C}_{.,2\pi}(\theta_1)) = D(\mathcal{C}_{.,2\pi}(\theta_2))$ for any θ_1 and θ_2, see Figure 3.12.*

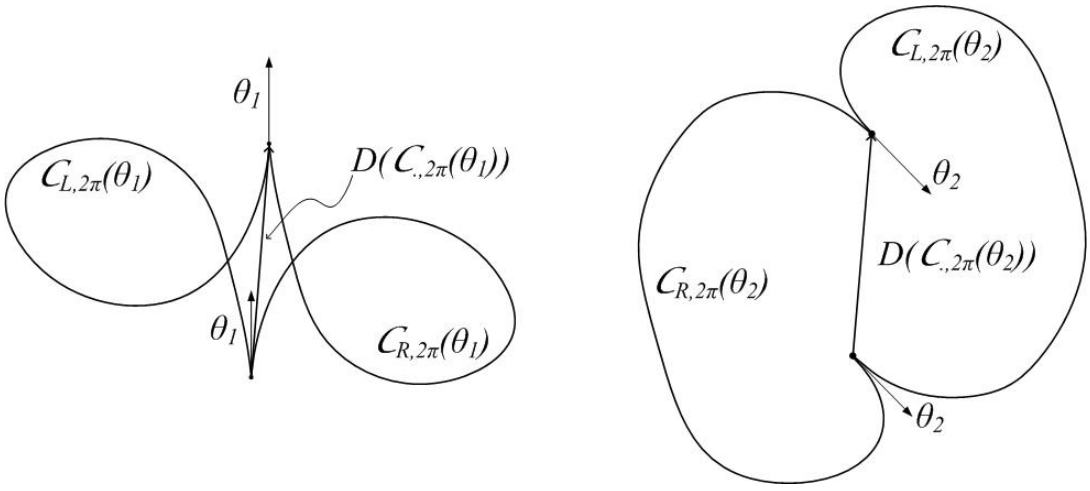

Figure 3.12: Properties of $\tau(\mathcal{C}_{.,2\pi}(\theta_1))$ and $D(\mathcal{C}_{.,2\pi}(\theta_1))$.

Since the travel time $\tau(.)$ and the displacement vector $D(.)$ are equal for the right-hand and left-hand 2π sharpest turn curves (Property 3.16), as well as for any pair of starting heading angles (Property 3.17), we simplify our notation to the following.

- $\tau(C_{2\pi}) := \tau(\mathcal{C}_{.,2\pi}(\theta_1)),\ \forall\theta_1.$
- $D(C_{2\pi}) := D(\mathcal{C}_{.,2\pi}(\theta_1)),\ \forall\theta_1.$

Property 3.18. *Due to the additivity property of time and displacement, the total travel time and the total displacement vector for any given path are equal to the sum of travel times and displacements for the path segments, respectively.*

Property 3.19. *Consider an arbitrary feasible path divided into a finite number of path segments which are then arbitrarily rearranged into an alternative continuous path. Time and space homogeneous properties of the functions $R(\theta)$ and $V(\theta)$ and*

Property 3.18 imply that the rearranged path has the same total travel time and displacement vector as the original path.

3.3.3 Characterization of an Optimal Path for a Convex Linear Path Attainable Region

In this section, we conduct an in-depth analysis of problems with a speed function, $V(\theta)$, that corresponds to a convex linear path attainable region (LPAR), as it is defined in section 2.1.3. This characteristic structure of the agent's speed is also referred to as a 'convex speed polar plot' in literature, and we use the two terms interchangeably. Prior to proceeding with the discussion of an optimal path with bounded curvature, we recall one of the key results for a convex LPAR, which is derived in Chapter II. We employ this result to establish further properties of the paths and the corresponding travel times that are beneficial to our subsequent analysis.

Properties of a Fastest Path for a Convex Linear Path Attainable Region

In the previous chapter, we discuss fastest-path finding for the problems without any constraints on the minimum turning radius (i.e., $R(\theta) = 0, \forall\theta$). For that scenario, we find an optimal path between any two points to be a straight line segment when the speed function corresponds to a convex linear path attainable region. For completeness, we restate the paraphrased version of the lemma here without providing the proof.

Lemma 3.20. *In the case of a convex linear path attainable region, the travel time along the straight line path from $a \in \mathbb{R}^2$ to $b \in \mathbb{R}^2$ is never greater than that of any other path from a to b.*

Proof. See Lemma 2.2. □

The result of Lemma 3.20 is a cornerstone of our analysis; however a present constraint on the minimum turning radius makes a straight line path infeasible for most of the cases. Therefore, a more general result is needed for our further investigation. The following theorem provides a key property of the paths corresponding to the convex speed polar plots.

Theorem 3.21. *Consider a rectifiable path p^* from point $a \in \mathbb{R}^2$ to point $b \in \mathbb{R}^2$ such that the curve p^* and a line segment ab enclose a convex set $S_{p^*} \subseteq \mathbb{R}^2$. (We call such path to be a* convex path*). Consider another path p from a to b, and let $S_p \subseteq \mathbb{R}^2$ denote the set of points enclosed by the curve p and a line segment ab. When the movement along any path is defined by a speed function corresponding to a convex linear path attainable region, the travel time along path p^*, denoted by $t(p^*)$, is never greater than the travel time along path p, $t(p)$, if $S_{p^*} \subseteq S_p$ (see Figure 3.13).*

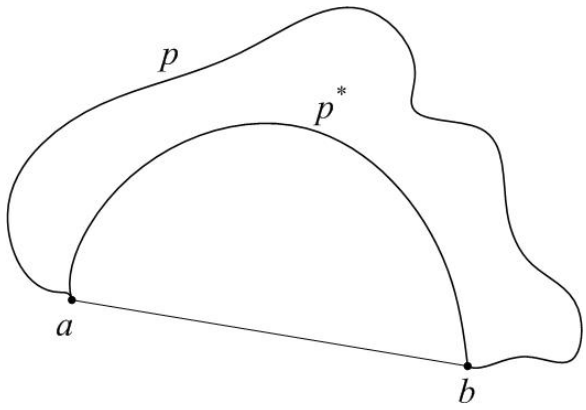

Figure 3.13: Example of paths p^* and p as described in Theorem 3.21.

Proof. To compute $t(p^*)$ we apply polygonal approximation to the path $p^* : [0, 1] \rightarrow \mathbb{R}^2$. Choose an arbitrary partition Π of the interval $[0, 1]$, i.e., let $\Pi = (r_0, r_1, r_2, ..., r_k)$ such that $0 = r_0 < r_1 < r_2 < ... < r_{k-1} < r_k = 1$. Let mesh $|\Pi|$ be the maximum length $r_i - r_{i-1}$ of a subinterval of Π, that is, $|\Pi| = \max_{1 \leq i \leq k}\{r_i - r_{i-1}\}$. Then Π

defines a polygonal approximation to p^*, i.e., the polygonal arc from $p^*(0) = a$ to $p^*(1) = b$ having successive vertices $p^*(r_0)$, $p^*(r_1)$, ..., $p^*(r_k)$ (see Figure 3.14).

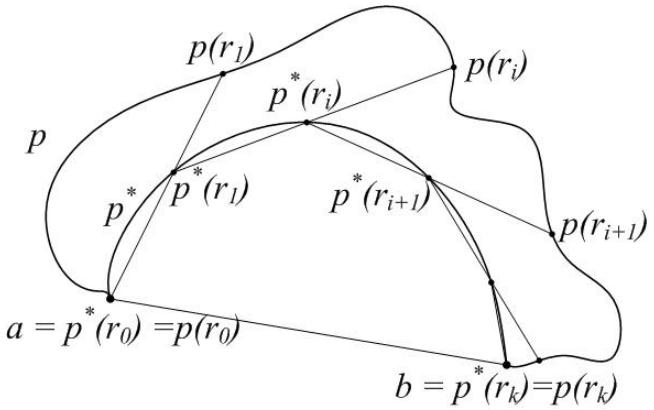

Figure 3.14: Illustration of the proof of Theorem 3.21.

Then, the travel time along a polygonal approximation of the path can be written as $\eta(p^*, \Pi) = \sum_{i=1}^{k} \tau(p^*(r_{i-1}), p^*(r_i))$, where a function $\tau(c, d)$ denotes the travel time along a straight line segment cd. However, as we let $|\Pi|$ approach zero, thus increasing the number of vertices, the polygonal approximation in the limit is equal to path p^*; then so are their travel times (this follows from the assumption that path p^* is rectifiable). Given this,

$$t(p^*) = \lim_{|\Pi| \to 0} \eta(p^*, \Pi). \tag{3.38}$$

Next, we compare $t(p^*)$ to $t(p)$. Since S_{p^*} is convex and $S_{p^*} \subseteq S_p$, a line containing segment $p^*(r_i)p^*(r_{i+1})$ for any $0 \leq i < k$ has to intersect curve p at least once. We let $p(r_{i+1})$ denote the intersection point closest to point $p^*(r_{i+1})$ and such that the line segment $p^*(r_i)p(r_{i+1})$ contains point $p^*(r_{i+1})$. Then, if we let $t(p(r_i, r_{i+1}))$ to represent the travel time along a segment of the curve p between the points $p(r_i)$ and $p(r_{i+1})$, Lemma 3.20 implies the following inequalities for $i \in [0, ..., k-1]$ (see

Figure 3.14).

(3.39)

$$\tau(p^*(r_i), p(r_i)) + t(p(r_i, r_{i+1})) \geq \tau(p^*(r_i), p(r_{i+1})) = \tau(p^*(r_i), p^*(r_{i+1})) + \tau(p^*(r_{i+1}), p(r_{i+1})).$$

Note that $p^*(r_0) = p(r_0)$ and $p^*(r_k) = p(r_k)$ implying that $\tau(p^*(r_0), p(r_0)) = \tau(p^*(r_k), p(r_k)) = 0$.

Summing together inequalities (3.39) for all $i \in [0, ..., k-1]$ results in the following inequality.

$$\begin{aligned}
\Sigma_{i=0}^{k-1} t(p(r_i, r_{i+1})) + \Sigma_{i=0}^{k-1} \tau(p^*(r_i), p(r_i)) &\geq \Sigma_{i=0}^{k-1} \tau(p^*(r_i), p^*(r_{i+1})) + \Sigma_{i=1}^{k} \tau(p^*(r_i), p(r_i)) \\
\Sigma_{i=0}^{k-1} t(p(r_i, r_{i+1})) + \tau(p^*(r_0), p(r_0)) &\geq \Sigma_{i=0}^{k-1} \tau(p^*(r_i), p^*(r_{i+1})) + \tau(p^*(r_k), p(r_k)) \\
\Sigma_{i=0}^{k-1} t(p(r_i, r_{i+1})) &\geq \Sigma_{i=0}^{k-1} \tau(p^*(r_i), p^*(r_{i+1})) \\
t(p) &\geq \eta(p^*, \Pi) \qquad (3.40)
\end{aligned}$$

Combining together equation (3.38) and inequality (3.40) we obtain the desired result $t(p) \geq t(p^*)$. □

Initial Analysis of an Optimal Path Structure

We now return to the discussion of a fastest-path finding problem when the curvature of a path is constrained by a minimum turning radius function $R(\theta)$. From Pontryagin's Principle and the derived Proposition 3.10 we know that there exists an optimal path that consists of the sharpest turn curves and straight line segments. Observe that following a sharpest turn curve is the only way to change the heading angle. On the other hand, in the case of a convex speed polar plot, following a straight line path is faster than following any other path (Lemma 3.20). So intuitively, one follows a sharpest turn curve to change the heading as fast as possible whenever it is necessary to do so; however, one prefers to travel along a straight

line segment whenever that is possible in order to reach the destination sooner. The problem consists of finding an optimal combination of the sharpest turn curves and line segments.

For a distinct pair of starting and target heading angles (i.e., $\theta_s \neq \theta_t$) we know that somewhere along an optimal path one has to traverse a right-hand or a left-hand sharpest turn curve from θ_s until θ_t. In other words, an optimal path must contain either $\mathcal{C}_R(\theta_s, \theta_t)$ or $\mathcal{C}_L(\theta_s, \theta_t)$, which may or may not be split into a finite number of segments throughout the path. (If $\theta_s = \theta_t$, we set $\mathcal{C}.(\theta_s, \theta_t) = \emptyset$ and follow the same discussion.) Property 3.19 implies that we can separate all path segments that make up the sharpest turn curve from θ_s until θ_t and rearrange them into such curve. Thus we divide an optimal path into a sharpest turn curve (either $\mathcal{C}_R(\theta_s, \theta_t)$ or $\mathcal{C}_L(\theta_s, \theta_t)$) and the 'sub-path' containing the remaining path segments. Consequently, an optimal path from (x_s, y_s) to (x_t, y_t) corresponds to either one of the following two cases (see Figure 3.15):

1. the curve $\mathcal{C}_R(\theta_s, \theta_t)$ and an optimal sub-path from point (x_R, y_R) to (x_t, y_t) (with some restrictions on the heading angles along the sub-path which will be discussed later); or

2. the curve $\mathcal{C}_L(\theta_s, \theta_t)$ and an optimal sub-path from point (x_L, y_L) to (x_t, y_t) (with some restrictions on the heading angles along the sub-path which will be discussed later).

In the subsequent sections we find fastest paths corresponding to each of the two scenarios; the path that has the smaller travel time among the two cases is established to be a fastest path for our problem. To simplify the notation, we let D_{Rt} denote the displacement vector from (x_R, y_R) to (x_t, y_t), that is $D_{Rt} := D_{st} - D(\mathcal{C}_R(\theta_s, \theta_t))$.

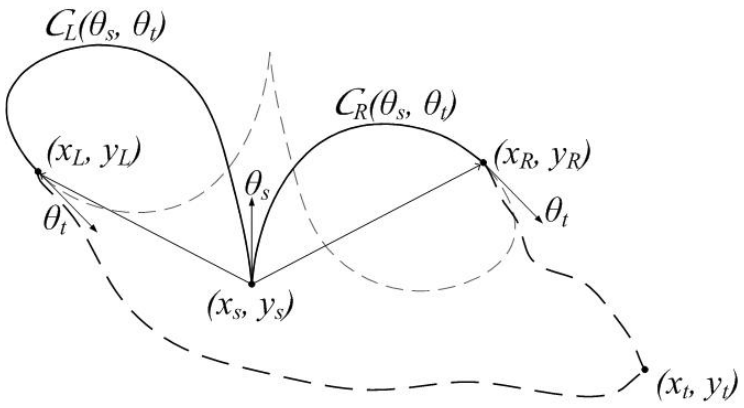

Figure 3.15: An optimal path can be broken down into a sharpest turn curve from θ_s until θ_t (bold solid lines) and the remaining sub-path (bold dashed lines).

Similarly, let $D_{Lt} := D_{st} - D(\mathcal{C}_L(\theta_s, \theta_t))$.

Lower and upper bounds on the minimum travel time

Next, we use our initial analysis of an optimal path and Lemma 3.20 to compute the lower and upper bounds on the minimum travel time. These bounds are important for evaluating potential improvement of a feasible solution and the tradeoff between the resources required to find an optimal path and decrease in travel time. In addition, we employ the lower bounds to establish the optimality of a proposed path and utilize the upper bounds to constrain a set of optimal-path candidates.

We know that an optimal path has to contain either a right-hand turn curve $\mathcal{C}_R(\theta_s, \theta_t)$ or a left-hand turn curve $\mathcal{C}_L(\theta_s, \theta_t)$, so we find lower and upper travel time bounds conditioned on the curve direction. The scenario corresponding to a smaller lower (or upper) bound determines the lower (or upper) bound for our problem.

First, we find a lower bound on the minimum travel time when an optimal path has to contain a right-hand sharpest turn curve $\mathcal{C}_R(\theta_s, \theta_t)$; we denote such bound by LB_R. We already established that an optimal path corresponding to this case contains the curve $\mathcal{C}_R(\theta_s, \theta_t)$ and an optimal sub-path from point (x_R, y_R) to point

(x_t, y_t). There are additional constraints on the heading angles along the sub-path, however they do not need to be incorporated into the lower bound calculation. From Lemma 3.20 we know that a straight line path is a fastest path for a convex linear path attainable region. Therefore, a travel time along an optimal sub-path from (x_R, y_R) to (x_t, y_t) cannot be less than the travel time along the straight line connecting the two points, D_{Rt}. Consequently,

$$LB_R = \tau(\mathcal{C}_R(\theta_s, \theta_t)) + \tau(D_{Rt}). \tag{3.41}$$

Analogously to the right-hand sharpest turn curve case, we find a lower bound on the minimum travel time when an optimal path has to contain a left-hand sharpest turn curve $\mathcal{C}_L(\theta_s, \theta_t)$. (The lower bound for this case is denoted by LB_L.)

$$LB_L = \tau(\mathcal{C}_L(\theta_s, \theta_t)) + \tau(D_{Lt}). \tag{3.42}$$

Considering that an optimal path has to correspond to one of the two scenarios discussed above, we can conclude that the overall lower bound on the minimum travel time, denoted by LB, is the minimum of the two bounds. That is,

$$\begin{aligned} LB &= \min\{LB_R; LB_L\} \\ &= \min\{\tau(\mathcal{C}_R(\theta_s, \theta_t)) + \tau(D_{Rt}); \tau(\mathcal{C}_L(\theta_s, \theta_t)) + \tau(D_{Lt})\}. \end{aligned} \tag{3.43}$$

To find an upper bound on the minimum travel time, we construct a feasible path which may or may not be optimal. Consider a path containing a 2π-curve, either $\mathcal{C}_{R,2\pi}(.)$ or $\mathcal{C}_{L,2\pi}(.)$. Then, a straight line with any slope is a feasible part of the path and can be inserted into the path without violating the minimum turning radius constraint. Consequently in the case when an optimal path has to contain the right-hand turn curve $\mathcal{C}_R(\theta_s, \theta_t)$, a feasible sub-path can consist of a 2π-curve and

the straight line segment connecting point $(x_R, y_R) + D(\mathcal{C}_{2\pi})$ to (x_t, y_t). This path provides an upper bound on the minimum travel time, denoted by UB_R.

$$UB_R = \tau(\mathcal{C}_R(\theta_s, \theta_t)) + \tau(\mathcal{C}_{2\pi}) + \tau(D_{Rt} - D(\mathcal{C}_{2\pi})). \tag{3.44}$$

Similarly, we find an upper bound on the minimum travel time when an optimal path has to contain a left-hand sharpest turn curve $\mathcal{C}_L(\theta_s, \theta_t)$; we denote such bound by UB_L.

$$UB_L = \tau(\mathcal{C}_L(\theta_s, \theta_t)) + \tau(\mathcal{C}_{2\pi}) + \tau(D_{Lt} - D(\mathcal{C}_{2\pi})). \tag{3.45}$$

Due to the fact that both of the constructed paths are feasible, the minimum of the two bounds delivers a tighter upper bound.

$$UB \;=\; \min\{UB_R; UB_L\}. \tag{3.46}$$

The constructed upper and lower bounds are used in our subsequent characterization of an optimal sub-path.

Characterization of an optimal sub-path.

We established earlier that an optimal path can be divided into a sharpest turn curve, either $\mathcal{C}_R(\theta_s, \theta_t)$ or $\mathcal{C}_L(\theta_s, \theta_t)$, and the remaining part of the path called 'sub-path'. As we rearrange the segments of an optimal path into a sharpest turn curve $\mathcal{C}.(\theta_s, \theta_t)$, the remaining segments are organized into a continuous sub-path from either (x_R, y_R) or (x_L, y_L) to (x_t, y_t) (see Figure 3.16). Note that the resulting sub-path might have points of discontinuous heading angle where the segments are joined together.

In order for a path to be optimal, any part of the path also has to be optimal. Therefore, we are interested in characterizing the optimal sub-paths from points

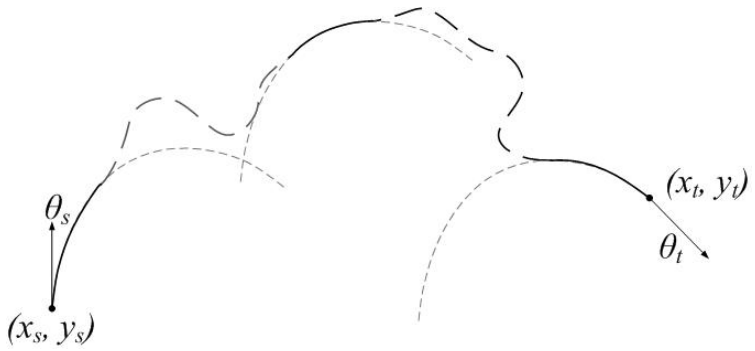

Figure 3.16: An optimal path consists of the segments making up the curve $\mathcal{C}_R(\theta_s, \theta_t)$ (bold solid lines), and the remaining segments (bold dashed lines).

(x_R, y_R) and (x_L, y_L) to the target point (x_t, y_t). The analysis for both cases (starting at (x_R, y_R) and (x_L, y_L)) is practically identical, and we limit our discussion to the case of an optimal sub-path starting at point (x_R, y_R) without loss of generality.

We first look at a special case of the problem when an optimal sub-path can be derived directly from the lower bound computation.

Proposition 3.22. *The straight line segment from (x_R, y_R) to (x_t, y_t) is a fastest sub-path, if the slope of the displacement vector from (x_R, y_R) to (x_t, y_t), i.e., D_{Rt}, is equal to one of the heading angles spanned by the right-hand sharpest turn curve $\mathcal{C}_R(\theta_s, \theta_t)$.*

In other words, if $\alpha(D_{Rt}) \in \Theta_R(\theta_s, \theta_t)$ then a fastest path from (x_s, y_s) to (x_t, y_t) containing a right-hand sharpest turn curve is equal to the curve $\mathcal{C}_R(\theta_s, \alpha(D_{Rt}))$, followed by a straight line segment D_{Rt} and finally followed by the curve $\mathcal{C}_R(\alpha(D_{Rt}), \theta_t)$, see Figure 3.17.

Proof. Straight line is a fastest path in the case of a convex speed polar plot, and the straight line segment from (x_R, y_R) to (x_t, y_t) gives a lower bound on the minimum travel time for the sub-path (see LB_R and equation (3.41)). If $\alpha(D_{Rt}) \in \Theta_R(\theta_s, \theta_t)$, the straight line segment D_{Rt} is feasible, and LB_R implies it is an optimal sub-

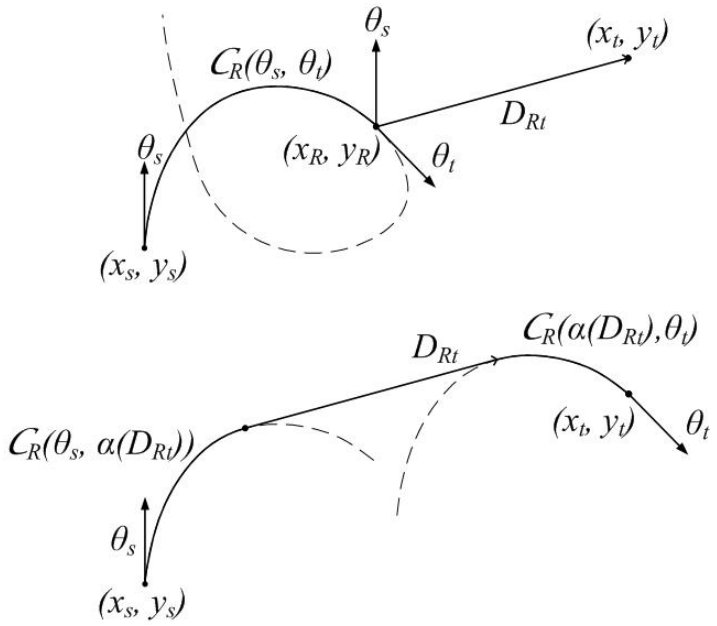

Figure 3.17: Construction of an optimal path as described in Proposition 3.22.

path. □

Proposition 3.22 describes an optimal sub-path (and the resulting path) when $\alpha(D_{Rt}) \in \Theta_R(\theta_s, \theta_t)$. The proposition is not applicable when the straight line from (x_R, y_R) to (x_t, y_t) is not feasible. Therefore, we continue the analysis of an optimal sub-path while assuming that $\alpha(D_{Rt}) \notin \Theta_R(\theta_s, \theta_t)$, which implies $\alpha(D_{Rt}) \in \Theta_L(\theta_s, \theta_t)$.

The necessary heading change from the starting angle θ_s to the target angle θ_t is already accomplished by the curve $\mathcal{C}_R(\theta_s, \theta_t)$ preceding the sub-path. Therefore, the total heading change along the sub-path must be equal to zero (or 2π in some cases). Consequently, every sharpest turn curve along the sub-path must be accompanied by the corresponding time reversed curve to negate the resulting heading change. The only exception is a curve along the sub-path spanning a complete 2π range of headings, in which case the total heading angle also does not change. The earlier computation of an upper bound UB leads to the following fastest sub-path when a

2π-curve is part of an optimal path.

Proposition 3.23. *A fastest sub-path consists of a curve $\mathcal{C}_{R,2\pi}(.)$ or $\mathcal{C}_{L,2\pi}(.)$ and a straight line segment from point $(x_R, y_R) + D(\mathcal{C}_{2\pi})$ to the target point (x_t, y_t), if an optimal sub-path contains a 2π-curve.*

Proof. From the upper bound computation, see equation (3.44), we know that when an optimal sub-path contains a 2π-curve $\mathcal{C}_{R,2\pi}(.)$ or $\mathcal{C}_{L,2\pi}(.)$, the remaining of the sub-path is a straight line. □

From Proposition 3.23 we know the structure of an optimal sub-path if it contains a 2π-curve. Consider the characterization of an optimal sub-path which is assumed not to contain a curve spanning the complete set $[0, 2\pi]$, and when every sharpest turn curve must be accompanied by the corresponding time reversed curve.

The convex property of a linear path attainable region does not guarantee that a travel time function will satisfy the strict triangle inequality, implying that an optimal path might not be unique. Therefore, we are interested in characterizing only one of the optimal sub-paths from (x_R, y_R) to (x_t, y_t). We prove a set of propositions where each consecutive statement adds more detail to the structure of an optimal sub-path without violating the preceding propositions. As a result, we obtain a very specific structure of a sub-path known to be optimal. Note, the assumption that an optimal sub-path does not contain a 2π-curve is implied in all of the following statements.

Proposition 3.24. *There exists a fastest sub-path that does not contain any curves spanning the heading angles belonging to the 'interior' of $\Theta_R(\theta_s, \theta_t)$. In other words, we can construct an optimal sub-path such that any curve $\mathcal{C}.(\theta_1, \theta_2)$ is not a part of it if $\Theta.(\theta_1, \theta_2) \bigcap (\Theta_R(\theta_s, \theta_t)/\{\theta_s, \theta_t\}) \neq \emptyset$.*

Proof. In the case when an arbitrary curve $\mathcal{C}.(\theta_1, \theta_2)$ with the corresponding $\Theta.(\theta_1, \theta_2) \subseteq \Theta_R(\theta_s, \theta_t)$ is part of an optimal sub-path we can replace the curve by a straight line segment equal to $D(C.(\theta_1, \theta_2))$. Without loss of generality, we assume that $\|\Theta.(\theta_1, \theta_2)\| \leq \pi$, since a curve not satisfying this assumption can be split into two curves $\mathcal{C}.(\theta_1, \theta')$ and $\mathcal{C}.(\theta', \theta_2)$, for $\theta' \in \Theta.(\theta_1, \theta_2)$. Property 3.13 then implies that $D(\mathcal{C}.(\theta_1, \theta_2)) \in \Theta.(\theta_1, \theta_2) \subseteq \Theta_R(\theta_s, \theta_t)$ and the feasibility of a straight line segment $D(\mathcal{C}.(\theta_1, \theta_2))$.

The convexity of the linear path attainable region and Lemma 3.20 guarantees that the travel time for the resulting path is not greater than for the original path containing the curve. Furthermore, because every curve is accompanied by its reversed curve, we apply the linear substitution to the time-revered pair of curves, to ensure that the total change in heading angle in the original part of the path and the replaced part of the path are equivalent (i.e., both equal to zero). In such manner, we replace all the 'interior' curves by the straight line segments resulting in the path described in the proposition statement. □

Proposition 3.25. *There exists an optimal sub-path such that the only curves it can contain are $\mathcal{C}_L(\theta_s, \theta_s^*)$ (accompanied by $\mathcal{C}_R(\theta_s^*, \theta_s)$) and $\mathcal{C}_R(\theta_t, \theta_t^*)$ (accompanied by $\mathcal{C}_L(\theta_t^*, \theta_t)$), for some $\theta_s^*, \theta_t^* \in [0, 2\pi]$ such that $\Theta_L(\theta_s, \theta_s^*) \bigcap \Theta_R(\theta_t, \theta_t^*) = \emptyset$.*

Proof. From Proposition 3.24 we know that there exists an optimal sub-path such that none of the curves span the set $\Theta_R(\theta_s, \theta_t)$ except for its 'boundary' points θ_s and θ_t. At the same time, the minimum turning radius constraint implies that there cannot be any discontinuity in the interval of heading angles spanned by the complete path. Consequently, the curve segments must start (and end, when followed by the corresponding time reversed curve) with the heading angles either equal to θ_s or

θ_t. Proposition 3.24 also implies that a curve starting with heading θ_s cannot be a right-hand turn curve, and a curve starting with θ_t cannot be a left-hand turn curve. Finally, the set of headings spanned by the two pairs of curves cannot intersect, otherwise the intersection can be replaced by a straight line similar to the proof of Proposition 3.24. The same proposition implies that there would not be any additional curves spanning the angles belonging to set $\Theta_R(\theta_s^*, \theta_t^*)$. □

It is important to note that both pairs of the curves described in Proposition 3.25 ($\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ and $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$) do not have to be part of an optimal sub-path. In many scenarios, only one of these pairs belongs to an optimal sub-path. For consistency of notation, we let $\theta_s^* = \theta_s$ or $\theta_t^* = \theta_t$ when either curve $\mathcal{C}_L(\theta_s, \theta_s^*)$ or curve $\mathcal{C}_R(\theta_t, \theta_t^*)$ is not part of an optimal sub-path, respectively. In such manner, we always refer to curves $\mathcal{C}_L(\theta_s, \theta_s^*)$ and $\mathcal{C}_R(\theta_t, \theta_t^*)$, recognizing that the curve's length might be equal to zero, implying that it is not part of the path.

It is interesting to observe that in the extreme case when $\Theta_L(\theta_s, \theta_s^*)$ or $\Theta_R(\theta_t, \theta_t^*)$ is equal to $\Theta_L(\theta_s, \theta_t)$, the scenario is equivalent to the path with a left-hand sharpest turn curve $\mathcal{C}_L(\theta_s, \theta_t)$, the 2π-curve $\mathcal{C}_{R,2\pi}(.)$ and the straight line corresponding to the discussed upper bound UB_L, see equation (3.45).

Proposition 3.26. *There exists an optimal sub-path such that it either does not contain any line segments, or all the line segments have the same heading angle and can be rearranged into a single straight line.*

Proof. Consider an optimal sub-path that contains two directed line segments D_1 and D_2 with the distinct heading angles θ_1 and θ_2, respectively. Our goal is to construct an alternative path that can replace D_1 and D_2 part of the given sub-path, contains at most one line segment, and has a travel time no greater than the sum of travel

times for D_1 and D_2. The alternative path has to have the total displacement equal to $D_1 + D_2$. Furthermore, the total heading change along the alternative path has to be equal to zero. Therefore, we construct an alternative path only replacing $\frac{1}{2}D_1$ and $\frac{1}{2}D_2$ parts of the path without the restriction on the total heading change. Property 3.16 implies that by replacing the second part of $\frac{1}{2}(D_1 + D_2)$ with the time-reversed path results in the proper alternative path and negates the overall heading change.

Continuity of the heading angle along a feasible path implies that either $\mathcal{C}_R(\theta_1, \theta_2)$ or $\mathcal{C}_L(\theta_1, \theta_2)$ has to be part of the path containing D_1 and D_2. Without loss of generality, assume $\mathcal{C}_R(\theta_1, \theta_2)$ is part of the path. Consequently, any proposed alternative path has to start with the heading angle in $\Theta_R(\theta_1, \theta_2)$ to ensure its feasibility.

Case 1: $\alpha(D_1 + D_2) \in \Theta_R(\theta_1, \theta_2)$.

When $\alpha(D_1 + D_2) \in \Theta_R(\theta_1, \theta_2)$, the straight line segment equal to the displacement $D_1 + D_2$ is feasible to include into the path, and we can replace D_1 and D_2 by that single line segment. Lemma 3.20 states that the travel time for the resulting path is not greater than for the original path, thus maintaining its optimality.

Case 2: $\alpha(D_1 + D_2) \notin \Theta_R(\theta_1, \theta_2) \Rightarrow \alpha(D_1 + D_2) \in \Theta_L(\theta_1, \theta_2)$.

Since $\alpha(D_1 + D_2) \in \Theta_L(\theta_1, \theta_2)$, we know that $\|\Theta_L(\theta_1, \theta_2)\| \leq \pi$. Let $\theta' := \alpha(D(\mathcal{C}_L(\theta_1, \theta_2)))$, then Property 3.13 implies $\theta' \in \Theta_L(\theta_1, \theta_2)$.

Due to the symmetry of $\mathcal{C}_R(\theta_2, \theta_1)$ and $\mathcal{C}_L(\theta_1, \theta_2)$, we assume $\alpha(D_1 + D_2) \in \Theta_L(\theta_1, \theta')$ as opposed to $\Theta_L(\theta', \theta_2)$, without loss of generality. Let the curve $\mathcal{C}_L(\theta_1, \theta_2)$ and the displacement vector $\frac{1}{2}(D_1 + D_2)$ start at the same point, denoted by a. Then, the curve must intersect the line containing the displacement vector $\frac{1}{2}(D_1 + D_2)$, and we call the intersection point d. Let $\theta_d \in$

$\Theta_L(\theta_1, \theta_2)$ denote the heading angle of the curve $\mathcal{C}_L(\theta_1, \theta_2)$ at point d, that is, $\frac{1}{2}(D_1 + D_2)||D(\mathcal{C}_L(\theta_1, \theta_d)$. Then, if $c := a + \frac{1}{2}(D_1 + D_2)$, either $d \in ac$ or $c \in ad$, and we consider the following two subcases separately.

Case 2a: $d \in ac$ (see Figure 3.18).

We can complete the path from a to c by adding the line segment dc to the curve $\mathcal{C}_L(\theta_1, \theta_d)$. Note that since $\Theta_L(\theta_1, \theta_d) \subseteq \Theta_L(\theta_1, \theta_2)$ and $\|\Theta_L(\theta_1, \theta_2)\| \leq \pi$, Property 3.13 implies that $\alpha(dc) = \alpha(D(\mathcal{C}_L(\theta_1, \theta_d))) \in \Theta_L(\theta_1, \theta_d)$. The resulting path consists of $\mathcal{C}_L(\theta_1, \alpha(dc))$ followed by the line segment dc and then the curve $\mathcal{C}_L(\alpha(dc), \theta_d)$. It is a convex path enclosed by a path $\frac{1}{2}D_1$ followed by $\frac{1}{2}D_2$, and Theorem 3.21 states that its travel time satisfies the requirements to maintain the optimality of the proposed path.

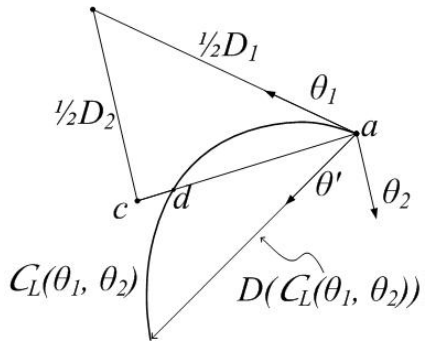

Figure 3.18: Illustration of Proposition 3.26 Case 2a.

Case 2b: $c \in ad$ (see Figure 3.19).

Let point b denote the end of segment $\frac{1}{2}D_1$ starting at a, that is, $b = a + \frac{1}{2}D_1$. Then $bc = \frac{1}{2}D_2$. We also let $\mathcal{C}_a$ denote the curve $\mathcal{C}_L(\theta_1, \theta_2)$ starting at a. The fact that $c \in ad$ and $\alpha(ac) \in \Theta_L(\theta_1, \theta')$ implies that curve $\mathcal{C}_a$ intersects bc, since it does not intersect ac and it cannot intersect ab (Property 3.14). Similarly, we let $\mathcal{C}_c$ to denote the curve $\mathcal{C}_L(\theta_1, \theta_2)$ ending at point c. Then, curve $\mathcal{C}_c$ must intersect ab and curve $\mathcal{C}_a$, we call $e := \mathcal{C}_a \bigcap \mathcal{C}_c$. Furthermore,

the intersection point must lie inside the triangle defined by vertexes *abc*. Then, the alternative proposed path is to follow curve $\mathcal{C}_a$ from point a to point e and then follow curve $\mathcal{C}_c$ from e to c. The constructed path is a convex path enclosed by a path $\frac{1}{2}D_1$ followed by $\frac{1}{2}D_2$, and Theorem 3.21 states that its travel time satisfies the requirements to maintain the optimality of the proposed path.

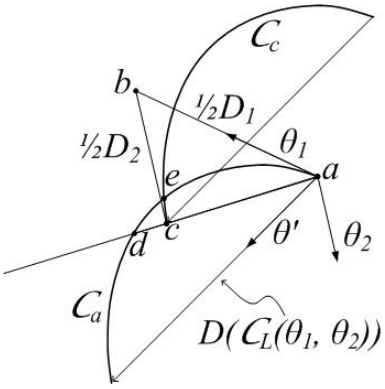

Figure 3.19: Illustration of Proposition 3.26 Case 2b.

□

Proposition 3.27. *There exists an optimal sub-path such that it can only contain the curve pairs* $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ *and* $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$ *and a line segment with the slope either equal to* θ_s^* *or* θ_t^*.

Proof. Proposition 3.25 states that there exists an optimal sub-path that can only contain the curves $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ and $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$. And Proposition 3.26 proves that we can have an optimal sub-path with at most one line segment. The two propositions do not contradict each other, and we can conclude that there exists an optimal sub-path that can only contain the given two pairs of curves and a single line segment. Thus, we are left to prove that the heading angle of the line can only be equal to θ_s^* or θ_t^*, instead of the 'interior' defined as $\Theta_R(\theta_s^*, \theta_t^*)/\{\theta_s^*, \theta_t^*\}$.

Consider a path containing line segment D_1 with the corresponding heading angle θ_1, such that $\theta_1 \in \Theta_R(\theta_s^*, \theta_t^*)/\{\theta_s^*, \theta_t^*\}$. Since $\|\Theta_R(\theta_s^*, \theta_t^*)\| \leq 2\pi$, either $\|\Theta_R(\theta_s^*, \theta_1)\| \leq \pi$ or $\|\Theta_R(\theta_1, \theta_t^*)\| \leq \pi$. Due to the symmetry of the argument, we assume $\|\Theta_R(\theta_1, \theta_t^*)\| \leq \pi$, without loss of generality. Then, our goal is to construct an alternative path with a travel time not greater than the current path, and such that the slope of an alternative line segment has a heading angle equal to θ_s^* or θ_t^*. Note that θ_s^* and θ_t^* are not the specific angle values, but a notation used to denote the 'boundaries' of the set of headings spanned by a path.

Employing Property 3.19, consider a part of the path consisting of D_1 and $\mathcal{C}_R(\theta_1, \theta_t^*)$ accompanied by $\mathcal{C}_L(\theta_t^*, \theta_1)$ which is arranged as follows (see Figure 3.20). Let curve $\mathcal{C}_a$ denote a curve $\mathcal{C}_R(\theta_1, \theta_t^*)$ starting at some point a and let point b denote the end of $\mathcal{C}_a$, that is, $b = a + D(\mathcal{C}_a)$. The line segment D_1 is assumed to start at point b and end at point c, where $c = b + D_1$. Finally, a curve denoted by $\mathcal{C}_c$ is the curve $\mathcal{C}_L(\theta_t^*, \theta_1)$ starting at point c and ending at point d. We let point e denote the midpoint of the line segment bc.

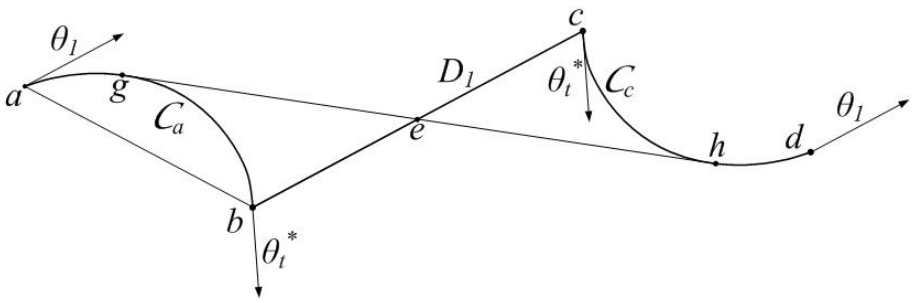

Figure 3.20: Illustration of Proposition 3.27 proof.

Since $D(\mathcal{C}_a) \in \Theta_R(\theta_1, \theta_t^*)$, the curve $\mathcal{C}_a$ and line segment ab enclose a convex set, and point e lies outside this set. Then, we construct two tangent lines to this convex set passing through point e and one of those lines is tangential to curve $\mathcal{C}_a$. We let

point g denote this tangential point and $\theta_g \in \Theta(\theta_1, \theta_t^*)$ be the slope of the tangent line ge, which is also the heading angle of the curve $\mathcal{C}_a$ at point g. Note that the symmetry of curves $\mathcal{C}_a$ and $\mathcal{C}_c$ relative to point e implies that the line containing segment ge is also tangent to curve $\mathcal{C}_c$ at a point we denote by h. The symmetry also implies that the heading angle of the curve $\mathcal{C}_c$ at point h is equal to θ_g. Then, the part of the path consisting of D_1, $\mathcal{C}_R(\theta_1, \theta_t^*)$ and $\mathcal{C}_L(\theta_t^*, \theta_1)$ can be replaced by a curve $\mathcal{C}_R(\theta_1, \theta_g)$, the directed line segment gh, and curve $\mathcal{C}_L(\theta_g, \theta_1)$. Subsequently, heading θ_g becomes the new θ_t^*.

Note that since we replace $\mathcal{C}_R(\theta_g, \theta_t^*)$, $\mathcal{C}_L(\theta_t^*, \theta_g)$ and D_1 by a straight line segment with the equivalent displacement, Lemma 3.20 ensures that the alternative path will have a travel time no greater than the original path, thus maintaining its optimality.

□

Proposition 3.28. *There exists an optimal sub-path described by one of the following statements:*

1. *It consists of only two curve pairs* $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ *are* $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$.

2. *It consists of only one curve pair* $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ *or* $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$ *and a straight line segment with the slope* θ_s^* *or* θ_t^**, respectively.*

Proof. Proposition 3.27 states that there exists an optimal sub-path that can only contain the curve pairs $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ and $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$, and a straight line segment D_1 with the heading angle either equal to θ_s^* or θ_t^*. We show that there exists an optimal sub-path that would not contain all three components.

Consider a path containing $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$, $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$ and a straight line segment D_1, where $\theta_s^* \neq \theta_s$ and $\theta_t^* \neq \theta_t$. Due to the symmetry of the

discussion, we assume that the heading angle of D_1 is equal to θ_t^*, without loss of generality. Employing Property 3.16, we consider one half of the path that consists of curve $\mathcal{C}_R(\theta_t, \theta_t^*)$, curve $\mathcal{C}_R(\theta_s^*, \theta_s)$, and a line segment $\frac{1}{2}D_1$. Our goal is to replace this part of the path with either two sharpest turn curves or one curve and a line segment. Then, implementation of a time reversed path for the second half delivers the necessary results.

Let point a denote the start point of curve $\mathcal{C}_R(\theta_t, \theta_t^*)$, and point b denote the curve's end point. Then, we assume the line segment $\frac{1}{2}D_1$ starts at b and ends at $c = b + \frac{1}{2}D_1$. Finally, we assume $\mathcal{C}_R(\theta_s^*, \theta_s)$ starts at point c and ends at point d. Note that the line segment ad is equal to $\frac{1}{2}D_{Rt}$, implying that $\alpha(ad) \in \Theta_L(\theta_s, \theta_t) = \Theta_R(\theta_t, \theta_s)$. Next, we consider all possible scenarios of the current half-path and prove the proposition for each case individually.

Case 1: Curves $\mathcal{C}_R(\theta_t, \theta_t^*)$ and $\mathcal{C}_R(\theta_s^*, \theta_s)$ intersect at some point e (see Figure 3.21). Consider an alternative path consisting of only two curves: part of the curve $\mathcal{C}_R(\theta_t, \theta_t^*)$ between points a and e, followed by part of the curve $\mathcal{C}_R(\theta_s^*, \theta_s)$ between points e and d. Because the proposed path is part of the original path, its travel time has to be less than or equal to the original travel time.

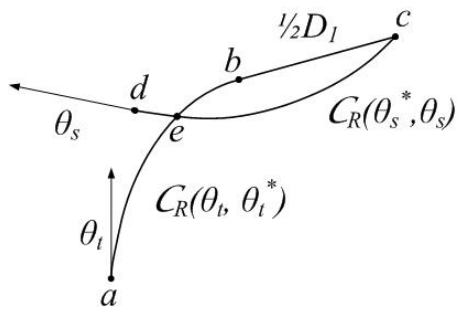

Figure 3.21: Illustration of Proposition 3.28 Case 1.

Case 2: Curve $\mathcal{C}_R(\theta_t, \theta_t^*)$ intersects itself (see Figure 3.22).

We assume that point d does not lie inside the region enclosed by the loop of curve $\mathcal{C}_R(\theta_t, \theta_t^*)$, otherwise curves $\mathcal{C}_R(\theta_t, \theta_t^*)$ and $\mathcal{C}_R(\theta_s^*, \theta_s)$ have to intersect corresponding to Case 1. Then, there exists at least one line tangent to curve $\mathcal{C}_R(\theta_t, \theta_t^*)$ and passing through point d. Let e denote the tangent point of the line to the curve $\mathcal{C}_R(\theta_t, \theta_t^*)$, such that the heading of the curve at that point is equal to $\alpha(ed)$. Consider an alternative path consisting of only one arc and a line segment: part of the curve $\mathcal{C}_R(\theta_t, \theta_t^*)$ between points a and e and a straight line segment ed. Since we replace part of the original path with a straight line segment, Lemma 3.20 implies that travel time of the alternative path is not greater than that of the original path.

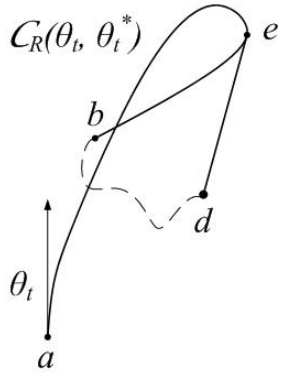

Figure 3.22: Illustration of Proposition 3.28 Case 2.

Case 3: $\alpha(cd) \in \Theta_L(\theta_t^*, \theta_t^* + \pi)$ (see Figure 3.23).

There exists a line segment ed that is tangent to curve $\mathcal{C}_R(\theta_t, \theta_t^*)$ at point e, such that the heading of the curve at that point is equal to $\alpha(ed)$. We construct an alternative path containing one curve an a line segment as discussed in Case 2.

Case 4: $\alpha(cd) \in \Theta_R(\theta_t^*, \theta_t^* + \pi)$.

A curve $\mathcal{C}_R(\theta_t^*, \theta_s)$ starting at point b must either intersect the line segment bd

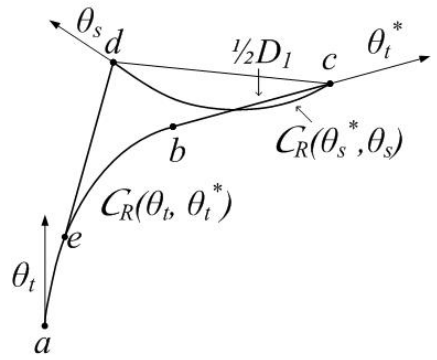

Figure 3.23: Illustration of Proposition 3.28 Case 3.

or curve $\mathcal{C}_R(\theta_s^*, \theta_s)$, before a possible intersection with the line segment bc.

Case 4a: Curve $\mathcal{C}_R(\theta_t^*, \theta_s)$ starting at point b intersects line segment bd (see Figure 3.24).

There exists a line segment ed that is tangent to curve $\mathcal{C}_R(\theta_t^*, \theta_s)$ at point e, such that the heading of the curve at that point is equal to $\alpha(ed)$. Consider an alternative path consisting of a single curve and a line segment: part of the curve $\mathcal{C}_R(\theta_t, \theta_s)$ between points a and e, followed by a straight line segment ed. Theorem 3.21 implies that travel time of the alternative path is not greater than that of the original path.

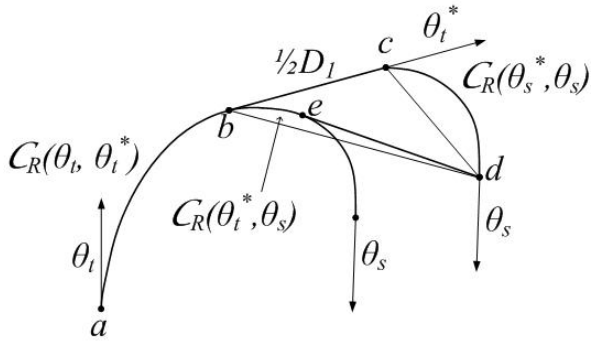

Figure 3.24: Illustration of Proposition 3.28 Case 4a.

Case 4b: Curve $\mathcal{C}_R(\theta_t^*, \theta_s)$ starting at point b intersects curve $\mathcal{C}_R(\theta_s^*, \theta_s)$ (see

Figure 3.25).

Let e denote the intersection point of the two curves. Then a path consisting of two curves: a part of the curve $\mathcal{C}_R(\theta_t, \theta_s)$ between points a and e, and a part of the curve $\mathcal{C}_R(\theta_s^*, \theta_s)$ between points e and d, has a travel time no greater than the original path (Theorem 3.21). Note that the set of heading angles spanned by the two new curves cannot intersect.

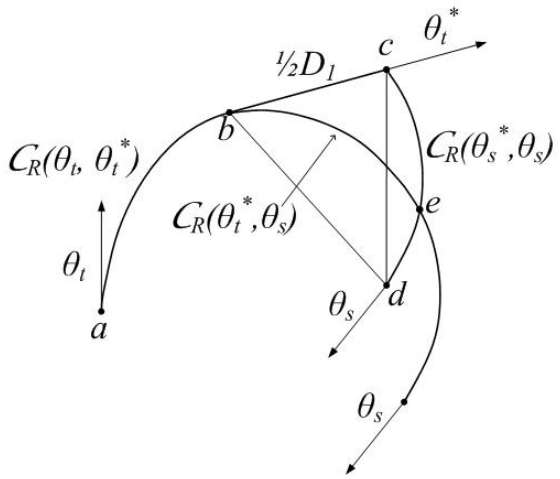

Figure 3.25: Illustration of Proposition 3.28 Case 4b.

□

Characterization of an optimal path

Theorem 3.29. *An optimal path from (x_s, y_s, θ_s) to (x_t, y_t, θ_t) is of the form $\{\mathcal{C}, \mathcal{C}, \mathcal{C}\}$, or $\{\mathcal{C}, S, \mathcal{C}\}$, where $\mathcal{C}$ denotes a sharpest turn curve and S denotes the straight line segment. It is implied that a path of the form $\{\mathcal{C}, \mathcal{C}, \mathcal{C}\}$ alternatively switches between left-hand and right-hand sharpest turn curves.*

Proof. The preceding propositions list all the possible forms of the optimal path candidates and show that each one is either of the form $\{\mathcal{C}, \mathcal{C}, \mathcal{C}\}$ or $\{\mathcal{C}S\mathcal{C}\}$. We discuss the case when an optimal path has to contain $\mathcal{C}_R(\theta_s, \theta_t)$; the case when an

optimal path has to contain the curve $\mathcal{C}_L(\theta_s, \theta_t)$ is analogous.

Case 1: An optimal sub-path is a straight line equal to D_{Rt} (Proposition 3.22).

A path consisting of the curve $\mathcal{C}_R(\theta_s, \alpha(D_{Rt}))$ followed by a straight line segment D_{Rt} which is followed by the curve $\mathcal{C}_R(\alpha(D_{Rt}), \theta_t)$ is of the form $\{\mathcal{C}, \mathbf{S}, \mathcal{C}\}$.

Case 2: A fastest sub-path consists of a 2π-curve and a straight line segment from point $(x_R, y_R) + D(\mathcal{C}_{2\pi})$ to the target point (x_t, y_t) (Proposition 3.23).

Let D_1 denote a displacement vector from point $(x_R, y_R) + D(\mathcal{C}_{2\pi})$ to the target point (x_t, y_t). Then an optimal path is as follows: a curve $\{\mathcal{C}_R(\theta_s, \theta_t), \mathcal{C}_R(\theta_t, \alpha(D_1))\}$, followed by a line segment equal to D_1, and concluded with a curve $\mathcal{C}_R(\alpha(D_1), \theta_t)$, which is of the form $\{\mathcal{C}, \mathbf{S}, \mathcal{C}\}$.

Case 3: An optimal sub-path consists of only curves $\mathcal{C}_L(\theta_s, \theta_s^*)$ (accompanied by $\mathcal{C}_R(\theta_s^*, \theta_s)$) and $\mathcal{C}_R(\theta_t, \theta_t^*)$ (accompanied by $\mathcal{C}_L(\theta_t^*, \theta_t)$) (Proposition 3.28).

An optimal path is as follows (in order to maintain the continuity of heading): a curve $\mathcal{C}_L(\theta_s, \theta_s^*)$, followed by a curve $\mathcal{C}_R(\theta_s^*, \theta_t^*) = \{\mathcal{C}_R(\theta_s^*, \theta_s), \mathcal{C}_R(\theta_s, \theta_t), \mathcal{C}_R(\theta_t, \theta_t^*)\}$ and followed by another curve $\mathcal{C}_L(\theta_t^*, \theta_t)$, which is of the form $\{\mathcal{C}, \mathcal{C}, \mathcal{C}\}$

Case 4: An optimal sub-path consists of only one curve pair $\{\mathcal{C}_L(\theta_s, \theta_s^*), \mathcal{C}_R(\theta_s^*, \theta_s)\}$ and a straight line segment with the slope θ_s^* (Proposition 3.28).

Then an optimal path is a curve $\mathcal{C}_L(\theta_s, \theta_s^*)$, a straight line segment with the slope θ_s^*, and then followed by a curve $\mathcal{C}_R(\theta_s^*, \theta_t) = \{\mathcal{C}_R(\theta_s^*, \theta_s), \mathcal{C}_R(\theta_s, \theta_t)\}$. This path is of the form $\{\mathcal{C}, \mathbf{S}, \mathcal{C}\}$.

Case 5: An optimal sub-path consists of only one curve pair $\{\mathcal{C}_R(\theta_t, \theta_t^*), \mathcal{C}_L(\theta_t^*, \theta_t)\}$ and a straight line segment with the slope θ_t^* (Proposition 3.28).

Then an optimal path is a curve $\mathcal{C}_R(\theta_s, \theta_t^*) = \{\mathcal{C}_R(\theta_s, \theta_t), \mathcal{C}_R(\theta_t, \theta_t^*)\}$, a straight

line segment with the slope θ_t^*, followed by a curve $\mathcal{C}_L(\theta_t^*, \theta_t)$. This path is of the form $\{\mathcal{C}, \mathbf{S}, \mathcal{C}\}$.

□

Theorem 3.29, which explicitly characterizes an optimal path in the case of a convex speed polar plot, is the key theorem of this chapter. It is interesting to note that the structure of our optimal path is similar to that of an isotropic Dubins car problem. However, it is important to remember that curves in our case do not generally correspond to the circles like in Dubins car problem. Instead, they might have very complex forms. Despite the very general form of the sharpest-turn curves, we show that our problem is controllable and has an optimal solution as characterized in Theorem 3.29.

3.3.4 Optimal Path Finding Algorithm for a Convex Linear Path Attainable Region

We develope a path finding algorithm to facilitate the implementation of an optimal path with bounded curvature as characterized in Theorem 3.29. While the main premise of our algorithm is the result of Theorem 3.29, we state additional propositions in order to further characterize an optimal path based on the relative location of the target state (x_t, y_t, θ_t).

Proposition 3.30. *If an optimal sub-path does not contain a 2π-curve, there exists an optimal sub-path from point (x_R, y_R) to point (x_t, y_t) such that it passes through the mid-point of the connecting line, D_{Rt}.*

Proof. From Proposition 3.27 we know that any sharpest turn curve of an optimal sub-path must be accompanied by the corresponding time-revered curve. Furthermore, Property 3.16 states that the displacement vector for a curve and its reversed

curve are equal. Similarly, we can split a straight line segment part of an optimal sub-path into two segments with equal lengths. Then, we can construct a sub-path such that the first and second halves of the path (in respect of time) are time-reversed of each other. We know that the total displacement vector for the first half of the path must be equal to the second half, and therefore corresponding to vector $\frac{1}{2}D_{Rt}$. $\square$

Any part of an optimal path also has to be optimal, therefore we construct half of an optimal sub-path with the displacement equal to $\frac{1}{2}D_{Rt}$. Then, applying Proposition 3.30 we set the second half of the sub-path to be the corresponding time-reversed path. Thus, Proposition 3.30 facilitates the construction of an optimal sub-path as characterized in Theorem 3.29.

Before stating the proposition, we introduce notation to ensure the clarity of our arguments. Let point $a = (x_R, y_R)$ and point b denote the mid-point of D_{Rt} starting at a, that is $b = a + \frac{1}{2}D_{Rt}$. Let curves $\mathcal{C}_{a,L} = \mathcal{C}_L(\theta_s, \theta_t)$ and $\mathcal{C}_{a,R} = \mathcal{C}_R(\theta_t, \theta_s)$ that start at point a and end at point $c = a + D(\mathcal{C}_{a,L}) = a + D(\mathcal{C}_{a,R})$. We let $\mathcal{S}_R \subseteq \mathbb{R}^2$ to denote a region enclosed the curve $\mathcal{C}_{a,R}$ and a line segment ac. Similarly, $\mathcal{S}_L$ is the region enclosed by $\mathcal{C}_{a,L}$ and ac. We set $\mathcal{S}' = \mathcal{S}_R \bigcup \mathcal{S}_L$.

Proposition 3.31. *When $\alpha(D_{Rt}) \notin \Theta_R(\theta_s, \theta_t)$ and a 2π-curve is not part of an optimal sub-path, one of the following cases describes the half of an optimal sub-path, from a to b.*

Case 1: *$\alpha(cb) \in \Theta_R(\theta_t, \theta_s)$ and $b \notin \mathcal{S}'$ (see Figure 3.26).*

There exist two lines passing through point b, such that one line is tangent to curve $\mathcal{C}_{a,L}$ and one is tangent to curve $\mathcal{C}_{a,R}$, at the corresponding points d_L and d_R, and the slopes $\theta_{d_L} = \alpha(d_L b)$ and $\theta_{d_R} = \alpha(d_R b)$, respectively. Then the half of a fastest sub-path is either the curve $\mathcal{C}_L(\theta_s, \theta_{d_L})$ followed by a line segment

$d_L b$, or the curve $\mathcal{C}_R(\theta_t, \theta_{d_R})$ followed by $d_R b$.

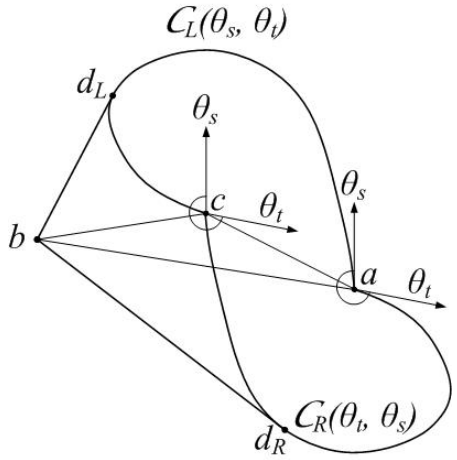

Figure 3.26: Illustration of Proposition 3.31 Case 1.

Case 2: *$\alpha(cb) \notin \Theta_R(\theta_t, \theta_s)$ and $b \notin \mathcal{S}'$ (see Figure 3.27).*

There exists only one line passing through point b that is tangent to either curve $\mathcal{C}_{a,L}$ or $\mathcal{C}_{a,R}$. We call the tangent point d and the heading of the sharpest turn curve at that point $\theta_d = \alpha(db)$. The half of a fastest sub-path is the curve $\mathcal{C}.(\theta_s, \theta_d)$ followed by a line segment db.

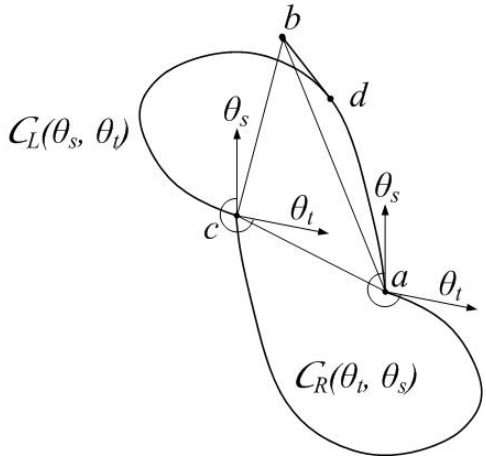

Figure 3.27: Illustration of Proposition 3.31 Case 2.

Case 3: *$\alpha(cb) \in \Theta_R(\theta_t, \theta_s)$ and $b \in \mathcal{S}'$ (see Figure 3.28).*

Due to symmetry of the argument, we assume $b \in \mathcal{S}_L$, without loss of generality. There exists a line passing through point b, tangent to curve $\mathcal{C}_{a,R}$, at point d with the heading of the sharpest turn curve at that point $\theta_d = \alpha(db)$. In addition, a curve $\mathcal{C}_L(\theta_s, \theta_t)$ ending at point b, denoted by $\mathcal{C}_{b,L}(\theta_s, \theta_t)$, intersects the curve $\mathcal{C}_{a,L} = \mathcal{C}_L(\theta_s, \theta_t)$ at some point denoted by e. Then, the half of a fastest sub-path will be one of the two paths: (1) curve $\mathcal{C}_R(\theta_t, \theta_d)$ followed by a line segment db, or (2) part of the curve $\mathcal{C}_{a,L} = \mathcal{C}_L(\theta_s, \theta_t)$ between points a and e followed by part of the curve $\mathcal{C}_{b,L} = \mathcal{C}_L(\theta_s, \theta_t)$ between points e and b.

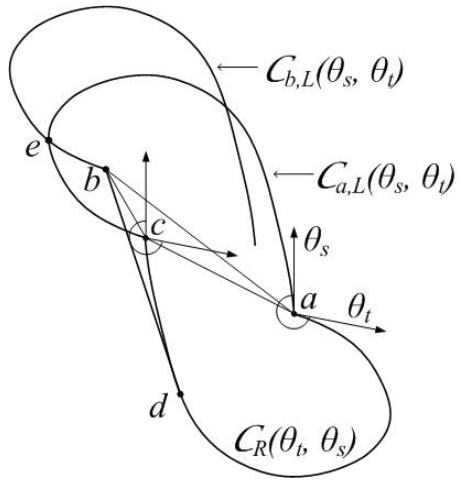

Figure 3.28: Illustration of Proposition 3.31 Case 3.

Case 4: *$\alpha(cb) \notin \Theta_R(\theta_t, \theta_s)$ and $b \in \mathcal{S}'$ (see Figure 3.29).*

Due to symmetry of the argument, we assume $b \in \mathcal{S}_L$, without loss of generality. Then, a curve $\mathcal{C}_L(\theta_s, \theta_t)$ ending at point b, denoted by $\mathcal{C}_{b,L}(\theta_s, \theta_t)$, intersects the curve $\mathcal{C}_{a,L} = \mathcal{C}_L(\theta_s, \theta_t)$ at a point denoted by e. And, the half of a fastest sub-path is part of the curve $\mathcal{C}_{a,L} = \mathcal{C}_L(\theta_s, \theta_t)$ between points a and e followed by part of the curve $\mathcal{C}_{b,L} = \mathcal{C}_L(\theta_s, \theta_t)$ between points e and b.

Proof. For each case, we demonstrate that the proposed path is faster than other candidate paths characterized in Proposition 3.28.

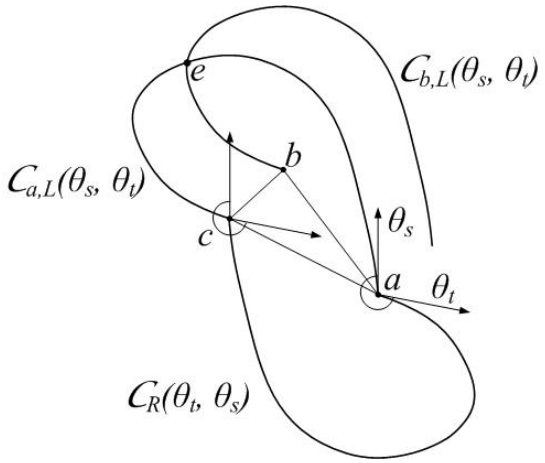

Figure 3.29: Illustration of Proposition 3.31 Case 4.

Case 1: When, $\alpha(cb) \in \Theta_R(\theta_t, \theta_s)$ and $b \notin \mathcal{S}'$, the constructed straight line segments $d_R b$ and $d_L b$ are faster than part of a second sharpest turn curve ending at point b (Theorem 3.21), since the second curves ending at point b either would not intersect the curves starting at point a or the set of heading angles spanned by parts of the two curves would overlap.

Case 2: The constructed path is the only feasible candidate. A second curve ending at point b either would not intersect a curve starting at point a or the set of heading angles spanned by parts of the two curves would overlap.

Case 3: Analogous to Case 1 for a candidate path (1). For a candidate path (2), it is then only feasible candidate including a curve $\mathcal{C}_{a,L} = \mathcal{C}_L(\theta_s, \theta_t)$.

Case 4: The constructed path is the only feasible candidate. A second curve ending at point b either would not intersect a curve starting at point a or the set of heading angles spanned by parts of the two curves would overlap.

□

We would like to note that in order to ensure that our analysis and results can be

implemented to a wide variety of applications, we make no specific restrictions on the minimum turning radius function. However, this very general direction-dependent function produces a diverse set of sharpest turn curve and limits how specific and definitive our path characteristic can be. Therefore, despite similar characterizations, the construction of an optimal path for our problem is more evolved than for Dubins car problem.

We now present an algorithm that finds a fastest path with bounded curvature for a direction-dependent speed function corresponding to a convex LPAR. We know that an optimal path has to contain a curve $\mathcal{C}_R(\theta_s, \theta_t)$ or $\mathcal{C}_L(\theta_s, \theta_t)$, and without knowing more specific information about the functions $R(\theta)$ and $V(\theta)$ we cannot conclude which one of these cases is optimal. Consequently, we construct paths for each scenario and compare the minimum travel time for each case.

Algorithm 4 Fastest Path with Bounded Curvature for a Convex Speed Polar Plot.

Step 1. For both cases $k = R$ and $k = L$, corresponding to the scenarios where an optimal path must contain a right-hand sharpest turn curve $\mathcal{C}_R(\theta_s, \theta_t)$, and a left-hand sharpest turn curve $\mathcal{C}_L(\theta_s, \theta_t)$, respectively, perform the following steps 1a - 1d.

Step 1a. Set $\tau_k = \infty$.

Step 1b. Let point $a = (x_s, y_s) + D(\mathcal{C}_k(\theta_s, \theta_t))$, and the displacement vector $D_{at} = (x_t, y_t) - a$. If $\alpha(D_{at}) \in \Theta_k(\theta_s, \theta_t)$, compute the travel time $\tau_k = \tau(\mathcal{C}_k(\theta_s, \theta_t)) + \tau(D_{at})$. STOP. Repeat steps 1a - 1d for the other k if needed, otherwise go to Step 2.

Step 1c. Set point $a' = a + D(\mathcal{C}_{2\pi})$ and the displacement vector $D_{a't} = (x_t, y_t) - a'$. Update the travel time $\tau_k = \min\{\tau_k, \tau(\mathcal{C}_k(\theta_s, \theta_t)) + \tau(\mathcal{C}_{2\pi}) + \tau(D_{a't})\}$.

Step 1d. Set $b = a + \frac{1}{2} D_{at}$. Construct an optimal sub-path as described in Proposition 3.31 and update τ_k if the found path is faster.

Step 2. Compare τ_R and τ_L, the smaller value is the minimum travel time, and a path corresponding to that travel time is optimal.

3.4 Conclusion

This chapter presents a closed form characterization of a fastest path with bounded curvature where the mobile agents speed and minimum turning radius are direction-

dependent functions. While path finding with curvature constraint is extensively studied in the fields of robotics and UAV routing, this work is the first to analytically characterize an optimal path in a generalized direction-dependent environment. Furthermore, the presented algorithm provides an explicit procedure for constructing an optimal path for given starting and target states of the system.

It is important to note that, while our main results are restricted to the problems with convex speed polar plots, most of our discussion pertains to a very general anisotropic speed function. We establish problem controllability, prove existence of an optimal path, and derive a necessary condition of optimality (an optimal path contains only sharpest turn arcs and straight line segments), without restricting the speed function to correspond to a convex linear path attainable region. In our future work, we plan to extend our analysis of an optimal path to a non-convex speed polar plot and derive an analytical characterization of a fastest path for a general anisotropic case.

CHAPTER IV

Dynamic Programming Modeling for Optimal Path Finding in a Direction, Location and Time Dependent Environment

4.1 Introduction

In previous chapters, an optimal path in a time and space homogeneous direction-dependent environment was found; in other words, the cost function and constraints are assumed to be independent of the time and location of an agent. In the current chapter, we relax the assumption of homogeneity and discuss a dynamic programming modeling for optimal path finding in a direction, location and time dependent environment.

Current technological advancement in real-time data collection and forecasting calls for an explicit incorporation of the available information into the decision making process. Innovative on-board sensors, such as a doppler radar in the Optimum Vessel Performance in Evolving Nonlinear Wave-Fields (OVPENWF) project, collect information about the surrounding environment in real time. The optimal path finding model presented in this chapter makes use of the gathered information. Physical sensors have a limited visibility horizon and cannot gather information about the medium beyond a specified distance. Consequently, the model presumes to have complete information about the environment (i.e., cost function and constraints) within the radar visibility horizon and limited information beyond that horizon.

The majority of optimal path finding problems require integration of additional constraints in order to facilitate real life implementation. In the navigation of aerial, ground and naval vehicles, the curvature of a feasible path is constrained by a minimum turning radius function. This prevents us from using a traditional dynamic programming path finding model that optimizes over a set of piecewise-linear paths. *Subsequently, instead of addressing the optimal-path finding and path-following aspects of the problem separately, we integrate the systems operability and dynamics constraints into an optimization model resulting in a control-feasible solution.*

The benefits of innovative sensor technology and real-time data collection are subject to timely utilization of the available information in a decision making process. Therefore, computational demand and run-time of the optimal path finding model is of particular significance. Traditional modeling of a path finding algorithm for a dynamic network (i.e., time-dependent cost functions) requires the addition of a time variable to the model state space. This increase in the dimension of a state space increments the computational time of an algorithm by orders of magnitude. In our optimal path finding model, we undertake this, and other, computationally demanding aspects of the problem and present modeling and implementation methods that significantly improve the run-time.

The optimal path finding model presented in this chapter is motivated by the MURI Optimum Vessel Performance in Evolving Nonlinear Wave-Fields project discussed in the introduction of this dissertation. Consequently, we address the problems faced while working on this project. Nevertheless, all the discussed features of our model arise in a great number of applications. And we ensure that the analysis and results presented here are general enough to be applicable to a large group of problems and can be easily translated to other applications. The vessel routing problem

is an example to illustrate a practical application and motivation of the model.

4.1.1 Related Work

In this subsection, we give a broad overview of the published work that is related to the optimal path finding problems in a direction, location and time dependent medium. In the forthcoming sections of the chapter, a more detailed analysis of the relevant publications as they pertain to the specific focus of each section is presented.

An optimal path finding problem in a location, and possibly time, dependent environment is widely studied in the literature. For example, the Zermelo Navigation Problem, introduced by Zermelo in 1931 [74], is to steer a vessel along a minimum-time path through a region of the position-dependent strong current vectors, while the ship's velocity relative to water is assumed to be constant. This is a classic problem in the fields of calculus of variation and optimal control theory [8], and a number of extensions and variations have been studied since the introduction of the problem.

The majority of the published work assumes that a closed form speed function is available to the user, facilitating an application of various optimal control and calculus tools to the analytical function [24, 25, 39, 54, 55]. However, a closed form cost function implies a simplistic model of the surrounding environment and its effect on the moving agent, consequently jeopardizing the accuracy of an optimal path finding model and its solution. In addition, the construction of an analytical function from data gathered by the on-board sensors in a discrete form is often too time consuming to facilitate its implementation in real-time.

When analytical cost and constraint functions are not available, discrete dynamic programming (DP) provides an efficient optimal path finding model. Dynamic programming has been widely studied and applied to an extensive list of problems for

over half a century, since the publication of a ground-breaking book by Richard Bellman in 1957 [5]. As Eric Denardo later writes, "It was Bellman who seized upon the principle of optimality and, with remarkable ingenuity, used it to analyze hundreds of optimization problems in mathematics, engineering, economics, operations research, and other fields" [18]. Application of dynamic programming is particulary beneficial to the complex problems requiring a sequential decision making. DP simplifies the problem at hand by creating a set of sub-problems, solutions of which results in an optimal solution to the original problem. Dynamic programming modeling of the path finding problems involves the discretization of the path domain into a set of 'waypoints' (or nodes), consequently reducing the problem to optimization in a directed network [18]. We expand this approach to deliver a control-feasible path in a dynamic environment while decreasing the algorithm's run-time and computational demand.

4.1.2 Overview of the Results

This chapter analyzes the optimal path finding problems with a direction, location and time dependent medium. We relax the assumption of time and space homogeneity from the preceding chapters and deliver an efficient path finding algorithm that incorporates the detailed real-time information about the surrounding environment.

Information about the neighboring environment and the corresponding cost function (e.g., agent's speed) is evaluated by the discreet computer simulations and forecasting models. Therefore, a cost function in closed form is not available, and an optimal path cannot be found analytically. We discretize the path space into a set of *waypoints* and construct a dynamic programming model that finds an optimal ordered set of waypoints to traverse from a start point to a target point.

A traditional dynamic programming formulation of an optimal path finding prob-

lem implements a straight line segment between a pair of consecutive waypoints. Alternatively, we present a model that delivers a control-feasible path and does not violate the system dynamics restrictions. We add mobile agent's heading angle to the state space of our model and evaluate a local minimum turning radius function that constrains the curvature of the feasible paths between the waypoints. The assumption of stationary distribution for the local environment (a region including the next set of waypoints to be considered at a given DP state) facilitates the implementation of our earlier results for fastest path with bounded curvature (Algorithm 4) in order to find an arc cost between a pair of states.

To incorporate the physical limitations of the data-collecting sensors and radars, we introduced a 'visibility horizon' that restricts how far from the agent's current location detailed information about the medium is available. Due to limited knowledge of the invisible region, the environment beyond the visibility horizon is assumed to be time and space homogeneous. Consequently, by implementing our results from previous chapters (Algorithm 1, Algorithm 3 or Algorithm 4) we find an optimal path to continue the travel from the visibility horizon to the target point located in the invisible region.

Finally, we address the computational demand associated with path finding in a time-dependent environment. A traditional approach to path finding in a dynamic network is to include the time variable into the DP state. We present a dynamic programming model that integrates the agent's speed controller, consequently eliminating the time variable form the model's state space. We redefine the cost function to denote the smallest elapse of time between the arrivals to a given waypoint and the subsequent waypoint along the path. This new arc cost implies that it is always optimal to reach the intermediate points of a path as fast possible and eliminates

the need to keep track of all the feasible arrival times to each waypoint. Algorithm 5 presented in this chapter integrates all the discussed aspects of the path finding model into a single algorithm facilitating its implementation.

The rest of this chapter is organized as follows. Subsection 4.1.3 introduces the notation to be used throughout the chapter and gives a formal statement of the problem. The following three sections discuss the three main aspects that are addressed in the path finding model: limited visibility horizon of the on-board sensors (Section 4.2), integration of the agent's dynamic constraints (Section 4.3), and efficient integration of the medium time-dependency (Section 4.4).

Section 4.5 integrates all the discussed aspects into a single path finding model. An algorithm is delivered that brings together all of the work presented in this dissertation into a single optimal path finding model for a direction, location and time dependent environment. Section 4.6 discusses specific techniques that further improve the efficiency of our Algorithm 5 at its implementation and programming stage. We also present the results of implementing the path finding model to the Optimum Vessel Performance in Evolving Nonlinear Wave-Fields project. Our main discussion of this chapter focuses on a fastest-path finding problems, however in Section 4.7 we discuss the general optimal path finding problems where the objective is to minimize a cost function other than the path travel time. Finally, Section 4.8 concludes this chapter.

4.1.3 Notation and Problem Statement

We are interested in finding a fastest path from a starting point $s = (x_s, y_s) \in \mathbb{R}^2$ to a target point $t = (x_t, y_t) \in \mathbb{R}^2$, where t_0 denotes the start time. Without loss of generality, we assume $t_0 = 0$. Since the curvature of any feasible path is constrained by a minimum turning radius function, the initial and target heading angles at points

s and t can affect the set of feasible paths considered in our problem. Therefore, we integrate the starting heading, denoted by $\theta_s \in [0, 2\pi]$, and the final heading, $\theta_t \in [0, 2\pi]$, as an input to our problem. For many applications, the target heading angle is not specified, in which case we find a path minimizing the travel time over all possible values of θ_t.

All paths from s to t lie in a direction, location and time dependent environment in $\mathbb{R}^2$. We let R_H denote a *radar visibility horizon* that restricts how far the onboard sensors can collect information about the surrounding environment relative to the agent's current location. Thus, for any point a inside the visible region and $t_a \geq t_0$, we are given the functions $V(a, \theta_a, t_a)$, denoting a maximum attainable speed, and $R(a, \theta_a, t_a)$, indicating the minimum turning radius, for a mobile agent located at the point a and heading in the direction θ_a at time t_a. We assume that in addition to knowing information about the environment at time t_0, the forecasting tools use real-time data and evaluate the functions $V(a, \theta_a, t_a)$ and $R(a, \theta_a, t_a)$ for $t_a \geq t_0$. We do not explicitly integrate an upper bound on t_a for which the information is available, implying that an agent leaves the visible region prior to achieving the forecasting upper bound.

The definition of R_H implies that we do not have the explicit $V(a, \theta_a, t_a)$ and $R(a, \theta_a, t_a)$ functions outside the radar visibility horizon. We defer our discussion on how to collect, characterize and integrate the necessary information for the region beyond R_H to Section 4.2.

Now, we give the formal statement of our problem.

Problem statement: Find a fastest path starting at time $t_0 = 0$ from the initial state $(s, \theta_s) \in \mathbb{R}^2 \times [0, 2\pi]$ to the target state $(t, \theta_t) \in \mathbb{R}^2 \times [0, 2\pi]$, where for all $a \in \mathbb{R}^2 : \|s - a\| \leq R_H$ and all $t_a \geq t_0$ the curvature of a feasible path is constrained

by a minimum turning radius function $R(a, \theta_a, t_a)$, and the maximum attainable speed is described by function $V(a, \theta_a, t_a)$.

For completeness of this subsection, we introduce additional notation that is not necessary for the problem statement, but is used throughout the chapter in the discussion of our DP model. To construct a discrete dynamic programming path finding model we have to discretize our $\mathbb{R}^2$ path domain. We let l denote a dynamic programming discretization parameter representing the distance between any pair of consecutive waypoints connected by an arc. We assume that a 'local region' with its radius equal to l can be approximated by a time and space homogeneous environment.

In addition, we let $\mathcal{N}_H$ denote a discrete set of waypoints (or nodes) on the visibility horizon through which all the paths from s to t considered by our model have to pass (assuming $\|t-s\| > R_H$). That is, $\forall a \in \mathcal{N}_H, \|a-s\| = R_H$. Usually, set $\mathcal{N}_H$ consists of equally spaced points on the circle centered at s with a radius equal to R_H. A more detailed discussion of the discretization parameters is presented later in the sections as they are introduced in our modeling.

4.2 Limited Visibility Horizon

Despite the continuous improvement of sensor and data-collection technology, all physical systems have limitations, and it would be unrealistic for an optimization model to assume an unlimited availability of information. To incorporate this limitation, we introduce the concept of *visibility horizon* and let R_H denote the radius of the sensor-visible region surrounding a mobile agent. Thus, the on-board sensor system is assumed to collect and forecast all the necessary information about the environment (i.e., $V(a, \theta_a, t_a)$ and $R(a, \theta_a, t_a)$) within the visibility horizon. Alternatively, the model does not have access to the detailed real-time information for the

medium lying further than R_H distance away from the current location.

Depending on a specific application, there are various ways to estimate and characterize the invisible environment. In some cases, information collected by the sensors inside the visibility region can be extrapolated beyond the horizon R_H. Alternatively, prior experience and historical data for a given or similar region can be used to evaluate a stationary distribution of the environment that an agent expects to observe. In other cases, supplementary forecast and sensor technology can be employed to give a global estimation for the medium. For example, most airplanes and vessels use meteorological and hydrological forecasts provided by the national and international agencies to obtain information about the surrounding environment.

The forecasts described above provide information about the environment on a global scale and describe an expected distribution of waves or wind an agent will face. As a result, the limited information beyond R_H requires us to assume a stationary environment for that region, implying time and space homogeneity of the cost function and constraints. Consequently, we assume that the environment outside the visibility horizon is a stationary distributed stochastic system, and for each instance of the problem a single fixed parameter can characterize this random distribution. For example, in the case of naval navigation, a parameter called 'sea state' describes the wave spectrum as a stationary random process over a short-term "time period in the range from $\frac{1}{2}$ hour to maybe 10 hours" [23].

Time and space homogeneity in the region beyond the radar visibility horizon implies that there is no need for a dynamic programming model to evaluate an optimal path for that part of the travel. Instead, the closed form solutions presented in Chapter II and Chapter III can quickly deliver an optimal path in the stationary random process, when an expected speed function and constraints are known for a

given distribution of the environment (e.g., sea state). Consequently, we develop a dynamic programming model that evaluates the fastest paths to a discrete set of points on the border of the visible region, i.e., all the points in set $\mathcal{N}_H$. Then, either Algorithms 1 (Chapter II) or Algorithm 4 (Chapter III) for a homogeneous medium finds the best path to continue from R_H border to the target point, t. Note that when the distance to a destination point is relatively long compared to the minimum turning radius of an agent, the curvature constraint can be neglected and the results of Chapter II (Algorithm 1) can be used to evaluate an optimal path to traverse through the invisible region, see Figure 4.1. Furthermore, the extreme weather conditions and physical obstacles beyond R_H can be integrated into the optimization model by implementing Algorithm 3 instead of Algorithm 1.

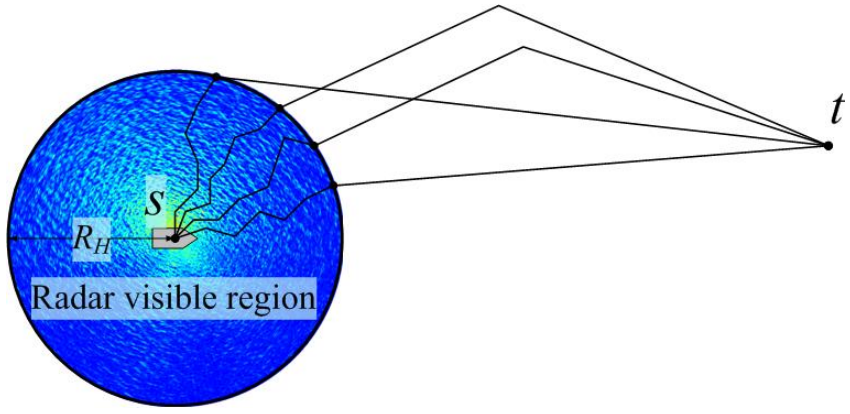

Figure 4.1: DP model evaluates the fastest paths to the points on R_H, and Algorithm 1 finds the best paths to continue.

Real-time data collection and timely implementation of the fastest-path finding algorithm allows us to continuously update the information about the surrounding environment and reevaluate an optimal path to reflect the most current information. As an agent moves along the path getting closer to the destination point, the radar visibility region moves towards t as well. Our goal is to deliver an optimal path finding algorithm with the computational time small enough to allow a user to reevaluate

an optimal path before a vehicle travels outside the visibility horizon. Then, the mobile agent never traverses the sensors' invisible region and additional real-time information is collected prior to the agent reaching that region.

Continuous reevaluation of an optimal path is similar in concept to a common dynamic programming practice called 'rolling horizon' [1, 36, 53]. However, it is important to differentiate that in a traditional DP rolling horizon approach the user makes a choice of how far out into the future the algorithm should look before making a decision for the next time period. In our case, one has no choice of how much information to integrate into the optimization model, and our goal is to find the most accurate optimal path while utilizing all the available information about the medium.

To illustrate the scale of a visibility horizon, we provide an example from the Optimum Vessel Performance in Evolving Nonlinear Wave-Fields project. In this particular application, we are interested in missions with the target point located approximately 30 to 90 minutes of travel time away from the start. It is clear that more sophisticated radar equipment with greater visibility radius always delivers a more accurate path finding model. However, our colleagues working on this project approximate that the current limitations of the radar restrict the visibility radius to approximately 10 minutes worth of travel time. While this implies that we do not have the complete information about the environment from the path start to its finish, we have a sufficiently large R_H to be able to reevaluate an optimal path before traveling outside the visible region.

4.3 System Dynamics Restrictions

Next, we discuss the details of a dynamic programming model to be implemented inside the visible region that finds the fastest paths from point s to all the points in $\mathcal{N}_H$. In this section, we direct our attention to integration of the system dynamic constraints.

Dynamic programming is a predominant approach to the fastest-path finding problems in a time and/or location dependent environment. A traditional DP path finding model discretizes the domain of a path into a set of 'waypoints' (or nodes) and a straight line path is implemented between a pair of neighboring waypoints. As a result, the classical dynamic programming model finds an optimal piecewise-linear path. However, in many applications (e.g., vessels, airplanes and cars) a mobile agent cannot instantaneously change its heading, making the piecewise-linear paths infeasible.

This control-infeasibility of an optimal path comes from a traditional approach where optimal-path finding and path-following are considered to be the separate stages of the problem. First, an optimization model is used to find a fastest path neglecting the system dynamics. Then, a control model is implemented to facilitate an agent following the optimal path as closely as possible. Since we know that an optimal path found by the traditional DP model is control infeasible, the inherited error cannot be avoided during the implementation of such a path. To address this concern, we integrate the system's operability and dynamics constraints into the optimization model, which in turn, delivers a control-feasible solution.

As described in Section 4.1.3, a minimum turning radius function $R(a, \theta_a, t_a)$ restricts the curvature of a feasible path and has to be accounted for in the optimization

model. Therefore, instead of implementing a straight line path between a pair of waypoints as it is done in a traditional DP model, we integrate agent's heading angle into the state space of the model and find a fastest path satisfying the curvature constraints between a pair of the new DP states.

We introduce a dynamic programming discretization parameter $l > 0$ that denotes a distance between any pair of consecutive waypoints. That is, at any particular waypoint, say point a, an algorithm chooses a waypoint to travel to next among all the points that are l distance away from point a. Then, in order to compute the travel time for a pair of our DP states (a specified waypoint and a corresponding heading angle), we assume local homogeneity at each discretized waypoint. In other words, a circular region centered at a given waypoint with a radius equal to l is assumed to be time and space homogeneous. Consequently, the problem of finding a minimum travel time path from one state of the model to the next is reduced to a problem of fastest path with bounded curvature in the anisotropic media, which we discuss in Chapter III.

To summarize, in order to integrate the system dynamic restrictions into a DP path finding model, we define the state of the system to be the agent's location in $\mathbb{R}^2$ and the heading at which it arrived. Then, for a given waypoint and heading angle, a dynamic programming model selects a waypoint to travel to next and the arrival direction at that point. We use information about the environment and functions $V(a, \theta_a, t_a)$ and $R(a, \theta_a, t_a)$ to approximate the local (within radius l) time and space homogeneous medium and the corresponding functions $V(\theta)$ and $R(\theta)$. Finally, the fastest path with bounded curvature presented in Chapter III (see Algorithm 4) computes the minimum time to travel to the next waypoint, assigning that to be the arc's cost. The forthcoming Section 4.5 goes into more details and illustrates

mathematical integration of the proposed method into the dynamic programming model.

4.4 Computational Demand of a Time-Dependent Environment

We continue the discussion of a dynamic programming path-finding model that we implement inside the visible region. In this section, we address the time-dependency of the medium. Recall that path finding through a discretized set of waypoints is equivalent to optimization in a directed network [18]. Therefore, we use the terminology from both fields interchangeably: a 'network node' and a 'waypoint' denote the same entity, and a 'network arc' is equivalent to a path connecting a pair of waypoints l distance away from each other.

A time-dependent environment implies that it is not always optimal to arrive at an intermediate waypoint of a path as soon as possible. For example, a mobile agent arriving at some point along a path just a few minutes later might observe more favorable weather, resulting in an overall better cost. To account for this fact, a time variable is traditionally added to the DP state in order to keep track of all possible times at which an agent might arrive at, and consequently leave, a particular waypoint. This additional variable significantly increases the number of dynamic programming states to be considered by an algorithm. We present an alternative formulation of the DP functional equation that allows us to eliminate the time variable from the DP state space.

4.4.1 Related Work

A number of papers have been published that study fastest-path finding problems for a time-dependent network. In such problems, the traveling time from an arbitrary node i to some other node j, denoted by $d_{ij}(t_i)$, is a function of the time one leaves

node i, denoted by t_i. The majority of the literature considers two cases of this problem. First, no waiting is permitted in the network nodes, implying that the time one leaves a given node is equal to the arrive time for that node. In the second case, waiting is allowed in the network nodes and node departure time can be greater than arrival time. We discuss the related work for each case in more details.

A general fastest-path finding problem that prohibits waiting at the nodes is more difficult to solve than a problem with unlimited waiting time. In fact, Orda and Rom [51] show that some instances of this problem are NP hard. However, Kaufman and Smith [32] introduce a *consistency* condition (also referred to in literature as *first in first out* or FIFO property) which guaranties that one would always want to arrive at each intermediate node of the optimal path as soon as possible and continue the travel without a delay. Kaufman and Smith show that under consistency, the time-dependent fastest path can be calculated with exactly the same computational complexity as the static fastest path. However, the consistency condition is too restrictive and does not apply to a number of applications areas, such as aerial and naval vehicle navigation.

Path finding algorithms have also been presented for problems that do not satisfy the consistency condition. Cooke and Halsey [15] propose an algorithm for the 'no-waiting-is-allowed problem' which is restricted to the discrete cost functions whose domain and range lie in the positive integers set. Their algorithm first finds an upper bound on the total travel time to the destination node. It then uses this bound to set the boundary condition for the DP backward recursive equations used to solve the problem. Alternatively, Chabini [10] assumes that all the travel time functions, $d_{ij}(t_i)$, are constant for all $t_i \geq M$, implying that the problem becomes static for any start time greater than M. This assumption lets Chabini use the static shortest

path to set the boundary condition for his dynamic programming formulation. Both papers propose the backward DP formulations of the problem with a time variable being part of the DP state. Furthermore, their methods are close to enumerating all the possible paths since their algorithms interpolate the fastest paths for all possible times of arrival at the destination node.

In the case when unlimited waiting in the nodes is allowed, the dynamic fastest path problem can be solved using Dijkstra's algorithm just as efficiently as in the case of a static network. Stuart Dreyfus [21] is the first to demonstrate this. He redefines the cost function $d_{ij}(t_i)$, so that "if travel schedules are such that a delay before departure decreases the time of arrival, $d_{ij}(t_i)$ represents the elapsed time between time t and the earliest possible time of arrival." This alternative definition of $d_{ij}(t_i)$ results in a straight forward dynamic programming formulation of the problem, where DP state only stores the current location in the network. However, stopping at the waypoints is impractical or infeasible for many application. For example, an airplane cannot stop in mid-air to wait for a storm to pass by. Similarly, it is not practical for a large vessel to come to a complete stop before continuing the travel. Consequently, we do not allow stopping or waiting in our model. Instead, we extend Dreyfus' approach to a path finding model permitting voluntarily speed loss (i.e., slow down) along a path.

Similarly to Dreyfus, Halpern [29] discusses the problems where waiting at any given node is only permitted for a fixed set of time intervals specific to each node. In that case, it is not necessarily optimal to arrive at an intermediate node of a path as fast as possible. Halpern presents an algorithm that stores the possible arrival times at each node as they are discovered, analogous to an approach for the no-waiting-is-allowed case.

Orda and Rom [51] also study a third type of the fastest-path finding problems, which is not discussed elsewhere in the literature. They look at the problems where waiting is only allowed at the source node of the network. The authors show that if the cost function $d_{ij}(t_i)$ is continuous or piecewise continuous with only negative discontinuities (i.e., $\forall t_i \; d_{ij}(t_i^-) \geq d_{ij}(t_i^+)$) and such that for all t_i either $d_{ij}(t_i) = d_{ij}(t_i^-)$ or $d_{ij}(t_i) = d_{ij}(t_i^+)$, then every shortest topological path in the unrestricted waiting model is also a shortest topological path in the source waiting model, having the same total travel cost. Orda and Rom then provide an algorithm based on this notion that finds the optimal waiting time in the source node before starting the travel along an optimal path.

4.4.2 Dynamic Programming for a Time-Dependent Environment

The main difficulty of finding a fastest path in a time-dependent network comes from the fact that it is not necessarily optimal to arrive at each intermediate node of a path as fast as possible. Since the time to traverse between a pair of nodes from i to j (i.e., arc (i, j)) depends on when we start the travel, it might be beneficial to arrive to node i at a later time and observe a smaller travel time. In this section, we discuss in more details Dreyfus's method of reducing a time-dependent problem to a static network [21], and then adapt his approach to our problem.

Dreyfus analyzes a fastest-path finding problem in a time-dependent network where unlimited waiting is allowed in all nodes. Since in such problems, a mobile agent is permitted to wait at any given node until the optimal time to continue its travel, it is favorable to arrive at each node as fast as possible. Then, the optimal waiting time at each node is part of the decision a fastest path algorithm must make. It might appear that adding waiting time to the decision space of the model increases the complexity of the optimal path finding procedure, the reality turns out

to be otherwise.

Dreyfus redefines a traversing time function for an arc (i, j), such that when a delayed departure from node i decreases the time of arrival, the traverse time function represents the elapsed time between the time of arrival to node i and the corresponding earliest possible time of arrival to node j. With this alternative definition of the arc cost, the consistency condition defined by Kaufman and Smith [32] holds true. Therefore, we can formulate the dynamic programming functional equation without the time variable being present in the DP state. In this manner, the problem is reduced to the optimal path finding problem in a static network. Correspondingly, the optimal solution to a fastest-path finding problem with unrestricted wait consists of the ordered set of nodes and the optimal delay (or waiting time) at each of those nodes.

As we discussed earlier, waiting at the nodes is not permitted for the problems of interest. However, by allowing an agent to vary its speed, it is possible to slow down enough for the vehicle to arrive at node i at the optimal time to continue traversing the following arc (i, j). Thus, instead of arriving at node i as soon as possible and waiting until the optimal time to depart it, our mobile agent intentionally arrives at node i at the precise time of the optimal departure.

To illustrate this, assume that Dreyfus' method finds that it is optimal to leave node i at time t_i, travel along the arc (i, j) arriving at node j at time t_j, and then wait for some time w_j before continuing the travel. Then we choose to leave node i at the same time t_i but at a lower speed, such that we arrive to node j at the exact time equal to $t_j + w_j$, and then continue on without waiting at node j. In such a way, we use the solution found by Dreyfus' algorithm to construct an optimal path feasible for the problem.

When we implement Dreyfus' algorithm to find the optimal times to depart each waypoint, we use maximum attainable speed of the mobile agent to evaluate the arc cost. Then, we assume that it is always feasible to travel along an arc with any speed less than or equal to the maximum attainable speed. We also assume that there are no constraints on how quickly we can change the speed for any given arc. If this assumption is not valid, applying a slower speed for some arc might not allow an agent to speed up enough to reach the desired speed for the following arc of an optimal path. In our future work, we plan to integrate bounded acceleration and deceleration into the optimal path finding model.

The following Section 4.5 give a more detailed and rigorous implementation of the approach discussed in this section.

4.5 A Dynamic Programming Model and an Optimal Path Planning Algorithm

In the preceding sections we discuss how to address various aspects of an optimal path finding DP model: limited visibility horizon in Section 4.2, mobile system dynamics restrictions in Section 4.3, and computational demand of the time-dependent environment in Section 4.4. In this section, we integrate all the components discussed above into a single dynamic programming path finding model and provide an algorithm to facilitate its implementation.

Let $\tau(a, \theta_a, b, \theta_b, t_a)$ denote the travel time along a fastest path with bounded curvature from point a to point b starting with a heading angle θ_a at time t_a and arriving at b with a heading angle θ_b. We use the input parameters a and t_a to approximate the local medium (within a distance equal to l) as a stationary time and space homogeneous environment. As discussed in Section 4.3, we implement our earlier results for finding a fastest path with bounded curvature in an anisotropic

medium (see Algorithm 4 in Chapter III) to evaluate an optimal path from initial state (a, θ_a) at time t_a to the target state (b, θ_b), assuming that a mobile agent moves with its maximum attainable speed. The value of $\tau(a, \theta_a, b, \theta_b, t_a)$ is equal to the travel time associated with the found path.

Next, we define $T(a, \theta_a, b, \theta_b, t_a)$ to be the smallest elapse of time from the moment of arrival at point a at time t_a until arriving at the state (b, θ_b) (this definition is adapted from Dreyfus [21]). In other words, we allow an agent to leave point a at any time after arriving there at time t_a, with an objective to arrive at the state (b, θ_b) at the earliest possible time. Alternatively, we can define $T(.)$ as follows,

$$T(a, \theta_a, b, \theta_b, t_a) := \min_{\Delta_t \geq 0} \{\Delta_t + \tau(a, \theta_a, b, \theta_b, t_a + \Delta_t)\}. \tag{4.1}$$

It is important to note that in the minimization of equation (4.1), we have a natural upper bound on the value of Δ_t, which is equal to $\tau(a, \theta_a, b, \theta_b, t_a)$. Since $\tau(a, \theta_a, b, \theta_b, t_a)$ is the minimum travel time corresponding to an agent leaving point a without a delay, it is never advantageous to delay the departure longer than that time value. As a result, for each state of the system (e.g., (a, θ_a)) we only consider the departure times belonging to the interval $[t_a, t_a + \max_{(b,\theta_b)}\{\tau(a, \theta_a, b, \theta_b, t_a)\}]$. Alternatively, a traditional DP formulation of an optimal path finding problem in a dynamic network considers the departure times belonging to an interval $[t_a, T^*]$, where T^* is an upper bound on the agent's arrival time at the target point of the network (see [10, 15] for examples). Since $T^* >> t_a + \max_{(b,\theta_b)}\{\tau(a, \theta_a, b, \theta_b, t_a)\}$ for most states, our model significantly decreases the computational time of the path finding algorithm.

We now present our dynamic programming functional equation. For each state $(a, \theta_a) : a \in \mathbb{R}^2, \|a - s\| \leq R_H$ and $\theta_a \in [0, 2\pi]$, define the optimal value function $g(a, \theta_a)$, such that $g(a, \theta_a)$ is the minimum travel time over all the paths from the

initial position (s, θ_s) to point a, that start at time t_0 and arrive at a with the heading angle θ_a. Recall that $t_0 = 0$. Then, we have the following dynamic programming forward functional equation,

$$g(b, \theta_b) = \begin{cases} \min_{\{a, \theta_a : \|b-a\|=l\}} \{g(a, \theta_a) + T(a, \theta_a, b, \theta_b, g(a, \theta_a))\} \\ 0 \text{ for } (b, \theta_b) = (s, \theta_s) \end{cases} \tag{4.2}$$

A recursive application of the functional equation (4.2) delivers the fastest paths from (s, θ_s) to all the points in $\mathcal{N}_H$. To find these optimal paths, one can apply Dijkstra's algorithm [19], an A* algorithm [30] or any other efficient algorithms for a static network. However, we find the implementation of Dijkstra's method to be more advantageous since it automatically delivers the optimal paths to all the nodes in the network. Consequently, we obtain the fastest paths to all waypoints in $\mathcal{N}_H$ with one run of the dynamic programming algorithm.

Every optimal solution found using equation (4.2) includes an ordered set of states and an optimal wait time for each node along the path. However, our problem statement assumes that no stopping is permitted anywhere along the path. Thus, we use the found optimal wait times to calculate the speed for each arc, in order to ensure that a node arrival time is equal to the optimal node departure time.

To calculate the optimal speed, consider an arbitrary arc belonging to an optimal path that connects the states (a, θ_a) and (b, θ_b). We are also given an optimal departure time for node a, denoted by t'_a, and an optimal wait time at node b, denoted by w_b. Then, when an agent leaves state (a, θ_a) at time t'_a and travels with the maximum attainable speed along a fastest path with bounded curvature (see Algorithm 4 in Chapter III), it arrives at state (b, θ_b) at time $t'_a + \tau(a, \theta_a, b, \theta_b, t'_a)$. However, the optimal time to depart state (b, θ_b) is equal to $t'_a + \tau(a, \theta_a, b, \theta_b, t'_a) + w_b$. Consequently, we need to adjust the agent's travel time from (a, θ_a) to (b, θ_b) by a

factor $\rho_{(a,\theta_a),(b,\theta_b)} \geq 1$, where

$$\rho_{(a,\theta_a),(b,\theta_b)} = \frac{\tau(a,\theta_a,b,\theta_b,t'_a) + w_b}{\tau(a,\theta_a,b,\theta_b,t'_a)}. \tag{4.3}$$

An agent then adjusts the speed (by varying engine revolutions-per-minute, or other controllers) to a $1/\rho_{(a,\theta_a),(b,\theta_b)}$ fraction of the maximum attainable speed in order to arrive at state (b,θ_b) precisely at the time equal to $t'_a + \tau(a,\theta_a,b,\theta_b,t'_a) + w_b$.

Note that the minimum turning radius function defined in Chapter III is only dependent on the heading angle and does not explicitly depend on the agent's speed. Therefore, slowing down along an arc does not affect the curvature constraint and the feasibility of the originally constructed path. For some applications, this assumption might not be realistic. As we stated in Chapter III, we plan to relax this assumption in our future work. However, we would like to note that in most practical scenarios a slower speed of an agent results in smaller minimum turning radius, and does not restrict the feasibility of a path with the maximum attainable speed. In the case when a slow speed results in a larger minimum turning radius (e.g., our vessel routing project) the continuity of a feasible speed and system controllability (see Section 3.2.2) allows us to find a feasible path from (a,θ_a) to (b,θ_b) with the desired travel time for most practical applications.

The following Algorithm 5 summarizes our discussion and provides a concise step-by-step procedure of finding an optimal path from (s,θ_s) to (t,θ_t).

4.6 Application and Numerical Results

Algorithm 5 integrates all analysis and results presented in this dissertation into a single fastest-path finding model. In this section, we discuss the practical aspects of the algorithm implementation and demonstrate its application to the Optimal Vessel Performance in Evolving Nonlinear Wave-Fields (OVPENWF) project. While we

Algorithm 5 Fastest Path from (s, θ_s) to (t, θ_t).

Step 1. Apply Algorithm 4 to compute the values of $\tau(a, \theta_a, b, \theta_b, t_a)$ for all inputs where $\|a - s\| \leq R_H, \|b - s\| \leq R_H$ and $\|a - b\| = l$.

Step 2. Compute the smallest elapse of time function $T(a, \theta_a, b, \theta_b, t_a)$ using equation (4.1).

Step 3. Apply Dijkstra's algorithm to the DP recursive equation (4.2) to find the fastest paths from (s, θ_s) to all the points in $\mathcal{N}_H$.

Step 4. Depending on the specific problem, apply Algorithm 1, Algorithm 3 or Algorithm 4 to find the fastest paths from all points in $\mathcal{N}_H$ to the target state (t, θ_t).

Step 5. Find the point in $\mathcal{N}_H$ that has the smallest sum of the corresponding travel times found in Step 3 and Step 4. A fastest path passing through such point is the optimal path.

Step 6. For the optimal path found in Step 5, adjust the speed for each arc by a factor $1/\rho$ as described in equation (4.3).

restrict this section's discussion to the vessel routing problem, the implementation techniques used to further improve the efficiency of the algorithm can be applied to any optimal path finding problem.

One of the main objectives of our work is to produce a fast and computationally efficient path finding algorithm. Hence, a fast code implementing our algorithm is an important part of this goal. We have aimed to improve the run-time of our code to the greatest of our ability; however, it is important to note that an expert computer scientist should be able to improve the efficiency of the code even further. The main purpose of the program developed as part of this dissertation is to test the algorithm and the found optimal paths, which is successfully achieved.

4.6.1 Efficient Implementation

In the introduction section of this chapter we discuss that the computational time of the optimal path finding algorithm is essential for real-time implementation. We now present some coding-specific methods that ensure a time-efficient implementation of Algorithm 5.

One of the main techniques to speed up the run-time of the algorithm is to pre-

process as much of the algorithm as possible. Thus, we perform a portion of the calculations off-line, before the vessel starts its travel. In fact, a set of calculations need to only be evaluated once for a specific vessel, and the results are stored in the form of a look-up table as part of the navigation software.

Observe that the values of the travel time function $\tau(a, \theta_a, b, \theta_b, t_a)$ in Step 1 can be pre-computed for all possible stationary distributions of a local environment. Thus, we use the input parameters a and t_a to evaluate and then characterize the local medium (e.g., sea state). Because the distance between a pair of consecutive waypoints is fixed, for a given sea state, only values of θ_{ab}, θ_a and θ_b are used to look up the value of the $\tau()$ function in a pre-computed table.

Similarly to Step 1 of the algorithm, we can pre-compute Step 4 either for all possible global sea states or perform the calculations shortly before the start of a particular trip when the distribution of the global environment becomes available. Unlike evaluating the values of the $\tau()$ function in Step 1, the distance between points in $\mathcal{N}_H$ and t is not fixed, however the characterization of paths found in Step 4 stays the same regardless of the distance. This is especially true when we choose to implement Algorithm 1 or Algorithm 3, neglecting the minimum turning radius.

In addition, we partially pre-compute Step 2 of Algorithm 5, by evaluating when for some instances of the function $\tau(a, \theta_a, b, \theta_b, t_a)$ the delay is never beneficial. For example, for some sea states we observe the minimum value of the function, implying that any other departure time (possibly corresponding to a different distribution of the local medium) would never result in a smaller travel time making delay suboptimal. We also evaluate an upper bound on the possible decrease in travel time by comparing a given value of the $\tau()$ function to the best case scenario, which provides a tighter bound on the interval of Δ_t values considered in the minimization of

equation (4.1).

Another effective technique for decreasing the run-time of our algorithm is parallel computing, where a number of calculations are carried out simultaneously. With the rapid advancement of computing power, the parallelization of the algorithm is no longer restricted by the number of processors available to a user. While our test runs are performed on a single processor, there could be a potential run-time improvement if multiple processors are used for the implementation.

4.6.2 Computational Results

At the final stage of the Optimum Vessel Performance in Evolving Nonlinear Wave-Fields project, the components developed by research groups working on the project will be integrated into a single onboard navigation system to be validated during the forthcoming field tests. When the radar prototype and other elements of the project are completed, our path finding algorithm will be tested onboard an actual vessel traveling through the ocean waves. However, until the other project subtasks are accomplished, we test our algorithm in a simulated wave-field using the S-175 containership model, the same 175-meter-long vessel as discussed in Chapter II.

We randomly generate a wave-field region by specifying the wave distribution parameters corresponding to a desired sea state. The wave propagation model, developed by our colleague, Dr.Okey Nwogu, forecasts the evolution of the created wave-field over a time interval. Next, the evolving wave-field data is used to produce a 'local sea state' map for the location and time dependent environment. As a result, we obtain a lookup table that delivers the local sea state parameter for each waypoint and time instance considered by the dynamic programming algorithm.

The previously derived maximum attainable speed table provides the maximum

vessel speed that can be achieved for each local sea state and heading angle without violating the operability constraints (e.g., root mean squared roll and probability of wet deck). (See [20] for the detailed computation of the speed values and operability constraints.) Our colleagues, Dr. Jing Sun and Dr. Zhen Li, provided the minimum turning radius table specified for each value of local sea state and vessel heading angle. The speed and turning radius input tables are used to compute the $\tau()$ values for each pair of consecutive waypoints. That is, we create a lookup table that, for the given sea state and vessel heading angle, returns the next set of DP states (i.e., each waypoint location and heading angle) and the cost of traversing to each of those states. Note that for a specific vessel these calculations are done only once, and they do not contribute to the run-time of our path finding model.

Algorithm Run-Time

The main objective of the presented code is to facilitate integration of the path finding algorithm with the other components of the OVPENWF project. To ease the communication and compatibility of various parts of the project, we chose to implement our algorithm using MATLAB in order to stay consistent with our colleagues. MATLAB is considered to be more user friendly than other programming languages, and it is a preferred language in many engineering fields. However, it is important to note that C++ is well known to have a significantly faster run-time and experienced programmers report improvement by factors ranging between 30 and 50. For example, in order to speed up our code, we integrated a sorting function (the major component of Dijkstra's algorithm) written using C++ language into our MATLAB code. This modification decreased the sorting time of our program by a factor greater ten. We anticipate that all parts of the navigation system developed as a result of this project will be translated into C++ for implementation in real-life.

As a result, we expect the run-time of our algorithm to decrease significantly.

All the test runs are performed on a PC machine with the Microsoft Windows Vista operating system and a single 2.9 GHz processor. For each run we record the number of DP states that have to be 'explored' by the algorithm in order to find an optimal path. A dynamic programming state is considered to be 'explored' when the algorithm finds the minimum cost associated with reaching that state and updates the optimal value function $g(a, \theta_a)$ for the states that immediately follow it (i.e., its successors). Each of our test runs explored between 57,000 and 132,000 DP states with the corresponding run-time ranging linearly between 105 and 413 seconds. As mentioned earlier, the run-time is expected to decrease significantly once the algorithm is reprogrammed using C++. After the acceleration of the sorting function, the current program spends approximately 65% of the time performing other computations. Based on our experience with sorting, the conservative estimate is that we can decrease the time required to perform those computations by 10 times, resulting in the overall decrease in runtime by a factor of 2.5. In addition, faster processors are currently available on the market with the speed of up to 3.5 GHz, which would correspond to a decrease in the computational time by at least 15%.

Optimal Path

To evaluate the improvement in vessel travel time when Algorithm 5 is implemented, we test our model in the simulated wave-field corresponding to sea state number 6.5. The significant wave height (the mean wave height of the one third highest waves) for such sea state is equal to 7 meters, and the peak wave period is equal to 15 seconds. We choose this particular wave-field to illustrate some interesting scenarios of the path finding problem. Due to the large size of the vessel (175 meters long) and its limited maneuverability (the minimum turning radius values

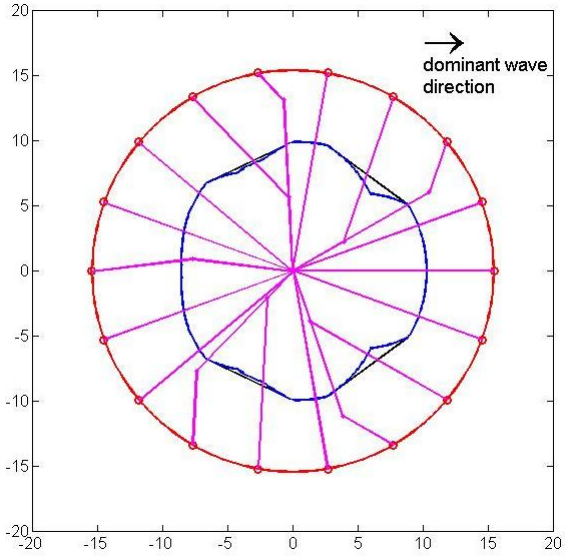

Figure 4.2: Linear path attainable region and the corresponding paths found using Algorithm 1 for sea state no. 6.5.

range between 290 and 305 meters), the smaller waves of lower sea states do not have as much impact on the vessel speed and do not always justify its maneuvers around the waves. The global sea state characterizing the waves beyond the radar visibility horizon is also set to 6.5 to ensure consistency when comparing the travel time for various paths. Figure 4.2 illustrates the linear path attainable region for sea state 6.5 that we use in our simulation, as well as the corresponding structure of the optimal paths in time and space homogeneous environment. We create a lookup table corresponding to the optimal paths for all target points equidistant from the start (the red circle on the figure). The paths and travel time are later scaled to fit a particular case.

One of the project objectives is to develop a radar that can accurately collect the information about the surrounding environment as far out as possible. The current conservative estimate of the radar visibility horizon is approximately 1.5 to 2 miles, equivalent to approximately 2,500 to 3,000 meters. While our colleagues' goal is to

increase the visibility to 5 miles or 8,000 meters, the wave-field region generated for our test runs corresponds to current radar capabilities (i.e., $R_H = 2500$ meter).

The predominant application of this project involves the short range trips. To reflect this fact, we consider a set of target points approximately 18,000 meters away from the current location. Considering the direction-dependent nature of our problem, the target points are positioned such that the directions to all the points span 360 degrees with a 20-degree increment. The distance between a pair of consecutive waypoints is set to 250 meters ($l = 250$), and we consider all consecutive states that can be reached from the predecessor without leaving the local sea state region. In Step 4 of Algorithm 5, we apply Algorithm 1 to compute the path travel time through the radar invisible region beyond R_H.

For each considered target point we run the MATLAB code to find an optimal path. The resulting travel time is then compared to that of the three alternative paths:

Suboptimal path 1 (p_1): a straight line path from starting point s to the target point t;

Suboptimal path 2 (p_2): a one-waypoint path from s to t found using Algorithm 1 where vessel makes a left turn at the waypoint; and

Suboptimal path 3 (p_3): a one-waypoint path from s to t found using Algorithm 1 where vessel makes a right turn at the waypoint.

It is also important to note that when the location of a target point corresponds to a convex part of the linear path attainable region (i.e., scenario 1 of Theorem 2.10), Algorithm 1 delivers a straight line path, and all suboptimal paths p_1, p_2 and p_3 are the same.

Results of Chapter II show that the one-waypoint paths p_2 and p_3 are optimal for the time and space homogeneous environment. Therefore, by comparing the optimal path to these suboptimal paths, we observe the travel time improvement one achieves by integrating the collected wave-field information into the path planning process. To make the comparison of travel times as accurate as possible, we set the initial heading of the vessel equal to the direction from the starting to the target point, denoted by θ_{st}. Nevertheless, since our dynamic programming model integrates the minimum turning radius constraint while neglecting it in the travel time calculations for paths p_2 and p_3, the actual travel time for the suboptimal paths has to be greater than our estimates, resulting in even more significant improvement in the minimum travel time than we report.

To compute the travel time for suboptimal paths p_1, p_2 and p_3, we use the wave-field data and find the local sea states a vessel traverses while following the specified path. This information is used to evaluate the actual travel time of the vessel within the visibility horizon. Analogous to finding the optimal travel time, the global sea state is used to evaluate the travel time for the remaining part of the path in the region beyond R_H.

Recall that in the time and space homogeneous environment, any path restricted to the two heading angles found in Algorithm 1 results in the same travel time. This is not the case when we take into account the available wave information inside the visibility horizon. Since p_2 and p_3 correspond to different paths (and travel times) within the visible region, the two cases with different orders in which the line segments of the one-waypoint path are implemented have to be considered separately.

Algorithm 1 is used to find the part of an optimal path outside the radar visibility horizon, as well as to evaluate the suboptimal paths p_2 and p_3; consequently, a large

part of these three paths is the same and has the same travel time. This is particularly true when the radar visibility horizon is small relative to the distance between the starting and target points. In such a case, the directions from all the points on the boundary of R_H to the target point t are very close to each other, implying that the invisible region part of the optimal path and of the alternative paths p_2 and p_3 consist of the linear segments with the same heading angles (see Figure 4.5). Due to time and space homogeneity of the region outside R_H, we find the 'merge points' x_2 and x_3 that lie in that region. These points are defined such that paths p_2 and p_3, respectively, differ from an optimal path between the start point and the merge point, and the suboptimal paths are the same as optimal path for the remaining part of travel.

In comparing the minimum trave time to suboptimal paths p_2 and p_3, we only use travel times of each path between the start point s and the corresponding merge points. Recall that as the vessel moves along a path, the new wave-field information is collected and an optimal path is reevaluated. Therefore, by the time a vessel reaches a merge point, the paths are reevaluated and we expect to see the same benefit of following a new optimal path for the next part of the path. The continuous reevaluation of an optimal path justifies the comparison of travel time only for the first part of paths leading up to the merge point.

Similar approach is used in comparing the optimal travel time when paths p_2 and p_3 correspond to a single line segment. When the heading angle of suboptimal straight line path is similar to that of the optimal path beyond R_H, we consider a closer target point equivalent to the merge point of a piecewise linear path. See Figure 4.3 and Figure 4.4 at the end of this section for illustration.

Table 4.1 summarizes the results of our test runs. The angle between starting

and target points is denoted by θ_{st} and the minimum travel time is represented by t^*. We let $t(p_i)$ for $i = 1, 2, 3$ denote the trave time along a path p_i from the starting point s to the corresponding merge point as discussed above. For consistency, it is implied that t^* is the travel time along an optimal path between the same pair of points (start and merge), when comparison is conducted. The values in the third column ($t^*/t(p_1)$) that are labeled by '*' correspond to the cases when the total travel time from s to t are compared, as considering a closer target point would alter the structure of an optimal path. In such cases, a large part of both paths are assumed to pass through the time and space homogeneous environment, and the values presented in the table are significantly lower than expected to be observed during the real-life implementation. The 'N/A' in Table 4.1 corresponds to the case when all three suboptimal paths are identical and we only report the run time improvement in the column corresponding to a straight line path.

Based on our analysis, we observe up to 9.7% improvement and on average between 4% and 6% improvement, in comparison to implementing Algorithm 1 while neglecting the wave-field data collected by the radar. The improvement reported here is expected to be significantly higher in the real-life applications, since the current models simulating the wave-field and vessel dynamics (e.g., speed) are very restrictive and do not fully capture the non-homogeneity of the system. In the following subsection, we discuss in details the limitations of the data available to us for the numerical test runs and their adverse effect on the reported travel time improvement.

Also note that the travel time decrease presented in Table 4.1 is a very conservative estimate, since our analysis neglects the bounded curvature for paths p_1, p_2 and p_3. In addition, the initial heading of the vessel is set to the direction θ_{st}, and the found optimal paths involve lesser vessel maneuvers than we would see for other starting

Run No.	θ_{st} (deg)	$t^*/t(p_1)$	$t^*/t(p_2)$	$t^*/t(p_3)$
1	0	0.9994	N/A	N/A
2	20	0.9418	N/A	N/A
3	40	0.9125	0.9476	0.9473
4	60	0.9846*	0.9334	0.9230
5	80	0.9456	N/A	N/A
6	100	0.9835*	0.9377	0.9684
7	120	0.9796*	0.9516	0.9605
8	140	0.9650	N/A	N/A
9	160	0.9033	N/A	N/A
10	180	0.9890	0.9601	0.9506
11	200	0.9307	N/A	N/A
12	220	1.0000	N/A	N/A
13	240	0.9634*	0.9488	0.9497
14	260	0.9822*	0.9479	0.9593
15	280	0.9823	N/A	N/A
16	300	0.9882*	0.9350	0.9495
17	320	0.9838*	0.9458	0.9526
18	340	0.9433	N/A	N/A

Table 4.1: Comparison of minimum travel time to the trave times for paths p_1, p_2 and p_3.

heading angle. However, it is important to remember that one of the advantages of our dynamic programming model is the integration of heading angle and turning radius constraints of the vessel into the path finding model. *Therefore, the presented path-finding model not only improves the travel time, but it also delivers a control-feasible path for any initial and target positions of the vessel.*

The following Figure 4.3, Figure 4.4, Figure 4.5 and Figure 4.6 illustrate examples of the found optimal paths and the alternative suboptimal paths.

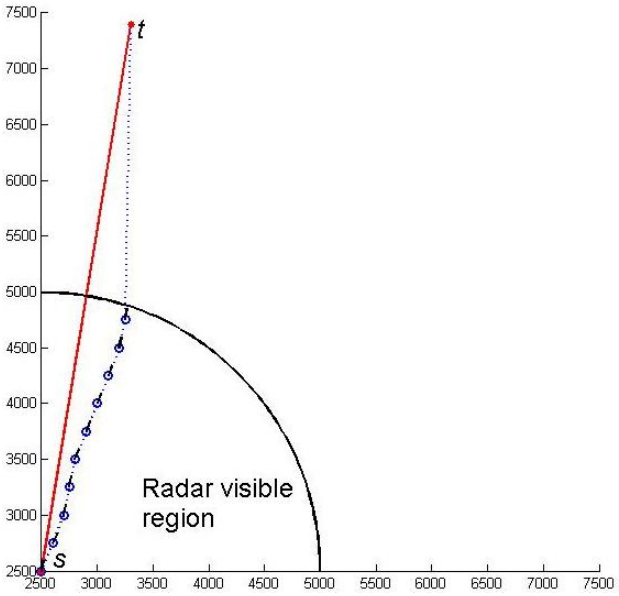

Figure 4.3: Test run number 5, where $\theta_{st} = 80$ degrees.

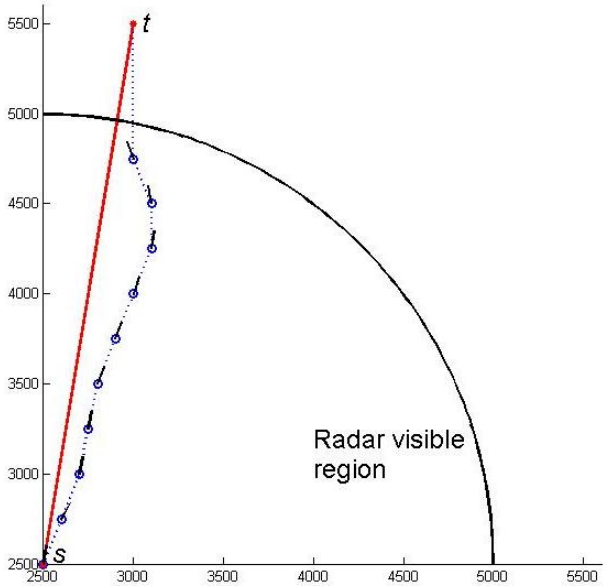

Figure 4.4: Test run number 5 with the target point relocated closer to point s.

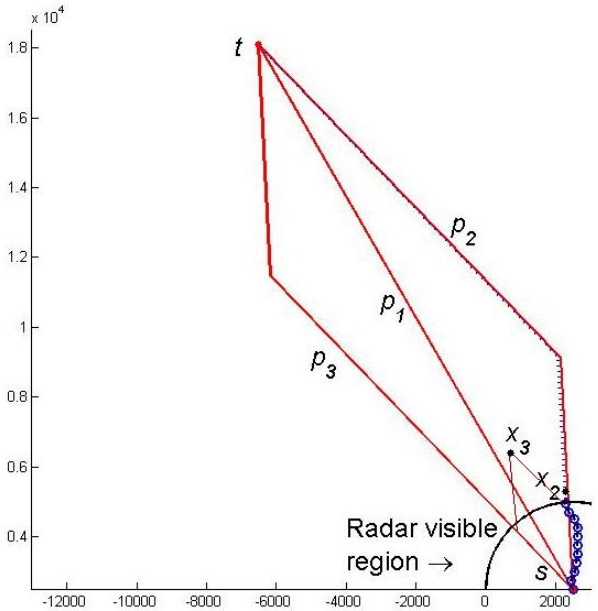

Figure 4.5: Test run number 7, where $\theta_{st} = 120$ degrees. The merge points for paths p_2 and p_3 are denoted by x_2 and x_3, respectively.

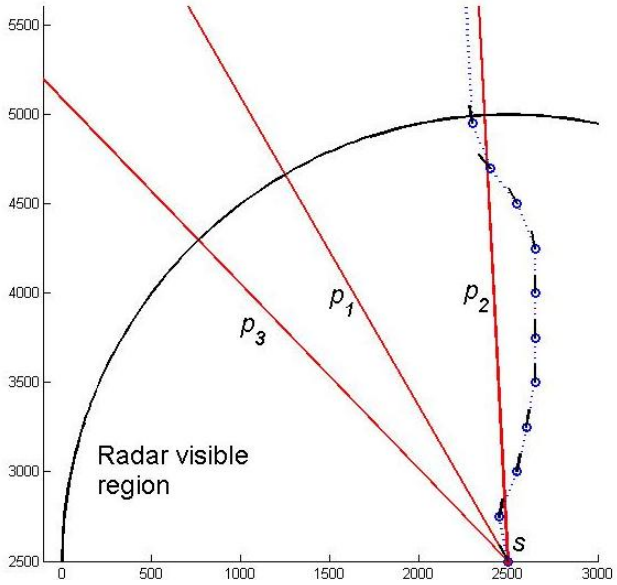

Figure 4.6: A detailed figure for test run number 7, illustrating the paths within the radar visible region.

Limitations of the Data Available for Numerical Analysis

The results of test runs summarized in Table 4.1 illustrate an improvement in the vessel travel time for 16 out of 18 cases, corresponding to a decrease ranging between 1.8% and 9.7%. However, due to the limitations of the data available to us for the numerical analysis, the reported benefits of our path finding model are extremely conservative, and we anticipate, in practice, to observe a significantly greater decrease in vessel travel time. In this section, we discuss the limitations of the data available for numerical results and their effects on our analysis.

The analysis is conducted for a large 175-meter long container ship that has relatively slow speed (ranging between 12 knots and 22.15 knots, or equivalently 6.2 - 11.4 meters per second) and a large turning radius (ranging between 290 and 305 meters). Since an average wave length for the considered sea state (No. 6.5) is equal to 90 meters, the vessel is not maneuverable enough to navigate around the large waves. Consequently, the presented path finding algorithm is not used to its fullest

potential. While all the gathered information about the surrounding wave-field is integrated into the path planning process, a significant amount of that information has no effect on the navigation for such a large vessel. For comparison, we scale the minimum turning radius of a vessel by a factor of 0.5 and recompute an optimal path for test run number 3 (i.e, $\theta_{st} = 40$ degrees). The minimum travel time for the new optimal path corresponds to the improvement of 14.6%, 9.5% and 15.4% when comparing to the travel times for paths p_1, p_2 and p_3, respectively, instead of 8.8%, 5.3% and 5.3% improvement for the original vessel as seen in Table 4.1. In addition to improved maneuverability, smaller vessels are also more suspectable to the effects of individual waves, further increasing the benefits of navigation around each wave.

In addition to limited maneuverability of the vessel, the model currently used to compute the added drag and corresponding vessel speed reduction is not as detailed and accurate as the model we anticipate to integrate upon the completion of this project. The maximum attainable speed function is currently averaged over a distribution of waves that a vessel is expected to observe for a specified sea state and heading angle. Therefore, it does not explicitly incorporate the effect an individual wave has on the vessel. We expect that a more intricate vessel motion model will result in the more complex optimal paths and the great time savings, especially so for the lower sea states.

To compare the minimum travel time to the alternative suboptimal paths, we use a simulation program to generate a realization of a time-evolving wave-field. The input to the program is a set of parameters characterizing the distributions of the waves, which results in the simulation generating a stationary wave-field. Subsequently, the large vessel turning radius and limitations of the vessel speed prediction model reduce our problem to optimal path finding in an ergodic system. This overall

'averaging' of the waves encountered by a vessel significantly limit an improvement in travel time observed by our model. The main advantage of real-time sensor data collection is that the environment does not have to be assumed or approximated by a stationary distribution. Therefore, in real-life applications, we expect to see significantly greater benefits of navigating around rough parts of the sea, and our path finding model would result in a greater decrease in travel time.

The estimates of travel time improvement presented in Table 4.1 are also greatly understated due to the fact that the minimum turning radius restriction is integrated into the optimal path, while the constraint is relaxed for the suboptimal paths p_1, p_2 and p_3. Integration of the turning constraint for the alternative paths would significantly increase their travel times and the corresponding improvement of the optimal path.

We would like to emphasize that the objective of the presented path finding code is to verify the algorithm validity and feasibility of the required run-time; and the reported benefits of our optimal path finding algorithm implementation are highly understated due to the very limited data available at this point of the project.

4.7 Optimal Path Finding for a Cost Function Other Than Travel Time

Throughout this and earlier chapters of the dissertation, we discuss an optimal path finding problems with the objective to minimize the agent's travel time. We often mention that our analysis and results can be directly extended to problems minimizing other cost functions. In many applications, we face an optimal path finding problem with alternative objective functions. For example in the case of the OVPENWF project, in addition to finding a fastest path, we are interested in minimizing root-mean-squared (RMS) motions and other measures of the path

'quality'. However, the extension of our analysis and presented path finding model is not straightforward for dynamic networks and path finding in a time-dependent environment. In this section we discuss how an optimal path finding algorithm changes when the objective function is other than travel time.

The problem of minimizing cost in a dynamic network is briefly discussed in the literature. In addition to the earlier discussed time-dependent fastest path problems, Chabini [10] looks at the minimum cost functions where travel time functions $d_{ij}(t_i)$ and cost functions $c_{ij}(t_i)$ are time-dependent. He extends backward DP formulation of the fastest path problems to this minimum cost path finding problem. Chabini assumes that $d_{ij}(t_i)$ and $c_{ij}(t_i)$ are constant for any time greater than some specified value, resulting in a static problem. This static problem solution is then used as the boundary condition for the dynamic programming formulation of the problem.

The difference between our earlier analysis of the fastest-path finding problems and modeling of a problem with a general cost function is that we can not eliminate the time variable from the dynamic programming state space (see Section 4.4 and equation (4.2)). Therefore, we have to set the DP state to be (a, θ_a, t_a) and consider all possible times of arrival at a given waypoint. Consequently, the resulting DP model delivers a classical functional equation and a straightforward application of Dijkstra's method or Chabini's approach can be used to find an optimal path. Since we do not present any significant improvement of the dynamic programming function equation for this set of problems, we will not go into any further details. We want to mention the difference between the models for the two types of problems.

We would also like to note that the cost function has to be additive to apply the standard dynamic programming recursive equation. However, the model can be adjusted to other objective functions. For example, the averaging measures of path

quality, like RMS motion, can be implemented by fixing a constant number of arcs for all considered feasible paths, or by adding a variable keeping record of the number of arcs traveled of the dynamic programming state space.

4.8 Conclusion

In this chapter, we relax the time and space homogeneous assumption of previous chapters and deliver an efficient model for finding an optimal path in a direction, location and time dependent environment. Our dynamic programming model integrates a number of aspects that are neglected by the traditional path finding DP models. We integrate the limitation of information available beyond a specified visibility horizon, by applying our results for a time and space homogeneous environment from the earlier chapters. To incorporate the dynamic constraints of the system, we employ our algorithm for optimal paths with bounded curvature and evaluate a feasible and optimal arc to traverse between a pair of DP states. Finally, by allowing a mobile agent to adjust its speed along a path, we are able to eliminate the time variable from the dynamic programming state space and significantly improve the run-time of our algorithm.

The developed Algorithm 5 is one of the key contributions of this dissertation. And while a number of individual results presented in the earlier chapters have significant stand-alone contributions to the various scientific areas, this chapter and Algorithm 5, in particular, bring together all the analysis and results presented throughout the dissertation into a single coherent optimal path finding model.

CHAPTER V

Conclusions

This dissertation discusses optimal path finding in a direction, location and time dependent environment. We deliver a computationally-efficient path finding algorithm with a sufficiently small run-time for real-time implementation. A traditional dynamic programming path finding model makes a number of restrictive assumptions that jeopardize its applicability to real-life problems. Alternatively, we present a model that integrates and addresses a set of limiting aspects previously neglected in the literature:

- Our dynamic programming (DP) path finding model integrates a limited visibility horizon and accounts for the lack of detailed information about a medium beyond a certain distance from the mobile agents current location.
- The presented DP model finds a smooth and control-feasible fastest path by integrating the systems dynamics into the optimization process.
- By integrating the agents controller (speed) into the decision space of the algorithm, the resulting model eliminates a time variable form the dynamic programming state space and improves efficiency and run-time of our model.

A number of special case problems corresponding to the assumption of a time and space homogeneous environment are solve analytically, and we present detailed

algorithms that facilitate the finding of closed-form solutions. These results deliver significant contribution to the studies of the anisotropic (direction-dependent) problems.

In future work, we plan to extend our results for a fastest path with bounded curvature to a very general set of direction-dependent speed functions, by relaxing convexity of a speed polar plot (Chapter III). By further analyzing the problem's properties, we also plan to dramatically simplify the proofs for the case of a convex linear path attainable region.

We are also interested in integrating the additional system constraints, such as bounded acceleration and deceleration, into the optimal path finding model in order to improve its accuracy and applicability. Finally, we plan to integrate uncertainty associated with data-collection and forecasting errors of the future environment. Our goal is to continue the study of integration of real-time data collection into the optimization models, especially with application to unmanned systems.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] J. M. Alden and R. L. Smith. Rolling horizon procedures in nonhomogeneous Markov decision processes. *Oper. Res.*, 40(suppl. 2):S183–S194, 1992.

[2] T. Allsopp, A. Mason, and A. Philpott. Optimal sailing routes with uncertain weather. In *Proceedings of The 35th Annual Conference of the Operational Research Society of New Zealand*, pages 65–74, December 2000.

[3] H. Alt and E. Welzl. Visibility graphs and obstacle-avoiding shortest paths. *Z. Oper. Res.*, 32(3-4):145–164, 1988.

[4] Applied Research Associates, Inc. Nighthawk micro air vehicle system description. Applied Research Associates: Robotic Systems, www.ara-robotics.com.

[5] R. Bellman. *Dynamic programming*. Princeton University Press, Princeton, N. J., 1957.

[6] J.-D. Boissonnat, A. Cérézo, and J. Leblond. A note on shortest paths in the plane subject to a constraint on the derivative of the curvature. Research Report RR-2160, INRIA, 1994.

[7] J.-D. Boissonnat, A. Cérézo, and J. Leblond. Shortest paths of bounded curvature in the plane. *Journal of Intelligent and Robotic Systems*, 11(1-2):5–20, 1994.

[8] A. E. Bryson, Jr. and Y. C. Ho. *Applied optimal control*. Hemisphere Publishing Corp. Washington, D. C., 1975. Optimization, estimation, and control, Revised printing.

[9] J. W. S. Cassels. *An introduction to the geometry of numbers*. Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen mit besonderer Berücksichtigung der Anwendungsgebiete, Bd. 99 Springer-Verlag, Berlin-Göttingen-Heidelberg, 1959.

[10] I. Chabini. Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Records*, 1645:170–175, 1998.

[11] S.-W. Cheng, H.-S. Na, A. Vigneron, and Y. Wang. Approximate shortest paths in anisotropic regions. *SIAM J. Comput.*, 38(3):802–824, 2008.

[12] L. P. Chew and R. L. S. Drysdale, III. Voronoi diagrams based on convex distance functions. In *SCG '85: Proceedings of the first annual symposium on Computational geometry*, pages 235–244, New York, NY, USA, 1985. ACM.

[13] H. Chitsaz and S. M. LaValle. Time-optimal paths for a dubins airplane. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 2379–2384, New Orleans, LA, December 2007.

[14] W. Collischonn and J. V. Pilar. A direction dependent least-cost-path algorithm for roads and canals. *Int. J. Geogr. Inf. Sci.*, 14(4):397–406, 2000.

[15] K. L. Cooke and E. Halsey. The shortest route through a network with time-dependent internodal transit times. *J. Math. Anal. Appl.*, 14:493–498, 1966.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT Press, Cambridge, MA, second edition, 2001.

[17] L. de Floriani, P. Magillo, and E. Puppo. Applications of computational geometry to geographic information systems. In *Handbook of computational geometry*, pages 333–388. North-Holland, Amsterdam, 2000.

[18] E. V. Denardo. *Dynamic programming*. Dover Publications Inc., Mineola, NY, 2003. Models and applications, Corrected reprint of the 1982 original.

[19] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959.

[20] I. S. Dolinskaya, M. Kotinis, M. G. Parsons, and R. L. Smith. Optimal short-range routing of vessels in a seaway. *Journal of Ship Research*, 53(2):1–9, June 2009.

[21] S. E. Dreyfus. An appraisal of some shortest-path algorithms. *Operations Research*, 17(3):395–412, 1969.

[22] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *Amer. J. Math.*, 79:497–516, 1957.

[23] O. M. Faltinsen. *Sea loads on ships and offshore structures / O.M. Faltinsen*. Cambridge University Press, Cambridge ; New York :, 1990.

[24] F. D. Faulkner. A general numerical method for determining optimum ship routes. *Navigation*, 10(2):143–148, 1963.

[25] F. D. Faulkner. Numerical methods for determining optimum ship routes. *Navigation: Journal of The Institute of Navigation*, 10(4):351–367, Winter 1963.

[26] A. F. Filippov. On certain questions in the theory of optimal control. *Journal of the Society for Industrial and Applied Mathematics, Series A: Control*, 1(1):76–84, 1962.

[27] S. Fortune and G. Wilfong. Planning constrained motion. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 445–459, New York, NY, USA, 1988. ACM.

[28] R. L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Inf. Process. Lett.*

[29] J. Halpern. Shortest route with time dependent length of edges and limited delay possibilities in nodes. *Z. Operations Res. Ser. A-B*, 21(3):A117–A124, 1977.

[30] P. E. Hart, N. J. Nilsson, and B. Raphael. Correction to "a formal basis for the heuristic determination of minimum cost paths". *SIGART Newsletter*, 37:28–29, 1972.

[31] S. Kapoor, S. N. Maheshwari, and J. S. B. Mitchell. An efficient algorithm for Euclidean shortest paths among polygonal obstacles in the plane. *Discrete Comput. Geom.*, 18(4):377–383, 1997.

[32] D. E. Kaufman and R. L. Smith. Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. *IVHS Journal*, 1(1):1–11, 1993.

[33] J. C. Kimball and H. Story. Fermat's principle, Huygens' principle, Hamilton's optics and sailing strategy. *European Journal of Physics*, 19:15–24, Jan. 1998.

[34] M. Lanthier, A. Maheshwari, and J.-R. Sack. Shortest anisotropic paths on terrains. In *Automata, languages and programming (Prague, 1999)*, volume 1644 of *Lecture Notes in Comput. Sci.*, pages 524–533. Springer, Berlin, 1999.

[35] J. P. Laumond, S. Sekhavat, and F. Lamiraux. Guidelines in nonholonomic motion planning for mobile robots. In *Robot motion planning and control*, volume 229 of *Lecture Notes in Control and Inform. Sci.*, pages 1–53. Springer, London, 1998.

[36] C.-Y. Lee and E. V. Denardo. Rolling planning horizons: Error bounds for the dynamic lot size model. *Mathematics of Operations Research*, 11(3):423–432, 1986.

[37] T. Lozano-Pérez and M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Commun. ACM*, 22(10):560–570, 1979.

[38] D. G. Luenberger. *Optimization by vector space methods*. John Wiley & Sons Inc., New York, 1969.

[39] W. Marks, T. R. Goodman, J. W. J. Pierson, L. J. Tick, and L. A. Vassilopoulos. An automated system for optimum ship routing. *Transactions - The Society of Naval Architects and Marine Engineers*, 76:22–55, 1968.

[40] T. G. McGee, S. Spry, and J. K. Hedrick. Optimal path planning in a constant wind with a bounded turning rate. In *Proceedings of the AIAA Conference on Guidance, Navigation and Control*, Ketstone, Colorado, August 2006.

[41] R. L. McNeely, R. V. Iver, and P. R. Chandler. Tour planning for an unmanned air vehicle under wind conditions. *Journal of Guidance, Control, and Dynamics*, 30(5):1299–1306, September-October 2007.

[42] J. S. B. Mitchell. A new algorithm for shortest paths among obstacles in the plane. *Ann. Math. Artificial Intelligence*, 3(1):83–105, 1991.

[43] J. S. B. Mitchell. L_1 shortest paths among polygonal obstacles in the plane. *Algorithmica*, 8(1):55–88, 1992.

[44] J. S. B. Mitchell. Shortest paths among obstacles in the plane. *Internat. J. Comput. Geom. Appl.*, 6(3):309–332, 1996.

[45] J. S. B. Mitchell. Geometric shortest paths and network optimization. In *Handbook of computational geometry*, pages 633–701. North-Holland, Amsterdam, 2000.

[46] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16(4):647–668, 1987.

[47] J. S. B. Mitchell and C. H. Papadimitriou. The weighted region problem: finding shortest paths through a weighted planar subdivision. *J. Assoc. Comput. Mach.*, 38(1):18–73, 1991.

[48] National Research Council of the National Academies. *Autonomous Vehicles in Support of Naval Operations*. The National Academic Press, Washington, DC, 2005.

[49] A. Nilim and L. E. Ghaoui. Algorithms for air traffic flow management under stochastic environments. In *Proceedings of American Control Conference*, volume 4, pages 3429–3434, July 2004.

[50] A. Nilim, L. E. Ghaoui, M. Hansen, and V. Duong. Trajectory-based air trafic management (tb-atm) under weather uncertainty. In *Proceedings of the Fourth International Air Traffic Management R&D Seminar ATM*, Santa Fe, New Mexico, December 2001.

[51] A. Orda and R. Rom. Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *J. Assoc. Comput. Mach.*, 37(3):607–625, 1990.

[52] J. Osborne and R. Rysdyk. Waypoint guidance for small UAVs in wind. In *Proceedings of the American Institute of Aeronautics and Astronautics Infotech@Aerospace Conference*, Arlington, VA, 2005.

[53] I. M. Ovacikt and R. Uzsoy. Rolling horizon algorithms for a single-machine dynamic scheduling problem with sequence-dependent setup times. *International Journal of Production Research*, 32(6):1243–1263, 1994.

[54] N. A. Papadakis and A. N. Perakis. Deterministic minimal time vessel routing. *Oper. Res.*, 38(3):426–438, 1990.

[55] A. N. Perakis and N. A. Papadakis. Minimal time vessel routing in a time-dependent environment. *Transportation Sci.*, 23(4):266–276, 1989.

[56] A. B. Philpott. Stochastic optimization and yacht racing. In *Applications of stochastic programming*, volume 5 of *MPS/SIAM Ser. Optim.*, pages 315–336. SIAM, Philadelphia, PA, 2005.

[57] A. B. Philpott and A. Mason. Optimising yacht routes under uncertainty. In *The 15th Cheasapeake Sailing Yacht Symposium*, 2001.

[58] A. B. Philpott, R. M. Sullivan, and P. S. Jackson. Yacht velocity prediction using mathematical programming. *European Journal of Operational Research*, 67(1):13–24, May 1993.

[59] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. *The mathematical theory of optimal processes*. Translated from the Russian by K. N. Trirogoff; edited by L. W. Neustadt. Interscience Publishers John Wiley & Sons, Inc. New York-London, 1962.

[60] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific J. Math.*, 145(2):367–393, 1990.

[61] J. H. Reif and Z. Sun. Movement planning in the presence of flows. *Algorithmica*, 39(2):127–153, 2004.

[62] N. C. Rowe. Obtaining optimal mobile-robot paths with nonsmooth anisotropic cost functions using qualitative-state reasoning. *The International Journal of Robotics Research*, 16(3):375–399, 1997.

[63] N. C. Rowe and R. S. Ross. Optimal grid-free path planning across arbitrarily contoured terrain with anisotropic friction and gravity effects. *IEEE Transactions on Robotics and Automation*, 6(5):540–553, October 1990.

[64] J. Sellen. Direction weighted shortest path planning. In *Proceedings of the International Conference on Robotics and Automation*, pages 1970–1975. IEEE Computer Society, 1995.

[65] P. Souères and J.-D. Boissonnat. Optimal trajectories for nonholonomic mobile robots. In *Robot motion planning and control*, volume 229 of *Lecture Notes in Control and Inform. Sci.*, pages 93–170. Springer, London, 1998.

[66] E. Stefanakis and M. Kavouras. On the determination of the optimum path in space. In A. U. Frank and W. Kuhn, editors, *Spatial Information Theory - A Theoretical Basis for GIS (COSIT'95)*, pages 241–257. Springer, Berlin, Heidelberg, 1995.

[67] Z. Sun and J. H. Reif. On finding approximate optimal paths in weighted regions. *J. Algorithms*, 58(1):1–32, 2006.

[68] Z. Sun and J. H. Rief. On finding energy-minimizing paths on terrains. *IEEE Transactions on Robotics*, 21(1):102–114, February 2005.

[69] H. J. Sussmann and G. Tang. Shortest path for the Reeds-Shepp car: A worked out example of the use of geometric techniques in nonlinear optimal control. Technical Report SYCON-91-10, Rutgers Center for Systems and Control, September 1991.

[70] US Department of Defence. *Office of the Secretary of Defence Unmanned Systems Roadmap (2007-2032)*. 2007.

[71] P. Widmayer, Y. F. Wu, and C. K. Wong. On some distance problems in fixed orientations. *SIAM J. Comput.*, 16(4):728–746, 1987.

[72] B. K. Youse. *Calculus with Analytical Geometry*. Holt, Rinehart and Winston, 1978.

[73] C. Yu, J. Lee, and M. J. Munro-Stasiuk. Extensions to least-cost path algorithms for roadway planning. *Int. J. Geogr. Inf. Sci.*, 17(4):361–376, 2003.

[74] E. Zermelo. Über das navigationsproblem bei ruhender oder veränderlicher windverteilung. *Zeitschrift für Angewandte Mathematik und Mechanik*, 11(2):114–124, 1931.