# Robotic Search Planning In Large Environments with Limited Computational Resources and Unreliable Communications

Benjamin A. Biggs

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

Daniel J. Stilwell, Chair

James W. McMahon

Craig A. Woolsey

Ryan K. Williams

Thinh T. Doan

February 3, 2023

Blacksburg, Virginia

Keywords: Informative Path Planning, Receding Horizon Path Planning, Greedy Algorithm, Non-Submodular Maximization

Copyright 2023, Benjamin A. Biggs

# Robotic Search Planning In Large Environments with Limited Computational Resources and Unreliable Communications

Benjamin A. Biggs

(ABSTRACT)

This work is inspired by robotic search applications where a robot or team of robots is equipped with sensors and tasked to autonomously acquire as much information as possible from a region of interest. To accomplish this task, robots must plan paths through the region of interest that maximize the effectiveness of the sensors they carry. Receding horizon path planning is a popular approach to addressing the computationally expensive task of planning long paths because it allows robotic agents with limited computational resources to iteratively construct a long path by solving for an optimal short path, traversing a portion of the short path, and repeating the process until a receding horizon path of the desired length has been constructed. However, receding horizon paths do not retain the optimality properties of the short paths from which they are constructed and may perform quite poorly in the context of achieving the robotic search objective. The primary contributions of this work address the worst-case performance of receding horizon paths by developing methods of using terminal rewards in the construction of receding horizon paths. We prove that the proposed methods of constructing receding horizon paths provide theoretical worst-case performance guarantees. Our result can be interpreted as ensuring that the receding horizon path performs no worse in expectation than a given sub-optimal search path. This result is especially practical for subsea applications where, due to use of side-scan sonar in search applications, search paths typically consist of parallel straight lines. Thus for subsea search applications, our approach ensures that expected performance is no worse than the usual subsea search path, and it might be much better.

The methods proposed in this work provide desirable lower-bound guarantees for a single robot as well as teams of robots. Significantly, we demonstrate that existing planning algorithms may be easily adapted to use our proposed methods. We present our theoretical guarantees in the context of subsea search applications and demonstrate the utility of our proposed methods through simulation experiments and field trials using real autonomous underwater vehicles (AUVs). We show that our worst-case guarantees may be achieved despite non-idealities such as sub-optimal short-paths used to construct the longer receding horizon path and unreliable communication in multi-agent planning. In addition to theoretical guarantees, An important contribution of this work is to describe specific implementation solutions needed to integrate and implement these ideas for real-time operation on AUVs.

# Robotic Search Planning In Large Environments with Limited Computational Resources and Unreliable Communications

Benjamin A. Biggs

(GENERAL AUDIENCE ABSTRACT)

This work is inspired by robotic search applications where a robot or team of robots is equipped with sensors and tasked to autonomously acquire as much information as possible from a region of interest. To accomplish this task, robots must plan paths through the region of interest that maximize the effectiveness of the sensors they carry. Receding horizon path planning is a popular approach to addressing the computationally expensive task of planning long paths because it allows robotic agents with limited computational resources to iteratively construct a long path by solving for an optimal short path, traversing a portion of the short path, and repeating the process until a receding horizon path of the desired length has been constructed. However, receding horizon paths do not retain the optimality properties of the short paths from which they are constructed and may perform quite poorly in the context of achieving the robotic search objective. The primary contributions of this work address the worst-case performance of receding horizon paths by developing methods of using terminal rewards in the construction of receding horizon paths. The methods proposed in this work provide desirable lower-bound guarantees for a single robot as well as teams of robots. We present our theoretical guarantees in the context of subsea search applications and demonstrate the utility of our proposed methods through simulation experiments and field trials using real autonomous underwater vehicles (AUVs). In addition to theoretical guarantees, An important contribution of this work is to describe specific implementation solutions needed to integrate and implement these ideas for real-time operation on AUVs.

# Dedication

*To my wife, Sydney, and our children, Jack, Maxwell, and Eleanor.*

# Acknowledgments

I am deeply grateful to my advisor, Dr. Stilwell, for the many hours that he has spent helping to guide my research and teaching me by word and deed. His constant encouragement and faith in my abilities has inspired me to grow academically and personally in wonderful ways that I never anticipated. I am also grateful to my committee members for their excellent teaching as well as their inspired questions.

I would like to thank the members of the Center for Marine Autonomy and Robotics. Their knowledge and experience been invaluable to me and their friendship has impacted every facet of my life in the best ways. The time that we've spent together is precious to me. I would like to give special thanks to Stephen Krauss, Chris Morency, and Charles Watson who have been with me since the beginning.

I am grateful to my parents, Bryon and Laura, for supporting me through thick and thin. I am especially grateful to my mother for home-schooling me and teaching me how to learn. I would also like to thank my siblings: Matthew, Liesle, Michael, and Devon. I am grateful to my children for helping me to stop and smell the roses. Their enthusiasm for life is contagious. Finally, I would like to express my deepest love and gratitude to my wife Sydney. She has made this adventure possible. The value of dedicating my dissertation to her and my children pales in comparison to the joy they have brought me. I suppose I will dedicate my whole life to them as well.

# Contents

**List of Figures** **xi**

**List of Tables** **xvi**

**1 Introduction** **1**

**2 Receding Horizon Planning: Background and Notation** **5**

2.1 General Search Problem Description . . . . . 6

2.2 Notation for Receding Horizon Paths . . . . . 7

**3 Performance Guarantees for Receding Horizon Search with Terminal Rewards** **9**

3.1 Introduction . . . . . 10

3.2 The Value-to-Go and Terminal Rewards . . . . . 12

3.3 Numerical Experiments . . . . . 18

3.3.1 Problem Formulation . . . . . 18

3.3.2 Results . . . . . 21

**4 Extended Performance Guarantees for Receding Horizon Search with Terminal Rewards** **27**

4.1 Sub-optimality Guarantees for Receding Horizon Path Planning with Terminal Rewards . . . 28

4.2 Numerical Experiments . . . 36

**5 Multi-agent Receding Horizon Search with Terminal Rewards 42**

5.1 Problem Description . . . 43

5.2 Sub-Optimality Guarantees for Receding Horizon Search with Terminal Rewards 44

5.3 Receding Horizon Search Path Planning with Sequentially Allocated Optimal Paths . . . 46

5.4 Numerical Experiments . . . 49

**6 Non-Submodular Maximization via the Greedy Algorithm and the Effects of Limited Information in Multi-Agent Execution 54**

6.1 Introduction . . . 55

6.2 Mathematical Preliminaries . . . 58

6.3 Properties of the Objective Function . . . 61

6.4 Bounds for the Greedy Algorithm Given a Non-Submodular Objective Function 63

6.4.1 Suboptimality of the Greedy Algorithm given a Non-submodular Objective Function Under a General Matroid Constraint . . . 63

6.4.2 Suboptimality of the Greedy Algorithm with Limited Information given a Non-submodular Objective Function . . . 65

6.5 The Benefit of Search Objective Function . . . 69

6.6 Properties of the Benefit of Search . . . . . 75

6.7 Approximation Guarantees with respect to A Submodular Approximate Objective Function . . . . . 78

6.8 Conclusions . . . . . 80

**7 Experiments in Underwater Feature Tracking with Performance Guarantees Using a Small AUV 81**

7.1 Introduction . . . . . 82

7.2 Gaussian Process Regression . . . . . 84

7.3 Objective Function . . . . . 86

7.4 Virginia Tech 690 AUV . . . . . 88

7.5 Field Trials . . . . . 90

7.5.1 Feasible Paths . . . . . 91

7.5.2 Terminal Rewards . . . . . 92

7.5.3 Computation of Reward Attained . . . . . 93

7.5.4 Receding Horizon Path Planning and Execution . . . . . 93

7.6 Results and Conclusions . . . . . 95

**8 Experiments in Collaborative Underwater Feature Tracking with Teams of Small AUVs 99**

8.1 Introduction . . . . . 99

8.2 Search Objective for Level Set Estimation . . . . . 102

8.2.1 Bayes' Risk and the Benefit Of Search . . . . . 103

8.2.2 Computing Expected Bayes' Risk . . . . . 105

8.3 Path Planning . . . . . 108

8.3.1 Augmented Receding Horizon Path Planning . . . . . 109

8.4 Online Gaussian Process Regression . . . . . 112

8.5 Field Trials . . . . . 113

8.5.1 Theory vs. Practice . . . . . 121

8.5.2 Results of Field Trials . . . . . 123

8.6 Proofs . . . . . 127

8.6.1 Proof of Lemma 8.2 . . . . . 127

8.6.2 Proof of Theorem 8.3 . . . . . 128

8.6.3 Derivation of $\mu^*$ . . . . . 130

8.6.4 Computing Expected Risk 1 . . . . . 131

8.6.5 Computing Expected Risk 2 . . . . . 134

8.7 Useful Identities . . . . . 137

8.7.1 Marginalization of Product of Gaussians . . . . . 138

**Bibliography** **150**

# List of Figures


3.1 The $61 \times 61$ gridded subsea search environment with cell color corresponding to the anticipated risk reduction for a single visit to each cell. . . . . 20

3.2 Example path . . . . . 22

3.3 Average value of RH paths constructed with $n = 2$ (top) and $n = 5$ (bottom) vs anticipated value of lawn mower path. . . . . 23

3.4 Standard RH path values with extended planning horizon vs anticipated lawn mower path values and average RH path values for RH paths constructed using terminal rewards. . . . . 24

3.5 Example of an RH path constructed without terminal rewards using parameters $n = 4$ and $l = 1000$. The path begins at the red end of the line (bottom left) and ends at the yellow end of the line. . . . . 25

3.6 Example of an RH path constructed with terminal rewards using parameters $n = 4$ and $l = 1000$. The path begins at the red end of the line (bottom left) and ends at the yellow end of the line. . . . . 26

4.1 150 x 150 cell sensor performance map of an environment measured in the approaches to Boston Harbor using side scan sonar on-board an unmanned underwater vehicle. The data corresponds to an environmental characterization metric which helps define how well the sonar will perform during search and characterization missions [53] . . . . . 39

4.2 Comparison of the resulting total accumulated reward for each of the planning approaches described in this paper where the mission length $l$ is varied. . . . 40

4.3 Results for the presented planners where the length $l$ is set to 5000. . . . . 41

4.4 Representative Search Path Using **MCTS-RH-TC** (a) and **MCTS-RH** (b) . The grayscale values indicate the reward for searching an area. The colored line represents the search path, the start at step 0 (dark blue) and the end at step 2500 (yellow). . . . . 41

5.1 $150 \times 150$ search grid representing the benefit of search given a single visit to each cell. . . . . 50

5.2 Cumulative reward and lower bounds for the path shown in Figure 5.3a . . . 51

5.3 Examples of receding horizon paths produced by 5 agents using Algorithm 1 with terminal rewards (a), dec-MCTS with terminal rewards (b), sequentally allocated optimal paths with no terminal rewards (c), and dec-MCTS with no terminal rewards (d). . . . . 52

5.4 Average path values for all path planners given differing numbers of search agents. . . . . 53

6.1 Average clique cover number of random graphs as a function of edge probability 70

6.2 The first row gives the normalized anticipated reward for $k$ visits to each cell. The second row gives the generalized curvature from Definition 6.4 for each cell assuming at most $k$ visits to each cell. The third row gives the inverse generalized curvature from Definition 6.5 for each cell assuming at most $k$ visits to each cell. All values are between 0 and 1 with darker regions indicating small values and lighter regions indicating large values. . . . . 74

7.1 690 vehicles being loaded for transportation to the operational area. . . . . 83

7.2 Simulated DVL beams shown in red. . . . . 89

7.3 Prior mean depth map with 15 meter isobath shown as a green line. . . . . 91

7.4 Claytor lake operational area. . . . . 96

7.5 The posterior predictive depth map within the bounded search region given the measurements acquired while executing the receding horizon path shown in red produced using terminal rewards (a) and the receding horizon path produced without the addition of terminal rewards (b). The 15 meter isobath is shown in green. . . . . 97

7.6 Cumulative rewards for the RH path and RH path constructed with terminal rewards are shown in dashed blue and solid green respectively. The cumulative reward for the RH path constructed with terminal rewards plus the lower bound on the value-to-go is shown as the solid red line. The guaranteed lower bound is shown as the solid black line. . . . . 98

8.1 690 AUVs . . . . . 115

8.2 The mission start waypoint and local tangent frame origin are shown as the green circle marked with the number 1. The boundaries of the operational area are shown by the white lines with the number 2 in the circles where boundary lines intersect. The line defining the deepest portion of the prior mean is shown as the blue line with the number 3 in blue circles at each end of the line. The values shown next to the lines give the length of each line segment. . . . . 116

8.3 Visual representation of discrete dynamics. Initial state is shown as the red asterisk with attached arrow. Minimum turning circles are shown at either side of the initial state. Each blue asterisk with its attached arrow represents the result of commanding a change in heading from the initial state. . . . . 118

8.4 Short horizon paths with appended lawn mower paths. Circles indicates the path start locations. . . . . 120

8.5 Panels (a), (b), and (c) show the difference between the posterior Bayes' risk given the measurements available to agents 1, 2, and 3 respectively and the posterior Bayes' risk given all measurements shown in Figure 8.8a. Note that all measurements not obtained by the agent itself must be received via acoustic communications. . . . . 123

8.6 Paths and Bayes' risk for experiments accomplished using 2 vehicles. Panels (a) and (b) show the prior Bayes' risk while panels (c) and (d) show the posterior Bayes' risk. . . . . 125

8.7 Results of receding horizon missions conducted with 2 vehicles. . . . . 125

8.8 Paths and Bayes' risk for experiments accomplished using 3 vehicles. Panels (a) and (b) show the prior Bayes' risk while panels (c) and (d) show the posterior Bayes' risk. . . . . 126

8.9 Results of receding horizon missions conducted with 3 vehicles. . . . . 126

# List of Tables

6.1 Maximum values of $\alpha$ and $\beta$ with respect to the maximum number of allowed search passes $k$. . . . . 78

7.1 Specifications of the Virginia Tech 690 AUV. . . . . 88

# Chapter 1

# Introduction

This work is inspired by robotic search applications where a robot or team of robots is equipped with sensors and tasked to autonomously acquire as much information as possible from a region of interest. To accomplish this task, robots must plan paths through the region of interest that maximize the effectiveness of the sensors they carry. The computational complexity of planning paths that maximize information gain can be prohibitive for robots (agents) with limited computational resources. *Receding horizon path planning* allows agents to iteratively construct a long path by planning a short optimal path and traversing a portion of the short path. This process is repeated until the mission is complete. However, only considering short paths can lead agents to devote excessive attention to local maxima, which can which can cause poor performance in the overall information gathering mission. Additionally, when considering a team of agents, the computational burden of planning even short paths is compounded exponentially as compared to the complexity of path planning for a single agent. One method of planning for a team of agents that reduces computational complexity is to impose an ordering on the team of agents and, in sequence, have each agent plan a path for itself considering only the paths planned by the preceding agents. Theoretical guarantees on the performance of paths produced using sequential planning methods generally rely on the objective function that the agents seek to maximize being *submodular* or possessing a diminishing returns property. We demonstrate, in Chapter 6, that the robotic search objective function that we consider throughout this work is non-submodular. Also,

theoretical guarantees on the performance of paths produced using this sequential planning method generally assume ideal communication between agents. Contributions of this work include the development of methods of constructing receding horizon paths that provide formal guarantees on the worst-case performance of the paths produced and the provision of theoretical worst-case guarantees for paths produced using sequential planning where the objective function is non-submodular and communication is potentially unreliable.

Receding horizon methods generally work well in practice and allow search agents to adapt plans to new information as it is acquired. However, short planning horizons make receding horizon paths susceptible to local maxima which can cause poor performance in the overall mission [77]. That is, although paths produced using receding horizon methods tend to perform well, there is generally no assurance that receding horizon paths will perform better than the worst path possible. The contributions of this work towards addressing the worst-case performance of paths produced using receding horizon methods are presented in Chapters 3 through 5. Specifically, in Chapter 3, we consider a single agent performing a search mission in a receding horizon manner. We assume that planning at each step is optimal and a single action is executed before replanning. We present a novel method of incorporating a terminal reward in the objective function that the agent seeks to maximize at each planning step. We prove that the addition of an appropriate terminal reward provides theoretical lower bound guarantees on the value of the receding horizon path executed by the agent. In Chapter 4, we extend the results of Chapter 3 by addressing the case that the search agent plans paths at each planning step that are not optimal but near-optimal. We consider that the search agent may need to execute several actions between planning steps to allow for planning methods such as Monte Carlo tree search that benefit from increased planning time. In Chapter 5, we extend the results of Chapter 4 to consider a team of search agents. We present a sequential planning algorithm that leverages our theoretical results to

provide a guaranteed lower bound on the value of joint paths produced. In each chapter, we provide numerical experiments that validate our theoretical contributions.

In robotics, having a team of agents sequentially plan paths for themselves while considering only the paths of preceding agents is sometimes referred to as *sequential-allocation* [64] or *sequential greedy assignment* [16]. More broadly, sequential decision making algorithms of this type may be represented as the well-known *greedy algorithm*[55] under specific constraints[18]. Two significant attributes of the greedy algorithm are that it allows for distributed planning and, in many cases, provides theoretical guarantees on the quality of the solutions produced [15, 23, 55]. However, most theoretical guarantees for the greedy algorithm require that the objective function be submodular [15, 18, 23, 31, 55, 64]. This submodularity requirement is limiting since many useful objective functions for applications such as experimental design, dictionary selection, subset selection [49], sparse Gaussian processes [44], and visibility optimization in social media [28] are non-submodular. We demonstrate, in Chapter 6, that the robotic search objective function that we consider throughout this work is non-submodular. Recent work has begun to provide theoretical guarantees for the greedy algorithm in maximizing non-submodular objective functions [3, 28, 57]. However, these recent works do not consider factors relevent to robotic search applications such as suboptimal decision making [18, 64] and limited information available to search agents which can be caused by unreliable communications. The contributions of this work towards addressing the lack of theoretical guarantees for the greedy algorithm in robotic search applications are presented in Chapter 6. Specifically, we utilize notions of curvature of the objective function to provide novel guarantees for the greedy algorithm with ideal communication and with limited information due to unreliable communication. In each case, our novel guarantees account for the case that sequential planning decisions are not optimal, but near-optimal as in [64]. Additionally, we prove that guarantees for sequential planning may be obtained with respect to the value

of an optimal solution for an approximate objective function.

In Chapters 7 and 8, we demonstrate the utility of the planning methods proposed in Chapters 3 through 5 by addressing the problem of autonomously localizing a depth contour or isobath within a bounded area of Claytor lake in Dublin, VA, USA. Determining the location of an isobath is an important element of quickly finding navigable paths in waterways. In Chapter 7, we utilize an objective function based on classification ambiguity [11, 30] and perform experiments using a single AUV. In Chapter 8, we present a decision theoretic objective function and perform experiments using teams of 2 or 3 AUVs. Throughout the experiments presented in Chapters 7 and 8, we integrate several recent developments for real-time isobath localization, all of which can be easily modified for similar real-time mapping applications: (1) use of a sparse Gaussian process for representing the bathymetry, which leads to a continuous representation of the environment along with a rigorous assessment of uncertainty, and (2) use of receding horizon methods proposed in Chapters 3 through 5 which ensure that the cumulative reward attained over the course of a mission is guaranteed to achieve a desired lower bound. An important contribution of this work is to describe specific implementation solutions needed to integrate and implement these ideas for real-time operation on AUVs.

# Chapter 2

# Receding Horizon Planning: Background and Notation

We consider the challenge of planning paths for search missions where the length of an optimal search path is too long to be computed in real-time. We do not address exhaustive search as is normally a primary goal for coverage path planning algorithms [27]. Rather, we seek solutions for time-limited or resource-limited search scenarios where not all locations are necessarily searched. We consider that optimal paths maximize a general deterministic value function. In the context of search, a sensor may perform well in some areas and poorly in others. Thus, we seek to plan search paths that maximize search effectiveness. Naturally, the performance of a search mission is stochastic; however, the anticipated performance used for planning search missions is deterministic.

The problem of computing search paths with a discrete set of possible actions has been shown to be NP-hard [42]. Receding horizon path planning addresses the computational complexity of planning long paths. Receding horizon path planning derives from receding horizon control, also known as model predictive control [47, 50, 51]. Receding horizon control solves for an optimal control law within a finite planning horizon and executes a portion of that optimal solution out to a finite execution horizon that is shorter than the planning horizon. At the end of the execution horizon, a new optimal solution is computed, and the process is repeated until the mission is completed i.e. until the allotted time or resources

are exhausted or the desired outcome is achieved. Likewise, a receding horizon path is constructed iteratively by solving for a short optimal or near-optimal path, traversing that path out to some execution horizon, and repeating the process of planning and executing until the search agent has traversed a path of the desired length. The planning horizon is generally chosen to accommodate time or computational resource constraints of search agents solving for optimal or near-optimal short paths.

## 2.1 General Search Problem Description

We consider a search region $\mathcal{G} \subset \mathbb{R}^2$. An $n$-length path through the search region beginning at the initial state $p_0 \in \mathcal{G}$ is denoted $\gamma_n(p_0)$ and is composed of a sequence of locations $\{p_0, p_1, \ldots, p_n\}$. A sequence is feasible if it can be traversed by a mobile search agent. Explicitly, we consider that the state of a mobile search agent at time $t$ may be represented by the tuple $s_t = \{p_t, x_t\}$ where $x_t$ gives all states of the mobile search agent excluding the position states $p_t$. We say that a state $s_t$ is feasible from the state $s_{t-1}$ if there exists an action $a$ available to the search agent such that applying $a$ to the state $s_{t-1}$ results in the new state $s_t$. An $n$-length path may then be represented as a sequence of states $\gamma_n(s_0) = \{s_0, s_1, \ldots, s_n\}$. The set of all feasible $n$-length paths beginning at state $s_0$ is denoted $\Gamma_n(s_0)$.

The value of a path is given by

$$J(\gamma_n(s_0)) = \sum_{i=1}^{n} g(s_i, s_{i-1}, \ldots, s_0) \tag{2.1}$$

where $g$ is a non-negative function that returns the anticipated value of the information gained while transitioning from one state to the next. We assume that the value of informa-

tion gained is dependent on previously visited locations. That is, $g(s_i, s_{i-1}, \ldots, s_0)$ returns the value of transitioning from state $s_{i-1}$ to $s_i$ given all states or locations where sensor measurements have been collected. In essence, if a location has been searched before then the reward for searching the location again will likely be discounted.

In the context of search, the goal is to plan a path that maximizes the amount of information gained while traversing the path. The optimal path with respect to (2.1) is

$$\gamma_n^*(s_0) = \underset{\gamma_n(s_0)\in\Gamma_n(s_0)}{\arg\max}\ J(\gamma_n(s_0)) \tag{2.2}$$

with value

$$J(\gamma_n^*(s_0)) = \max_{\gamma_n(s_0)\in\Gamma_n(s_0)} J(\gamma_n(s_0)). \tag{2.3}$$

For notational simplicity we define

$$g(s_i) \triangleq g(s_i, s_{i-1}, \ldots, s_0) \tag{2.4}$$

and

$$J_n(s_0) \triangleq J(\gamma_n(s_0)) \tag{2.5}$$

## 2.2 Notation for Receding Horizon Paths

Receding horizon paths are constructed from a sequence of $n$-length optimal or near-optimal paths. For this discussion, we presume that a search agent is able to solve for optimal $n$-length paths; and a new, $n$-length, optimal path is computed after traversing a single step along the previously computed $n$-length optimal path or, in other words, the execution

horizon is one step. The $r$-length RH path is denoted

$$\boldsymbol{P}_r = \{s_0, s_1, \ldots, s_r\} \tag{2.6}$$

Because the RH path is composed of a sequence of single steps that come from optimal $n$-length paths,

$$\{s_j, s_{j+1}\} \subset \boldsymbol{P}_r \tag{2.7}$$

are the first two elements of the optimal path $\gamma_n^*(s_j)$ for all $j + 1 \leq r$.

To clarify, the construction of a RH path proceeds as follows. Beginning at the initial state $s_0$, an optimal path $\gamma_n^*(s_0)$ is computed with the planning horizon $n$. The search agent traverses the first step of the path $\gamma_n^*(s_0)$. Thus the first 2 elements of the RH path are $\boldsymbol{P}_1 = \{s_0, s_1\} \subset \gamma_n^*(s_0)$. The optimal path $\gamma_n^*(s_1)$ is computed, and the process repeats until the mission is completed.

The value of a receding horizon path $\boldsymbol{P}_r$ is

$$J(\boldsymbol{P}_r) = \sum_{i=1}^{r} g(s_i) \tag{2.8}$$

where $g$ is the non-negative function in (2.1).

It should be noted that, despite being composed of steps along optimal paths, a receding horizon path does not necessarily retain optimality properties. In fact, myopic planning horizons can lead agents to make poor decisions in the scope of a search mission by overemphasizing immediate rewards leading to poor overall performance.

# Chapter 3

# Performance Guarantees for Receding Horizon Search with Terminal Rewards

We present a novel method of using terminal rewards in the construction of a receding horizon search path. We prove that the proposed method of constructing search paths provides a theoretical lower bound on the performance of the search path. Our result can be interpreted as ensuring that the receding horizon path performs no worse in expectation than a given sub-optimal search path. This result is especially practical for subsea applications where, due to use of side-scan sonar in search applications, search paths typically consist of parallel straight lines. Thus for subsea search applications, our approach ensures that expected performance is no worse than the usual subsea search path, and it might be much better. We demonstrate the efficacy of the proposed method by planning search paths in simulation using real-world data that was acquired by an autonomous underwater vehicle during a subsea survey of Boston Harbor.

## 3.1 Introduction

The main contribution of this chapter is a method of incorporating terminal rewards into a receding horizon search path planner such that the resultant receding horizon search paths are theoretically guaranteed to perform at least as well as any path that is already available. For a subsea search mission, a typical search path is a sequence of parallel straight lines called a "lawn mower" or "mowing the lawn" path. In this case, our approach is guaranteed to perform no worse than a mowing the lawn path in expectation, and might perform much better. The methods proposed in this chapter and in Chapter 3 are inspired by works in receding horizon control such as [19, 37, 58] and [48].

One of the challenges for path planning using receding horizon (RH) methods occurs when an agent is unable to effectively construct desirable paths because of local extrema in the search space that arise due to myopic planning horizons [73]. Yoo et al. [77] propose a method to address the deficiencies of RH planning by extending the planning horizon until the expected reward satisfies a given threshold such that the resultant path jumps from thoroughly explored areas to unexplored areas that possess high reward. In [4, 46, 54] an approximation of the minimum feasible arrival time to a goal location is used as a terminal cost to help the controller generate feasible paths in complex environments and avoid becoming trapped in concave obstacles. The approaches presented in [4, 46, 77] and [54], are shown to be effective using numerical experiments; however, no formal performance guarantee is provided. We note that it is shown in [54] that a terminal reward is a bound on the reward of the receding horizon trajectory plus the terminal reward at the next iteration. Our approach, which builds upon the same insight, provides a guaranteed lower bound on expected search performance for an entire mission.

The receding horizon approach described in [20] is similar to ours, except that in [20] a

Lyapunov-like constraint is appended to the optimization problem, and at each location in the workspace, there must be a next step for which the Lyapunov-like function decreases. We achieve a similar result using a terminal reward instead of a constraint. The terminal reward is required to converge to zero over the sequence of RH optimization solutions, but it is not required to converge monotonically. It is proven in [20] that a terminal constraint can be used to guarantee convergence to a desired state. In contrast, the method proposed in this chapter uses sub-optimal paths to determine terminal rewards for receding horizon search paths. It is proven that these sub-optimal paths provide a lower bound to the performance of the corresponding RH path.

We note that the performance guarantees provided in this chapter do not imply optimality, they simply guarantee a lower bound on performance. In other words, the method ensures that a receding-horizon path does not behave poorly. A limitation of our approach is the existence of suitable terminal rewards. For subsea search applications, terminal rewards can be derived from typical sub-optimal search paths.

This Chapter is organized in the following manner. The use of terminal rewards in the construction of a receding horizon search path and performance guarantees are discussed in Section 3.2. We demonstrate in simulation, using real-world data, that the proposed method not only provides a lower bound on the performance of the receding horizon search path, but that the search paths produced using the proposed method perform significantly better than this theoretical lower bound in most cases. In Section 3.3 we also demonstrate that search paths constructed using the proposed method are competitive with receding horizon search paths generated with much longer planning horizons.

## 3.2 The Value-to-Go and Terminal Rewards

To address the problem of poor performance caused by myopic planning horizons, we consider a path of length $l \gg n$. By equation (2.1), the reward associated with the $l$-length path is

$$J_l(s_0) = \sum_{i=1}^{l} g(s_i). \tag{3.1}$$

We note that the summation on the right-hand side of (3.1) may be separated into 2 parts and written as

$$J_l(s_0) = \sum_{i=1}^{n} g(s_i) + \sum_{j=n+1}^{l} g(s_j). \tag{3.2}$$

We call the right-most summation in (3.2) the *value-to-go*. Notice that the first summation on the right-hand side of (3.2) represents the value of the portion of a path that can be computed optimally and the value-to-go represents the value of the remainder of the path that is normally ignored when planning a receding horizon path. Therefore, the value of the optimal $l$-length path may be represented as

$$J_l^*(s_0) = \max_{\gamma_n(s_0) \in \Gamma_n(s_0)} \left( J_n(s_0) + J_{l-n}^*(s_n) \right). \tag{3.3}$$

We specifically assume that the value-to-go represents the maximum achievable value of the right-most summation in (3.2) as in (3.3).

We assume that the search agent has only sufficient computational resources to compute $n$-length optimal paths. Thus, we assume that there exists a lower bound on the value-to-go.

**Definition 3.1.** A lower bound on the value-to-go in 3.2 is the function $B_l(s_n)$ that satisfies

$$B_l(s_n) \leq J_{l-n}^*(s_n) \tag{3.4}$$

for all $n \leq l$ with $B_l(s_l) = 0$, and for the infinite case where $l = \infty$, $B_l(s_r) \to 0$ as $r \to \infty$.

We assume that the lower bound on the value-to-go may be efficiently computed in real-time. Using this lower bound on the value-to-go, we define the value of an $n$-length path as

$$\mathcal{V}(\gamma_n(s_0)) = J_n(s_0) + B_l(s_n) \tag{3.5}$$

For notational simplicity we define

$$\mathcal{V}_n(s_0) \triangleq \mathcal{V}(\gamma_n(s_0)) \tag{3.6}$$

An optimal path with respect to (3.5) is now is now defined as

$$\gamma_n^*(s_0) = \underset{\gamma_n(s_0) \in \Gamma_n(s_0)}{\arg\max} \ \mathcal{V}_n(s_0) \tag{3.7}$$

with value

$$\mathcal{V}_n^*(s_0) = \max_{\gamma_n(s_0) \in \Gamma_n(s_0)} \mathcal{V}_n(s_0) \tag{3.8}$$

Notice the similarity between (3.5) and (3.3). The intuition behind (3.5) is the following. If the lower bound on the value-to-go may be found efficiently and approximates the value of the optimal value-to-go, then the value of an $n$-length path with an appended terminal reward approximates the value of the optimal $l$-length path. In practice, it may be difficult to efficiently approximate the value-to-go. As such, we do not seek to plan optimal paths, but instead we seek to provide a lower bound on the value of receding horizon search paths that approximates the value of the optimal $l$-length path as nearly as is feasible.

**Proposition 3.2.** *Suppose the $l$-length receding horizon path $\boldsymbol{P}_l$ is composed of 1-step sequences along a sequence of $n$-length optimal paths that maximize (3.5). For every $n$-length*

*optimal path*

$$\gamma_n^*(s_r) = \{s_r, s_{r+1}^*, \ldots, s_{r+n}^*\} \tag{3.9}$$

*suppose that there exists a location* $\tilde{s}_{r+n}$ *that is feasible from* $s_{r+n-1}^*$ *for which*

$$g(\tilde{s}_{r+n}) + B_l(\tilde{s}_{r+n}) \geq B_l(s_{r+n-1}^*) \tag{3.10}$$

*then the RH path satisfies*

$$J(\boldsymbol{P}_l) \geq B_l(s_0). \tag{3.11}$$

*Proof.* For any location $\tilde{s}_r$ along the RH path, the path that maximizes (3.5) is

$$\gamma_n^*(\tilde{s}_r) = \{\tilde{s}_r, s_{r+1}^*, \ldots, s_{r+n}^*\} \tag{3.12}$$

and the optimal $(n-1)$-length path that starts at $s_{r+1}^*$ is

$$\gamma_{n-1}^*(s_{r+1}^*) = \{s_{r+1}^*, s_{r+2}^*, \ldots, s_{r+n}^*\} \tag{3.13}$$

Suppose $\tilde{\gamma}_n(\tilde{s}_r)$ is an $n$-length path composed of $\gamma_{n-1}^*(\tilde{s}_r)$ and a location $\tilde{s}_{r+n}$ that is feasible from the final location $s_{r+n-1}^*$ along $\gamma_{n-1}^*(\tilde{s}_r)$ such that

$$\tilde{\gamma}_n(\tilde{s}_r) = \{\gamma_{n-1}^*(\tilde{s}_r), \tilde{s}_{r+n}\} \tag{3.14}$$

The value of $\tilde{\gamma}_n(\tilde{s}_r)$ is

$$\mathcal{V}(\tilde{\gamma}_n(\tilde{s}_r)) = \mathcal{V}_{n-1}^*(\tilde{s}_r) - B_l(s_{r+n-1}^*) + g(\tilde{s}_{r+n}) + B_l(\tilde{s}_{r+n}) \tag{3.15}$$

By hypothesis

$$g(\tilde{s}_{r+n}) + B_l(\tilde{s}_{r+n}) \geq B_l(s^*_{r+n-1}) \tag{3.16}$$

Therefore (3.15) may be rewritten as

$$\mathcal{V}(\tilde{\gamma}_n(\tilde{s}_r)) \geq \mathcal{V}^*_{n-1}(\tilde{s}_r) \tag{3.17}$$

and because $\tilde{\gamma}_n(\tilde{s}_r)$ is not optimal

$$\mathcal{V}^*_n(\tilde{s}_r) \geq \mathcal{V}(\tilde{\gamma}_n(\tilde{s}_r)) \tag{3.18}$$

such that

$$\mathcal{V}^*_n(\tilde{s}_r) \geq \mathcal{V}^*_{n-1}(\tilde{s}_r) \tag{3.19}$$

From the principle of optimality

$$\mathcal{V}^*_n(\tilde{s}_r) = g(\tilde{s}_{r+1}) + \mathcal{V}^*_{n-1}(\tilde{s}_{r+1}) \tag{3.20}$$

Substituting the right-hand side of (3.20) into the left-hand side of (3.19) yields

$$g(\tilde{s}_{r+1}) + \mathcal{V}^*_{n-1}(\tilde{s}_{r+1}) \geq \mathcal{V}^*_{n-1}(\tilde{s}_r) \tag{3.21}$$

Subtracting $\mathcal{V}^*_{n-1}(\tilde{s}_{r+1})$ from both sides of (3.21) gives

$$g(\tilde{s}_{r+1}) \geq \mathcal{V}^*_{n-1}(\tilde{s}_r) - \mathcal{V}^*_{n-1}(\tilde{s}_{r+1}) \tag{3.22}$$

Evaluating (3.22) at different values of $r$ from $r = 0$ to $r = l - 1$ yields

$$\begin{aligned} g(\tilde{s}_1) \geq & \mathcal{V}^*_{n-1}(\tilde{s}_0) - \mathcal{V}^*_{n-1}(\tilde{s}_1) \\ g(\tilde{s}_2) \geq & \mathcal{V}^*_{n-1}(\tilde{s}_1) - \mathcal{V}^*_{n-1}(\tilde{s}_2) \\ & \vdots \\ g(\tilde{s}_l) \geq & \mathcal{V}^*_{n-1}(\tilde{s}_{l-1}) - \mathcal{V}^*_{n-1}(\tilde{s}_l) \end{aligned} \tag{3.23}$$

such that summing the inequalities in (3.23) and simplifying yields

$$\sum_{j=1}^{l} g(\tilde{s}_j) \geq \mathcal{V}^*_{n-1}(\tilde{s}_0) - \mathcal{V}^*_{n-1}(\tilde{s}_l) \tag{3.24}$$

Because we consider a path of length $l$, $\mathcal{V}^*_{n-1}(\tilde{s}_l)$ evaluates to

$$\mathcal{V}^*_{n-1}(\tilde{s}_l) = 0 \tag{3.25}$$

We also recognize that $B_l(s_0)$ represents a lower bound on the value-to-go (see (3.5)) such that

$$\mathcal{V}^*_{n-1}(\tilde{s}_0) \geq B_l(\tilde{s}_0) \tag{3.26}$$

and (3.24) becomes

$$\sum_{j=1}^{l} g(\tilde{s}_j) \geq B_l(\tilde{s}_0) \tag{3.27}$$

where $\sum_{j=1}^{l} g(\tilde{s}_j)$ is the value of the $l$-length receding horizon path $\boldsymbol{P}_l$. Therefore we conclude that

$$J(\boldsymbol{P}_l) \geq B_l(s_0) \tag{3.28}$$

□

The lower bound $B_l(s_r)$ on the value-to-go could be defined as the value of traversing a naive path through the search space beginning from the position $s_r$. In the case of subsea search, lawn mower paths perform well [26] and can be computed efficiently. This makes them well suited to determine a lower bound on the value-to-go.

**Corollary 3.3.** *Suppose that*

$$\gamma_{l-r-n}(\tilde{s}_{r+n}) = \{\tilde{s}_{r+n}, \dots \tilde{s}_l\} \tag{3.29}$$

*is a given naive, suboptimal path, and let the value-to-go be defined*

$$B_l(\tilde{s}_{r+n}) \triangleq \sum_{j=r+n+1}^{l} g(\tilde{s}_j) \tag{3.30}$$

*Then there exists $\tilde{s}_{r+n+1}$ feasible from $\tilde{s}_{r+n}$ which satisfies*

$$B_l(\tilde{s}_{r+n}) \leq g(\tilde{s}_{r+n+1}) + B_l(\tilde{s}_{r+n+1}) \tag{3.31}$$

*Proof.* From (3.30), we note that

$$\sum_{j=r+n+1}^{l} g(\tilde{s}_j) = g(\tilde{s}_{r+n+1}) + \sum_{j=r+n+2}^{l} g(\tilde{s}_j) \tag{3.32}$$

Given that $B_l(\tilde{s}_{r+n+1})$ is defined by the right-most summation in (3.32), we write

$$\sum_{j=r+n+1}^{l} g(\tilde{s}_j) = g(\tilde{s}_{r+n+1}) + B_l(\tilde{s}_{r+n+1}) \tag{3.33}$$

Substituting the left-hand side of (3.30) into the left-hand side of (3.33) yields

$$B_l(\tilde{s}_{r+n}) = g(\tilde{s}_{r+n+1}) + B_l(\tilde{s}_{r+n+1}) \tag{3.34}$$

which satisfies

$$B_l(\tilde{s}_{r+n}) \leq g(\tilde{s}_{r+n+1}) + B_l(\tilde{s}_{r+n+1}) \tag{3.35}$$

□

## 3.3 Numerical Experiments

### 3.3.1 Problem Formulation

The numerical experiments presented in this section address the problem of planning search paths in an uncertain subsea environment as presented in [53]. We consider the $61 \times 61$ gridded search space shown in Figure 3.1. We note that length scales have been normalized so that each grid element has unit length and width. Each cell $h_i$ in the search space contains an unknown number of targets $x \in X$. Each cell also has associated environment type $e \in E$. We assume that the environment within the cell is from a finite set of possible environment types $\{e_1, e_2, \ldots, e_m\}$. A probability distribution on the environment type in cell $h_i$ is known *a priori*. The number of targets contained within a cell and the environment type within the cell are assumed to be independent of all other cells. We assume the search sensor performance is dependent on the environment $e$ within the cell. That is, the environment type $e$ of the $i^{\text{th}}$ cell $h_i$ directly influences the uncertainty of the sensor measurements acquired from $h_i$.

Upon visiting a cell $h_i$, the search agent acquires a noisy measurement $y \in Y$ of the environment type $e$ and $z \in Z$ of the number of targets contained within the cell. The anticipated value of searching a cell is determined through a decision-theoretic reward function. Specifically, the anticipated value of searching a cell is the expected reduction in Bayes risk due to the measurements $z_i$ and $y_i$ acquired upon visiting a cell. That is, given the estimator

$d^\star(y_i)$ which minimizes the posterior expected loss of computing the environment type estimate and the estimator $\delta^\star(x_i)$ which minimizes the posterior expected loss of computing the estimate of the number of targets in cell $h_i$ given $d^\star(y)$, Bayes risk is the posterior expected loss of computing the estimate on the number of targets in cell $h_i$. The anticipated value of searching cell $h_i$ is

$$B(i) = \rho_i - r(i,k) \tag{3.36}$$

where $\rho_i$ is the current risk before acquiring the measurements $z$ and $y$, and is defined as

$$\rho_i = \sum_{j=1}^{m} P(e_j)\mathbb{E}[L(x,\delta^\star)|e_j] \tag{3.37}$$

$r(i,k)$ is the anticipated risk given $k$ visits to the $i^{\text{th}}$ cell $h_i$ and is defined as

$$r(i,k) = \sum_{z_{i,1}}\sum_{y_{i,1}}\cdots\sum_{z_{i,k}}\sum_{y_{i,k}} P(z_i,y_i)\mathbb{E}\big[L(x,\delta^\star(z_i))|z_i,d^\star(y_i)\big] \tag{3.38}$$

where $z_i = [z_{i,1},\ldots,z_{i,k}]$ is the set of independent measurements of the number of targets in cell $h_i$, $y_i = [y_{i,1},\ldots,y_{i,k}]$ is the set of independent measurements of the environmental type in cell $h_i$, and

$$P(z_i,y_i) = \sum_{x}\sum_{e_j} P(x|z_1,e_j)\cdots P(x|z_k,e_j)P(y_1|e_j)\cdots P(y_k|e_j)P(x)P(e_j) \tag{3.39}$$

is the joint probability of sensing $z_i, y_i$ given the prior environmental distribution $P(e_j)$ for cell $h_i$ and the prior distribution on the number of targets $P(x)$ in cell $h_i$.

The actual value of visiting cell $h_i$ is the actual risk reduction given measurements $y$ and $z$. That is, given $\rho_i$ is the current risk before acquiring the measurements $z$ and $y$ and $\bar{\rho}_i$ is the current risk after acquiring measurements $z$ and $y$, then the actual risk reduction for visiting

cell $h_i$ is

$$\bar{B}(i) = \rho_i - \bar{\rho}_i \tag{3.40}$$

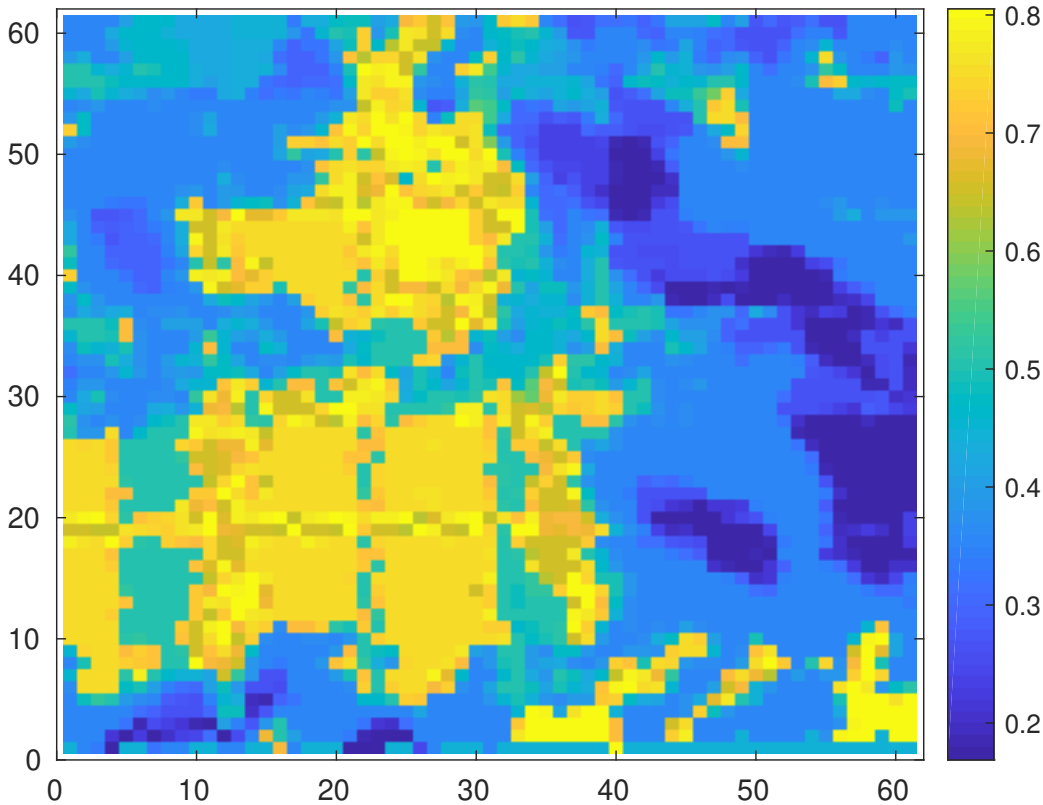



Figure 3.1: The $61 \times 61$ gridded subsea search environment with cell color corresponding to the anticipated risk reduction for a single visit to each cell.

These numerical experiments consider that the search agent is an autonomous underwater vehicle (AUV) carrying side-scan sonar. Side-scan sonar operations require the vehicle to traverse straight line paths in order to acquire useful data when performing synthetic aperture sonar processing [32]. At each location, the AUV may *move forward*, *turn left*, or *turn right*. We consider that each action contributes to the total length of the search path. However, we presume that sensor data acquired during a turning maneuver does not contain useful

information and thus the function $g$ used to determine the anticipated value of traversing a path is

$$g(s_r) = \begin{cases} B(i) & \text{for } s_r \in h_i \text{ with no turning maneuver} \\ 0 & \text{else} \end{cases} \tag{3.41}$$

Similarly, after obtaining measurements $y$ and $z$ of the environment type within cell $h_i$ and the number of targets contained within $h_i$ when traversing cell $h_i$ without executing a turning maneuver, the non-negative function $g$ used to determine the actual value of traversing cell $h_i$ is

$$\bar{g}(s_r) = \begin{cases} \bar{B}(i) & \text{for } s_r \in h_i \text{ with no turning maneuver} \\ 0 & \text{else} \end{cases} \tag{3.42}$$

An example of a feasible path for the AUV in a $15 \times 15$ uniform search grid is shown in Figure 3.2. The solid line represents the receding horizon path composed of 1-step segments along $n$-length optimal paths. A feasible $n$-length path is represented by the asterisked solid line. The lawn-mower "tail" corresponding to the $n$-length path is shown as the dashed line.

### 3.3.2 Results

Throughout these numerical experiments, we determine optimal paths within the planning horizon $n$ using exhaustive search. These optimal paths are used to construct RH paths as discussed in Chapter 2. We note that the value of visiting a cell tends to decrease with consecutive visits. As such, we do not consider paths that traverse the same cell more than 2 times when computing optimal paths.

Figure 3.3 shows the average value (risk reduction) of RH paths constructed with terminal rewards within the environment represented in Figure 3.1 using total path length $l = 100$ to

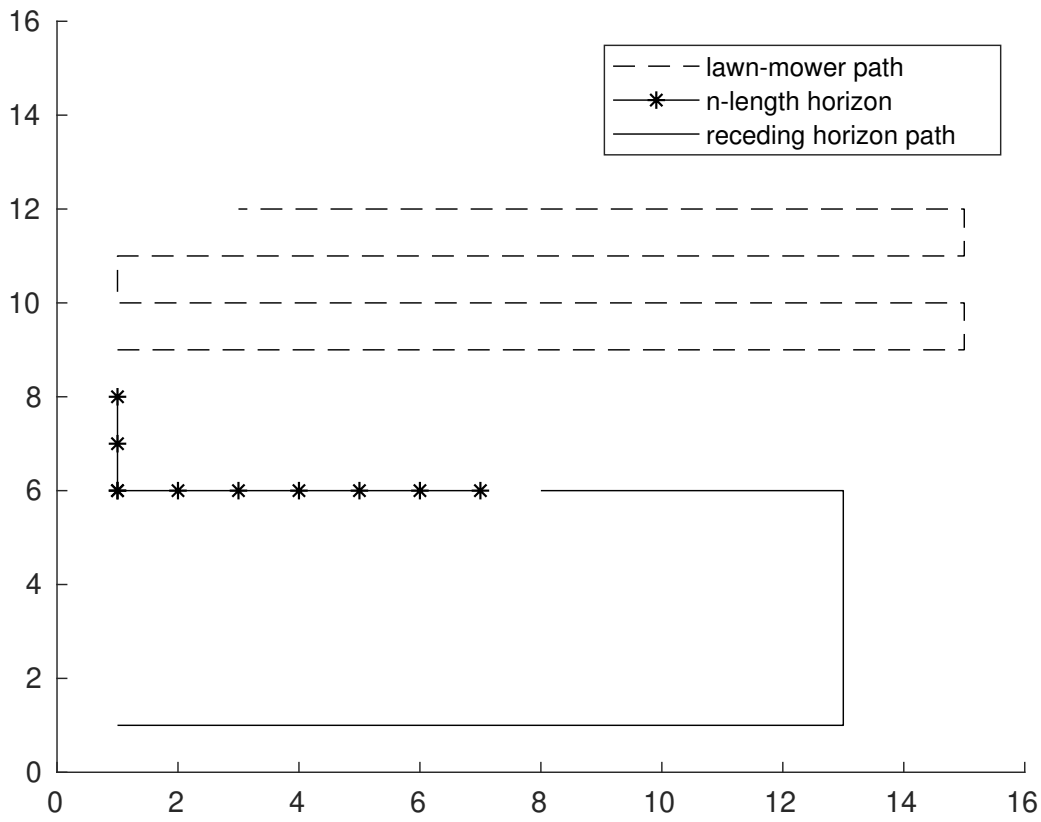



Figure 3.2: Example path

$l = 2000$ in increments of 100. Each red asterisk in Figure 3.3 (top) represents the average risk reduction of 31 RH paths planned using the planning horizon $n = 2$. Similarly, each blue asterisk (bottom) represents the average risk reduction of 31 RH paths planned using planning horizon $n = 5$. The dashed black line (top and bottom) represents the anticipated risk reduction of the lawn mower paths used to determine the lower bound on the value-to-go in (3.5). As can be seen in Figure 3.3, the value-to-go does indeed provide a lower bound on the performance of the receding horizon search paths and is outperformed by the receding horizon paths in most cases. The maximum standard deviation using planning horizon $n = 2$ is 12.6, while the maximum standard deviation using planning horizon $n = 5$ is 6.9.

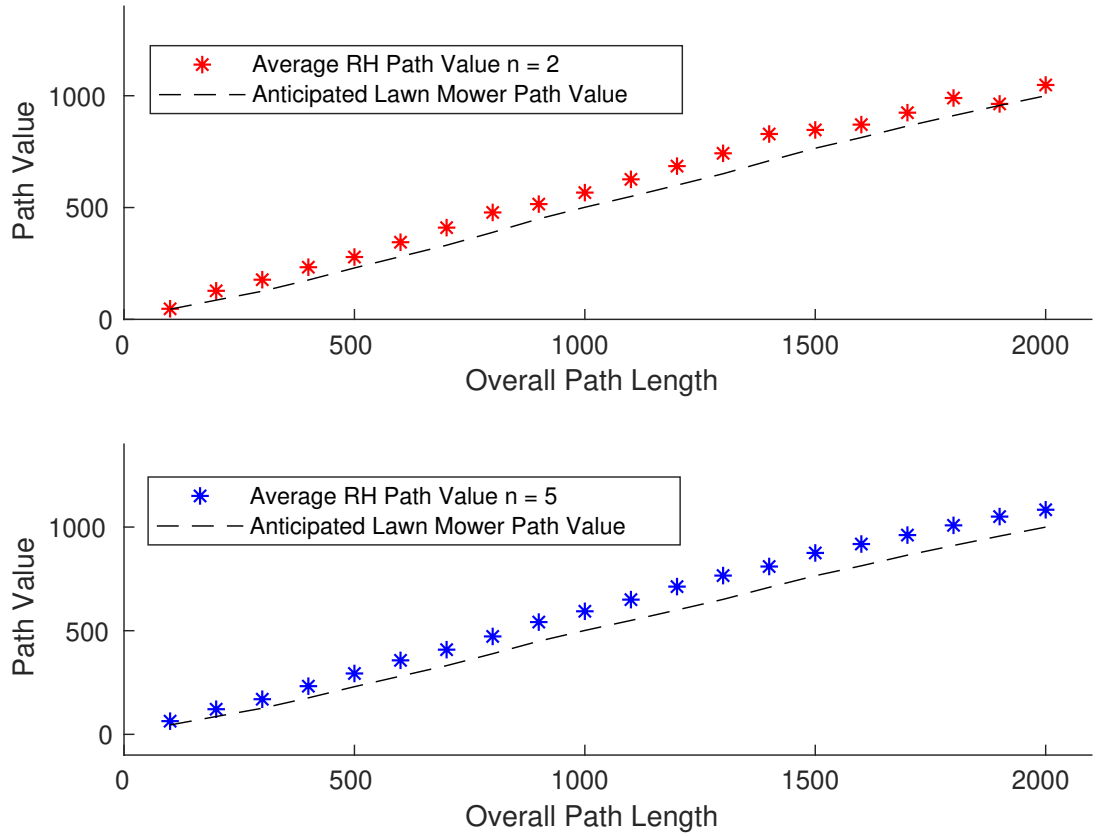



Figure 3.3: Average value of RH paths constructed with $n = 2$ (top) and $n = 5$ (bottom) vs anticipated value of lawn mower path.

One method of improving the performance of RH search paths without utilizing terminal rewards is extending the planning horizon $n$. This represents a serious computational challenge as the problem of finding the optimal path within only the planning horizon grows exponentially with the length of the planning horizon. Each red square in Figure 3.4 represents the actual value obtained by traversing a RH path constructed without terminal rewards using planning horizon $n = 10$ for total path length $l = 100$ to $l = 2000$. The anticipated values of lawn mower paths and the average values of RH paths that are constructed with terminal rewards using planning horizon $n = 5$ (shown in Figure 3.3) are also shown in Figure 3.4 for reference. Notably, extending the planning horizon to double the maximum planning

horizon considered in Figure 3.3 provides only comparable performance to RH paths with much shorter planning horizons which consider terminal rewards.

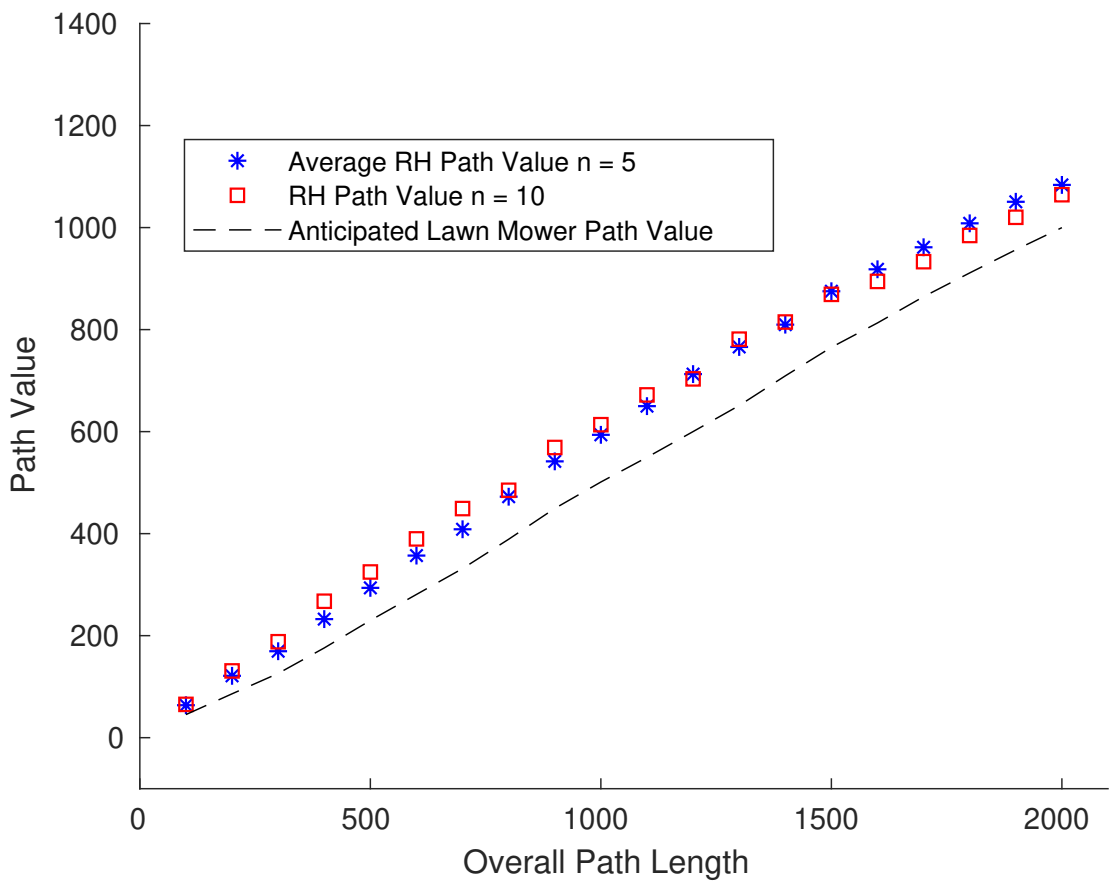



Figure 3.4: Standard RH path values with extended planning horizon vs anticipated lawn mower path values and average RH path values for RH paths constructed using terminal rewards.

Figure 3.5 shows an example of a RH path constructed without the use of terminal rewards using total path length $l = 1000$ and planning horizon $n = 4$. The path shown in Figure 3.5 resulted in a total risk reduction (path value) of approximately 580. Figure 3.6 shows an example of a RH path constructed with terminal rewards using total path length $l = 1000$ and planning horizon $n = 4$, which results in a total risk reduction of approximately 610. The path shown in Figure 3.5 displays greedy characteristics leading to unnecessary turning.

In comparison, the path shown in Figure 3.6 retains beneficial attributes of the naive lawn mower paths used to determine terminal rewards during path construction.

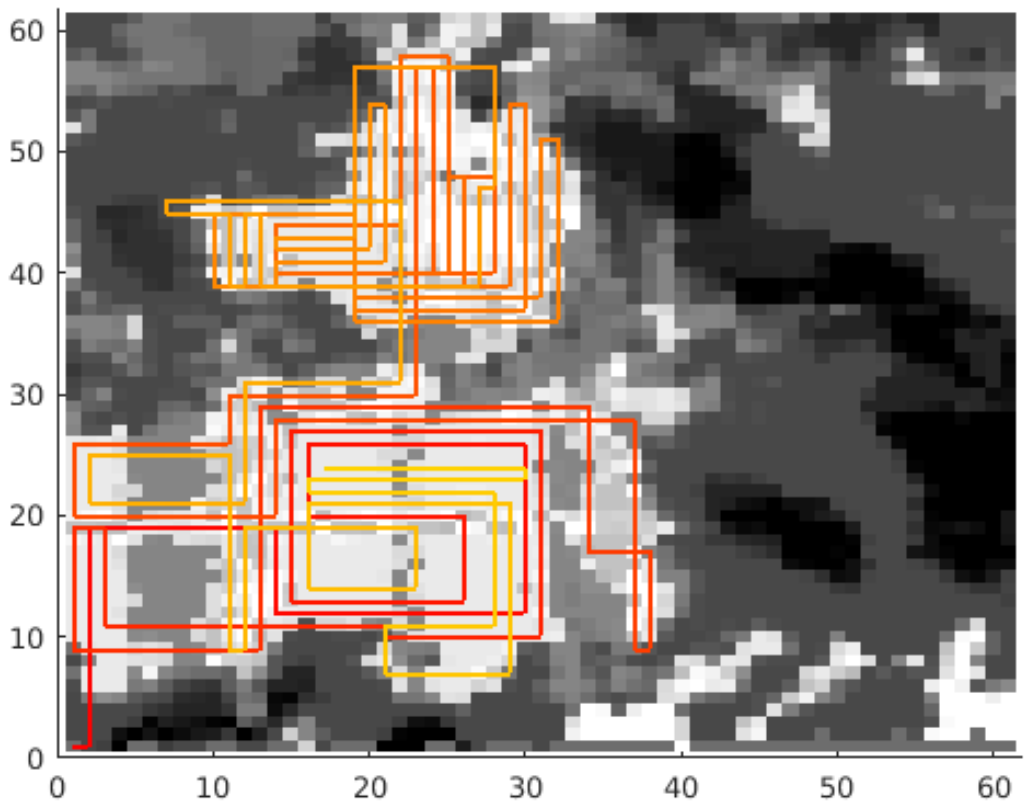



Figure 3.5: Example of an RH path constructed without terminal rewards using parameters $n = 4$ and $l = 1000$. The path begins at the red end of the line (bottom left) and ends at the yellow end of the line.

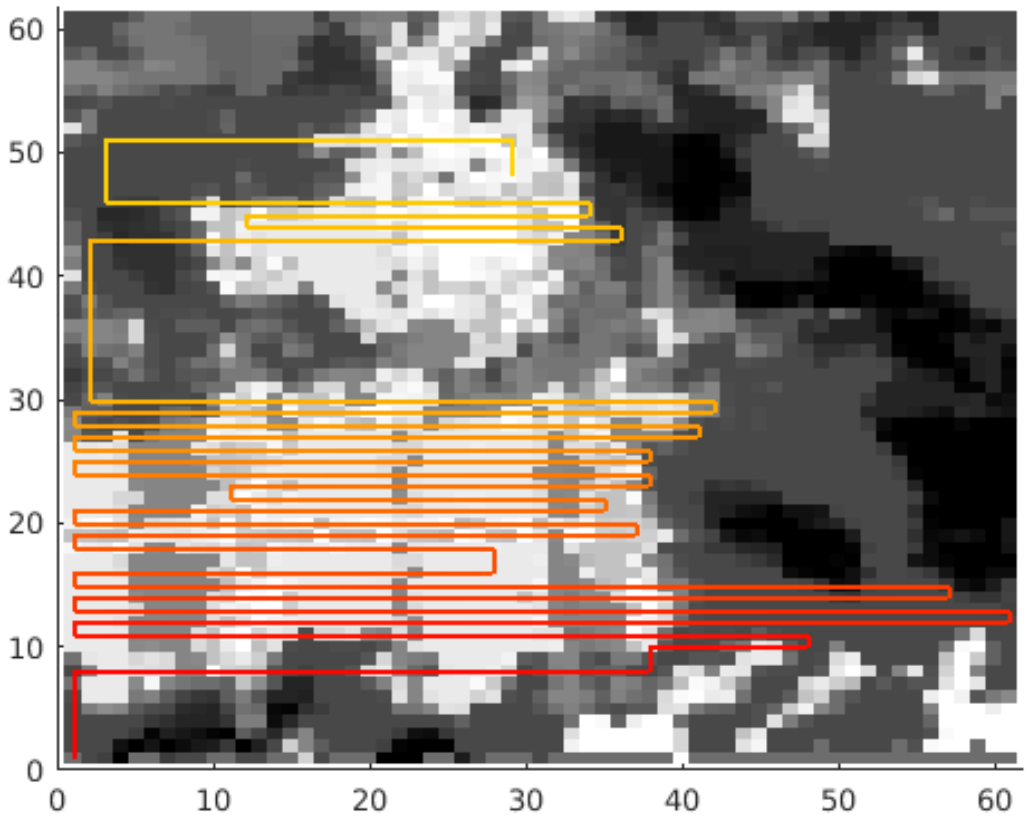



Figure 3.6: Example of an RH path constructed with terminal rewards using parameters $n = 4$ and $l = 1000$. The path begins at the red end of the line (bottom left) and ends at the yellow end of the line.

# Chapter 4

# Extended Performance Guarantees for Receding Horizon Search with Terminal Rewards

In Chapter 3, we showed that a specific class of terminal reward could be appended to the optimization problem that would lead to useful lower bounds for informative path planning, and that for applications such as subsea mine-hunting, which is the motivation for our work, a suitable terminal reward is always available. Our approach in Chapter 3 addresses the case that the short-horizon path is the optimal, but cannot be directly extended to the case that the short-horizon path is sub-optimal, as would happen using many real-time planning approaches, such as sample-based planning (e.g., Monte-Carlo Tree Search). In this chapter, we specifically address the case of sample-based planning that generates sub-optimal short-horizon paths. We propose a specific terminal reward and corresponding sufficient conditions for which the receding horizon path inherits a sub-optimal property from the sub-optimality of each short-horizon path.

Approaches to informative path planning related to this chapter include [1, 2, 62, 69], where the concept of algebraic redundancy is used to prune a search tree. These methods allow for longer planning horizons, but do not utilize terminal rewards. In [77], the planning horizon is extended until the reward of a path surpasses a threshold value, but no terminal reward

is appended. The methods in [34, 38] employ sampling based methods to plan non-myopic search paths with guaranteed convergence to the optimal search path in the case of [38]. In contrast we use sampling based methods (Monte Carlo Tree Search) within a short planning horizon and append terminal rewards in order to guarantee a lower bound on the performance of search paths. Novel roll-out policies for Monte Carlo Tree Search are presented in [76] in the context of subsea search path planning. We use a slightly modified version of Monte Carlo Tree Search within a short planning horizon and incorporate terminal rewards.

## 4.1 Sub-optimality Guarantees for Receding Horizon Path Planning with Terminal Rewards

In Chapter 3 it was shown that appropriate use of a terminal reward when planning receding horizon search paths can guarantee a lower bound on the value of the receding horizon search paths given that the path produced at each planning step is optimal. We extend the main result from Chapter 3 to the case that at each planning step the path produced is *nearly optimal*. We also consider a multi-step execution horizon $m$ instead of the single step execution horizon considered in Chapter 3. One motivation for using an $m$-step execution horizon is that some optimization methods such as Monte Carlo Tree Search can generate better paths when the planning time is longer. In other words, allowing the search agent to complete multiple state transitions while planning may allow for extended planning horizons that may in turn benefit search performance.

In Proposition 4.1, we state a condition under which the value of a receding horizon path constructed from near-optimal paths is always bounded below by the lower bound on the

value-to-go. We say that a path $\gamma'_n(s_r)$ is *nearly optimal* if the value of the path satisfies

$$\mathcal{V}'_n(s_r) \geq \mathcal{V}^*_n(s_r) - \epsilon \tag{4.1}$$

for some $\epsilon \geq 0$.

**Proposition 4.1.** *Suppose the $l$-length receding horizon path $\boldsymbol{P}_l$ is composed of $m$-step sequences along a sequence of $n$-length near-optimal paths that seek to maximize (3.5) satisfying (4.1). For every $n$-length near-optimal path*

$$\gamma'_n(s_r) = \{s_r, s'_{r+1}, \ldots, s'_{r+n}\} \tag{4.2}$$

*suppose that there exists a set of states $\{\tilde{s}_{r+n+1}, \ldots, \tilde{s}_{r+n+m}\}$ that are feasible from $s'_{r+n}$ such that*

$$B_l(s'_{r+n}) + \epsilon \leq \sum_{i=r+n+1}^{r+m} g(\tilde{s}_i) + B_l(\tilde{s}_{r+n+m}) \tag{4.3}$$

*then the RH path satisfies*

$$J(\boldsymbol{P}_l) \geq B_l(s_0) \tag{4.4}$$

*Proof.* Consider the $(n+m)$-length path $\tilde{\gamma}_n(s_r) = \{\gamma'_n(s_r), \tilde{s}_{r+n+1}, \ldots, \tilde{s}_{r+n+m}\}$ composed of an $n$-length near-optimal path $\gamma'(s_r) = \{s_r, s'_{r+1}, \ldots, s'_{r+n}\}$ and a feasible set of states $\{\tilde{s}_{r+n+1}, \ldots, \tilde{s}_{r+n+m}\}$ satisfying (4.3). Then from equation (3.5) the value of $\tilde{\gamma}_n(s_r)$ is

$$\tilde{\mathcal{V}}_{n+m}(s_r) = \sum_{i=r+1}^{r+n} g(s'_i) + \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + B_l(\tilde{s}_{r+n+m}) \tag{4.5}$$

or equivalently

$$\tilde{\mathcal{V}}_{n+m}(s_r) = \mathcal{V}'_n(s_r) - B_l(s'_{r+n}) + \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + B_l(\tilde{s}_{r+n+m}). \tag{4.6}$$

By hypothesis

$$B_l(s'_{r+n}) + \epsilon \leq \sum_{i=r+n+1}^{r+m} g(\tilde{s}_i) + B_l(\tilde{s}_{r+n+m}) \tag{4.7}$$

such that

$$\tilde{\mathcal{V}}_{n+m}(s_r) \geq \mathcal{V}'_n(s_r) + \epsilon, \tag{4.8}$$

and $\mathcal{V}'_n(s_r)$ is bounded below by $\mathcal{V}^*_n(s_r) - \epsilon$ yielding

$$\tilde{\mathcal{V}}_{n+m}(s_r) \geq \mathcal{V}^*_n(s_r). \tag{4.9}$$

We note that (4.5) may be equivalently written as

$$\tilde{\mathcal{V}}_{n+m}(s_r) = \sum_{i=r+1}^{r+m} g(s'_i) + \sum_{i=r+m+1}^{r+n} g(s'_i) + \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + B_l(\tilde{s}_{r+n+m}). \tag{4.10}$$

That is, there exists some feasible $n$-length path $\hat{\gamma}_n(s'_{r+m})$ such that

$$\hat{\mathcal{V}}_n(s'_{r+m}) = \sum_{i=r+m+1}^{r+n} g(s'_i) + \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + B_l(\tilde{s}_{r+n+m}) \tag{4.11}$$

and

$$\tilde{\mathcal{V}}_{n+m}(s_r) = \sum_{i=1}^{m} g(s'_i) + \hat{\mathcal{V}}_n(s'_{r+m}). \tag{4.12}$$

Replacing the left-hand side of (4.9) with the right-hand side of (4.12) gives

$$\sum_{i=r+1}^{r+m} g(s'_i) + \hat{\mathcal{V}}_n(s'_{r+m}) \geq \mathcal{V}^*_n(s_r) \tag{4.13}$$

where $\hat{\gamma}_n(s'_{r+m})$ is not optimal and therefore

$$\sum_{i=r+1}^{r+m} g(s'_i) + \mathcal{V}_n^*(s'_{r+m}) \geq \mathcal{V}_n^*(s_r) \tag{4.14}$$

where subtracting $\mathcal{V}_n^*(s'_{r+m})$ from both sides gives

$$\sum_{i=r+1}^{r+m} g(s'_i) \geq \mathcal{V}_n^*(s_r) - \mathcal{V}_n^*(s'_{r+m}). \tag{4.15}$$

Evaluating (4.15) at different values of $r$ from $r = 1$ to $r = l - m$ in increments of $m$ gives

$$\sum_{i=1}^{m} g(s'_i) \geq \mathcal{V}_n^*(s_0) - \mathcal{V}_n^*(s'_m) \tag{4.16}$$

$$\sum_{i=m+1}^{2m} g(s'_i) \geq \mathcal{V}_n^*(s_m) - \mathcal{V}_n^*(s'_{2m}) \tag{4.17}$$

$$\vdots$$

$$\sum_{i=l-m+1}^{l} g(s'_i) \geq \mathcal{V}_n^*(s_{l-m}) - \mathcal{V}_n^*(s'_l). \tag{4.18}$$

And summing the inequalities in (4.16) gives

$$\sum_{i=1}^{l} g(s'_i) \geq \mathcal{V}_n^*(s_0) - \mathcal{V}_n^*(s'_m) + \mathcal{V}_n^*(s_m) - \mathcal{V}_n^*(s'_{2m}) + \ldots + \mathcal{V}_n^*(s_{l-m}) - \mathcal{V}_n^*(s'_l). \tag{4.19}$$

Which, because in receding horizon path planning $s_m = s'_m$,$s_{2m} = s'_{2m}$ and so on, reduces to

$$\sum_{i=1}^{l} g(s'_i) \geq \mathcal{V}_n^*(s_0) - \mathcal{V}_n^*(s'_l). \tag{4.20}$$

We note that the summation on the left-hand side of (4.20) represents the value of the $l$-length receding horizon path $\boldsymbol{P_l}$ and that $\mathcal{V}_n^*(s'_l) = 0$. Using these observations, we rewrite

(4.20) as

$$J(\boldsymbol{P_l}) \geq \mathcal{V}_n^*(s_0). \tag{4.21}$$

Finally, because $\mathcal{V}_n^*(s_0) \geq B_l(s_0)$, we conclude that

$$J(\boldsymbol{P_l}) \geq B_l(s_0). \tag{4.22}$$

□

As $\epsilon$ increases, the existence of a set of feasible locations satisfying (4.3) may be a difficult requirement to satisfy. Therefore, in Proposition 4.2 we state a condition under which the value of a receding horizon path is always no less than the lower bound on the value-to-go from the initial state along the $l$-length receding horizon path. In Corollary 4.3 we show that the hypothesis of Proposition 4.2 can always be satisfied given a terminal reward defined by the best path in a set of readily available naive paths, such as mowing-the-lawn paths that are used often for subsea applications. Corollary 4.3 also satisfies (4.3) given $\epsilon = 0$.

**Proposition 4.2.** *Suppose that at each planning step the path $\gamma_n'(s_r)$ satisfies*

$$\mathcal{V}_n'(s_r) \geq \mathcal{V}_{n-m}'(s_r) \tag{4.23}$$

*where $\mathcal{V}_{n-m}'(s_r)$ is the value of the $n-m$ steps not included in the execution horizon of $\gamma_n'(s_{r-1})$ and that*

$$\mathcal{V}_n'(s_0) \geq B_l(s_0) \tag{4.24}$$

*then the RH path satisfies*

$$J(\boldsymbol{P}_l) \geq B_l(s_0) \tag{4.25}$$

*Proof.* From (3.5), the reward attained while traversing the first $m$-steps along a path may

be represented by

$$\sum_{i=r+1}^{r+m} g(s'_i) = \mathcal{V}'_n(s_r) - \sum_{i=r+m+1}^{r+n} g(s'_i) - B_l(s'_{r+n}). \tag{4.26}$$

Where $\sum_{i=r+m+1}^{r+n} g(s'_i) + B_l(s'_{r+n})$ is the value of an $(n-m)$-length path remaining of $\gamma'_n(s_r)$ after traversing the first $m$ steps. We use the notation

$$\mathcal{V}'_{n-m}(s'_{r+m}) = \sum_{i=r+m+1}^{r+n} g(s'_i) + B_l(s'_{r+n}) \tag{4.27}$$

and write

$$\sum_{i=r+1}^{r+m} g(s'_i) = \mathcal{V}'_n(s_r) - \mathcal{V}'_{n-m}(s'_{r+m}). \tag{4.28}$$

As in the proof of Proposition 4.1, we evaluate (4.28) at different values of $r$ from $r = 1$ to $r = l - m$ in increments of $m$ to get

$$\begin{aligned} \sum_{i=1}^{m} g(s'_i) &= \mathcal{V}'_n(s_0) - \mathcal{V}'_{n-m}(s'_m) \\ \sum_{i=m+1}^{2m} g(s'_i) &= \mathcal{V}'_n(s_m) - \mathcal{V}'_{n-m}(s'_{2m}) \\ &\vdots \\ \sum_{i=l-m+1}^{l} g(s'_i) &= \mathcal{V}'_n(s_{l-m}) - \mathcal{V}'_{n-m}(s'_l) \end{aligned} \tag{4.29}$$

where summing the inequalities in (4.29) gives

$$\sum_{i=1}^{l} g(s'_i) = \mathcal{V}'_n(s_0) - \mathcal{V}'_{n-m}(s'_m) + \mathcal{V}'_n(s_m) - \mathcal{V}'_{n-m}(s'_{2m}) + \ldots + \mathcal{V}'_n(s_{l-m}) - \mathcal{V}'_{n-m}(s'_l). \tag{4.30}$$

By hypothesis $\mathcal{V}'_n(s_r) \geq \mathcal{V}'_{n-m}(s_r)$. Therefore, because in receding horizon path planning

$s_m = s'_m$,$s_{2m} = s'_{2m}$ and so on, (4.30) reduces to

$$\sum_{i=1}^{l} g(s'_i) \geq \mathcal{V}'_n(s_0) - \mathcal{V}'_{n-m}(s'_l). \tag{4.31}$$

We note that the summation on the left-hand side of (4.31) represents the value of the $l$-length receding horizon path $\boldsymbol{P_l}$ and that $\mathcal{V}'_{n-m}(s'_l) = 0$. Using these observations, we rewrite (4.31) as

$$J(\boldsymbol{P_l}) \geq \mathcal{V}'_n(s_0), \tag{4.32}$$

and from (4.24) we conclude that

$$J(\boldsymbol{P_l}) \geq B_l(s_0). \tag{4.33}$$

□

If the best path in a set of readily available, naive paths is used to determine the lower bound on the value-to-go then there always exists an $n$-length path at the next planning step satisfying (4.23). Specifically (4.23) is satisfied because the first $m$ steps along the best naive path in the set of naive paths used to determine the lower bound on the value-to-go may be appended to the remainder of the optimal $n$-length path from the previous planning step guaranteeing the existence of an $n$-length path satisfying (4.23).

Let $\Gamma^K_{l-r-n}(s_{r+n})$ be a set of $K$ feasible naive paths such as a set of mowing-the-lawn paths that are commonly used in subsea applications.

**Corollary 4.3.** *Given $\Gamma^K_{l-r-n}(s_{r+n})$, let the lower bound on the value-to-go be defined*

$$B_l(s_{r+n}) = \max_{\tilde{\gamma}_{l-r-n}(s_{r+n}) \in \Gamma^r_{l-r-n}(s_{r+n})} J(\tilde{\gamma}_{l-r-n}(s_{r+n})) \tag{4.34}$$

*Then the inequality*

$$B_l(\tilde{s}_{r+n}) \leq \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + B_l(s_{r+n+m}) \tag{4.35}$$

*is always satisfied.*

*Proof.* Let $\tilde{\gamma}^*_{l-r-n}(s_{r+n})$ be a path in $\Gamma^K_{l-r-n}(s_{r+n})$ satisfying

$$\tilde{\gamma}^*_{l-r-n}(s_{r+n}) = \underset{\tilde{\gamma}_{l-r-n}(s_{r+n}) \in \Gamma^K_{l-r-n}(s_{r+n})}{\arg\max} J(\tilde{\gamma}_{l-r-n}(s_{r+n}) \tag{4.36}$$

such that

$$B_l(s_{r+n}) = J(\tilde{\gamma}^*_{l-r-n}(s_{r+n})). \tag{4.37}$$

We recall that

$$J(\tilde{\gamma}^*_{l-r-n}(s_{r+n})) = \sum_{j=r+n+1}^{l} g(\tilde{s}_j) \tag{4.38}$$

and note that

$$\sum_{j=r+n+1}^{l} g(\tilde{s}_j) = \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + \sum_{j=r+n+m+1}^{l} g(\tilde{s}_j). \tag{4.39}$$

Given that $B_l(\tilde{s}_{r+n})$ is defined by the right-most summation in (4.38), we write

$$B_l(\tilde{s}_{r+n}) = \sum_{i=r+n+1}^{r+n+m} g(\tilde{s}_i) + \sum_{j=r+n+m+1}^{l} g(\tilde{s}_j). \tag{4.40}$$

We note that $\sum_{j=r+n+m+1}^{l} g(\tilde{s}_j)$ represents the value of a single path in the set of available paths used to define the lower bound on the value-to-go and as such may not represent the highest valued naive path such that

$$\sum_{j=r+n+2}^{l} g(\tilde{s}_j) \leq B_l(s_{r+n+m}). \tag{4.41}$$

Using (4.41) and (4.40) we write

$$B_l(\tilde{s}_{r+n}) \leq g(\tilde{s}_{r+n+1}) + \ldots + g(\tilde{s}_{r+n+m}) + B_l(s_{r+n+m}) \tag{4.42}$$

which concludes the proof. □

The main benefit of Corollary 4.3 is the ability to leverage strengths of distinct naive paths. For example, a greedy receding horizon planner is computationally efficient. Therefore, an unbiased path such as a lawnmower path may be used alongside a greedy planner that incorporates a terminal reward to determine a lower bound on the value-to-go from a given state. Leveraging the strengths of several naive paths then allows the receding horizon planner to more closely approximate the value of (3.3) than would normally be possible with a single naive path. This does not imply optimality of the receding horizon path. Instead an improved lower bound on the value-to-go guarantees an improved lower bound on the value of the receding horizon path produced.

## 4.2 Numerical Experiments

In order to verify the sub-optimality guarantees presented in propositions 4.1 and 4.2, we employ a probabilistic implementation of receding horizon control using Monte Carlo Tree Search (MCTS). MCTS has been used to address a wide variety of problems [10]. Theoretical analysis of the classical upper confidence bound applied to trees (UCT) algorithm is provided in [39, 40]. The UCT algorithm is designed to balance exploitation of best actions found so far and exploration of other actions in an effort to ensure that the algorithm has a small probability of choosing a sub-optimal action if stopped prematurely and, given enough time, will determine the best action. To estimate the value of a state associated with a new leaf

node the MCTS algorithm utilizes a finite set of Monte Carlo simulations to provide an expected value.

In this chapter, we utilize a basic implementation of MCTS with minor modifications to highlight the effect of appending a terminal reward to a receding horizon path. First, the UCT is computed by normalizing the estimated reward to a factor of a naively computed upper bound,

$$\frac{\mathbb{E}}{B_u(s_o)} + C\sqrt{\frac{\ln(N)}{n}} \tag{4.43}$$

where in this case we compute $B_u(s_o)$ by summing the $l$ highest value cells at the start of the mission. Second, instead of back-propagating the average reward of a Monte Carlo simulation, we back-propagate the maximum value. Lastly, in the experiments the search agent has three available actions: $\text{move forward}, \text{turn left}, \text{turn right}$. While there is no incurred penalty while performing a turn, the search agent does not sample the cell when turning. This is to more closely mirror the real-world application of subsea search where a search sensor is less effective when not traveling in straight line paths. To account for this in the MCTS planner, the roll-out policy performs a biased sampling of random actions where forward is sampled at a rate of 50% and the other two actions are sampled both at 25%. We then compare the results of a receding horizon MCTS planner with terminal rewards to a brute-force implementation which finds the optimal solution to (3.5) using the same mission length, horizon, and execution values: $l$, $n$, and $m$, respectively. Furthermore, we compare the results to both MCTS and brute force implementations where a terminal reward is not appended.

Throughout these numerical experiments, we leverage strengths of simple lawnmower paths in the context of subsea search [26] to compute a lower bound on the value-to-go from a given

state. In contrast to Chapter 3 where a horizontal lawnmower path was used to determine a lower bound on the value-to-go, we use the higher reward between a horizontal lawnmower path and a vertical lawnmower path starting from a given state.

Figure 4.1 is the environment the search agent is tasked with exploring. The map shows the reward for sampling a cell given a characterization metric for a subsea search sensor, the data is real data collected from a 21" bluefin UUV in the approaches to Boston Harbor (see [53] for a detailed explanation on the performance estimation metric). Using this environment we compare the planning results from receeding horizon planners that utilize a n-step receeding horizon with $n = 6$, and an m-step execution horizon of $m = 3$. The MCTS performed 20 simulations per leaf node evaluation and was given 4 seconds per step to plan (we planned for a total of 12 seconds per cycle since 3 steps were performed each cycle). For each experiment the starting position of the search agent was randomly selected from one of the four corners of the map. The mission length was set to values ranging from 500 steps to 5000 steps. MCTS (**MCTS-RH-TC**) experiments were conducted 40 times while the brute-force (**BF-RH-TC**) method was conducted 4 (one for each corner, the solution is deterministic so no additional trials are required). We compare the results of a receeding horizon planner with the terminal reward appended to the n-step path to a planner not utilizing a terminal reward (**BF-RH** and **MCTS-RH**, brute force and MCTS respectively). We also compare the results to the initial lower-bound computation (**LB**, the better of two naive lawnmower patterns starting from the initial position) as well as a **MCTS** planner that searches for the full-length path (otherwise using the same parameters as the **MCTS-RH** and **MTCS-RH-TC** planners). Figure 4.2 shows the results of the experiments, figure 4.3 show the results for $l$ at 5000, namely to highlight how close the **MCTS** results are able to mirror the optimal **BF** results. Figure 4.4a and 4.4b show two representative trials of the MCTS planners with and without the appended terminal reward respectively.

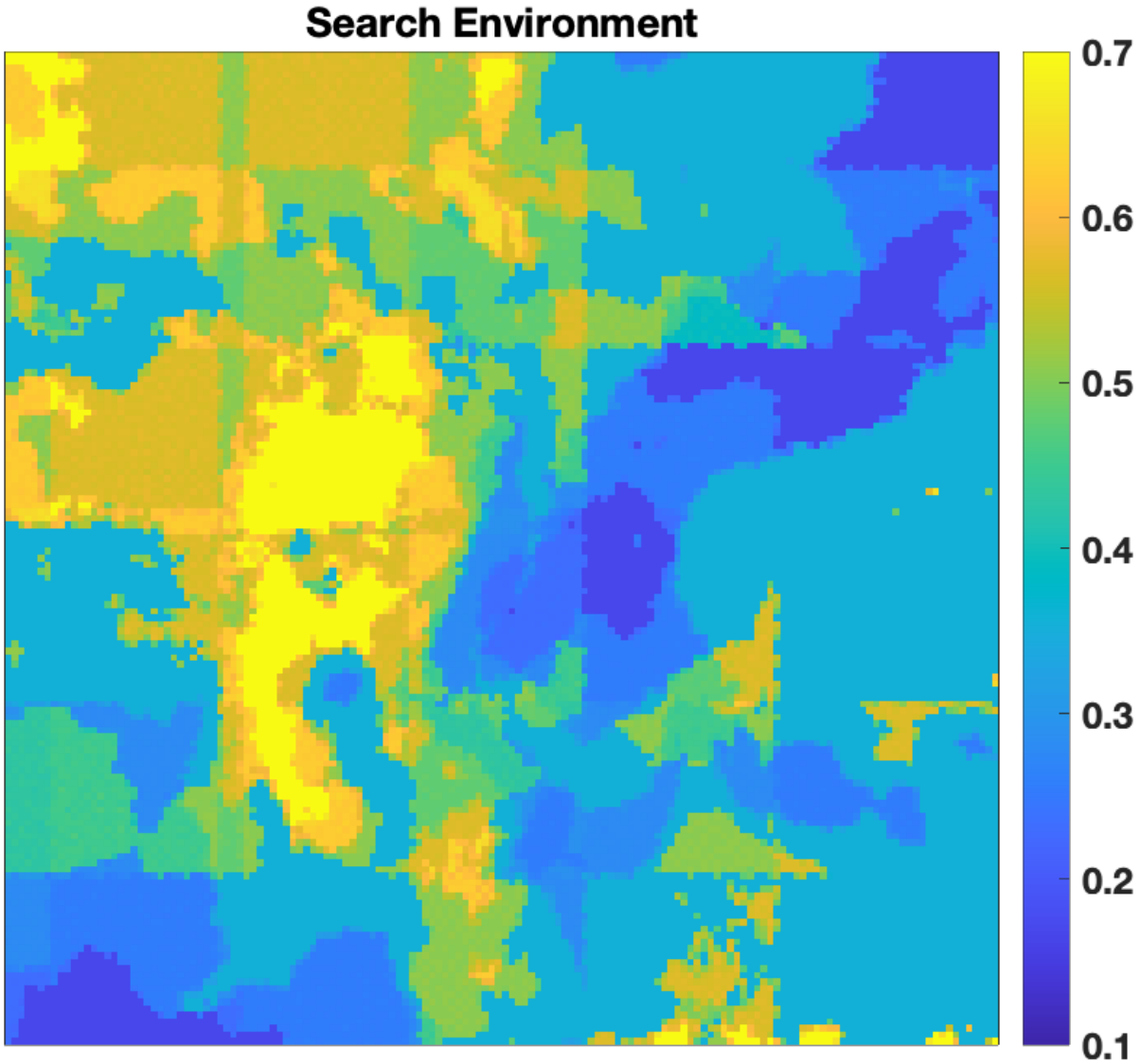



Figure 4.1: 150 x 150 cell sensor performance map of an environment measured in the approaches to Boston Harbor using side scan sonar on-board an unmanned underwater vehicle. The data corresponds to an environmental characterization metric which helps define how well the sonar will perform during search and characterization missions [53]

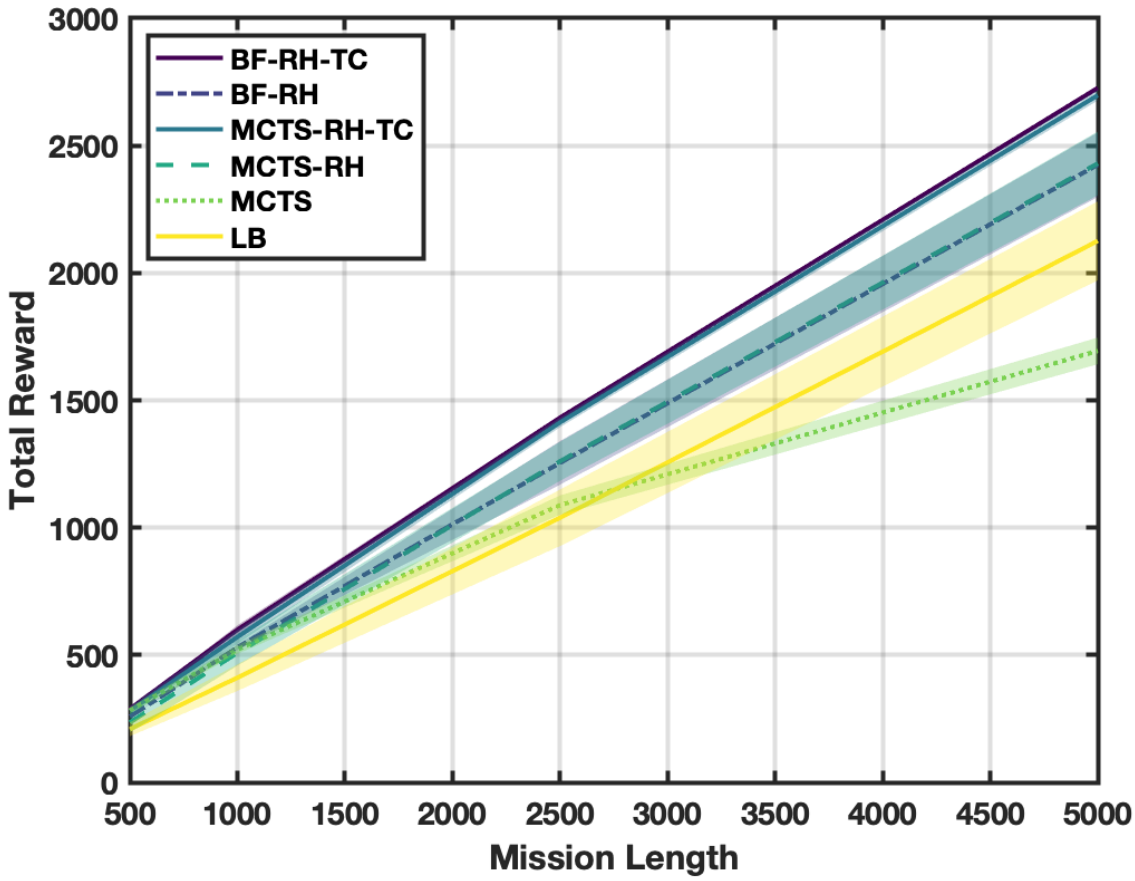
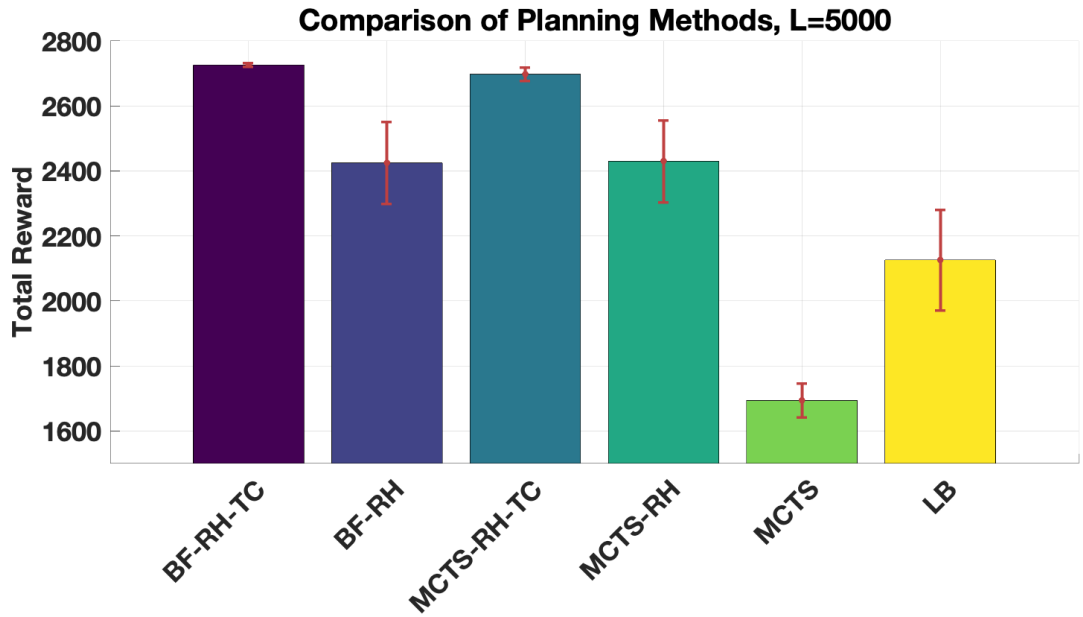
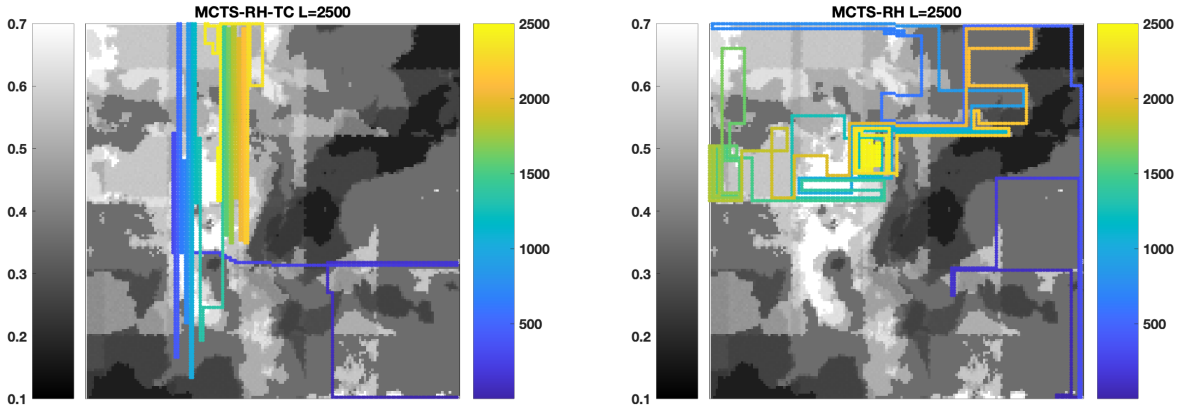



Figure 4.2: Comparison of the resulting total accumulated reward for each of the planning approaches described in this paper where the mission length $l$ is varied.




bl

Figure 4.3: Results for the presented planners where the length $l$ is set to 5000.




(a) (b)

Figure 4.4: Representative Search Path Using **MCTS-RH-TC** (a) and **MCTS-RH** (b) . The grayscale values indicate the reward for searching an area. The colored line represents the search path, the start at step 0 (dark blue) and the end at step 2500 (yellow).

# Chapter 5

# Multi-agent Receding Horizon Search with Terminal Rewards

We address the problem of planning paths for a team of autonomous search agents. As in Chapters 3 and 4, we specifically consider the case that the search area is large enough that an exhaustive search of the environment is likely not possible and the search mission is long enough that mission-length optimal paths cannot feasibly be computed in real time. The primary contribution if this chapter is the extension of formal guarantees presented in Chapters 3 and 4 to the multi-agent case. Multi-agent path planning presents a significant challenge as the computational complexity associated with joint path planning generally compounds as team size increases. This computational complexity prompts the use of sequential planning methods and sampling based planning methods, which allow teams of agents to plan nearly optimal paths in a computationally feasible manner. Significantly, we show, through numerical experiments, that the formal guarantees presented in this chapter may be achieved using a variety of planning methods which demonstrates the flexibility of our proposed methods.

Sequential planning methods facilitate joint path planning and allow for the use of efficient single agent planning methods such as those presented in [2, 16, 62]. Sampling based methods such as those presented in [34, 38] allow agents to plan long paths (often mission length) and generally perform well given enough time with [38] eventually converging to an optimal solution. The Dec-MCTS algorithm [5] (a decentralized variant of MCTS) allows multiple

agents to plan near-optimal joint paths within a long planning horizon and performs well even in complex environments [67]. The authors of [36] construct a receding horizon coverage path by using what amounts to a special case of the Dec-MCTS algorithm to plan joint paths within a short planning horizon for a team of robots. The results presented in this chapter draw from the results presented Chapter 4 where a single agent utilizes a slightly modified version of MCTS within a short planning horizon and appends a terminal reward. We extend the results of Chapter 4 to the multi-agent case and demonstrate the efficacy of the methods in the context of subsea search.

## 5.1 Problem Description

We consider that $K \geq 1$ agents have been tasked to search a region $G \subset \mathbb{R}^2$ where $G$ is composed of $q$ disjoint cells $h_i$ such that $\bigcup_{i=0}^{q} h_i = G$ with $h_i \bigcap h_j = \emptyset$ for $i \neq j$. The goal of the search agents is to solve for a joint path that maximizes the reward attained by the agents while traversing the path.

Because we consider several agents, we adopt the notation $s_i^k$ to represent the state of agent $k$, $s_i^{(k)}$ to represent the joint state of all agents not agent $k$, and $s$ to represent the joint state of all agents. The subscript $i$ denotes the time step associated with the state. The position of a search agent within the search region $G$ at any given time step is contained in the agent's state variable $s_i^k$.

We define an $l$-length path for agent $k$ as a feasible sequence of states $\gamma_n^k(s_0^k) = \{s_0^k, s_1^k, \ldots, s_l^k\}$. A sequence is feasible if there is some action $a$ available to agent $k$ such that applying $a$ while in state $s_i^k$ results in the agent transitioning to state $s_{i+1}^k$. The set of all possible $l$-length paths available to agent $k$ beginning at state $s_i^k$ is denoted by $\Gamma_l^k(s_0^k)$. The set of all $l$-length joint paths beginning from the joint state $s_0$ is then $\Gamma_l(s_0) = \Gamma_l^1(s_0^1) \times \ldots \times \Gamma_l^K(s_0^K)$.

Defining joint states and paths in this manner, we are able to utilize the notation presented in Chapters 2 and 3 with the understanding that state and path arguments represent joint states and paths unless indicated otherwise. As in Chapters 3 and 4, the goal of the team of search agents is to maximize the path value function given in equation 2.1.

## 5.2 Sub-Optimality Guarantees for Receding Horizon Search with Terminal Rewards

In Chapter 3 it was shown that incorporating a terminal reward defined as a lower bound on the value-to-go in the construction of a receding horizon path can provide desirable lower bound guarantees on the value of the receding horizon path produced given that paths are planned optimally within the planning horizon. In Chapter 4 the optimality constraint of Chapter 3 was relaxed and two tests were provided that, if satisfied at each planning step in the construction of a receding horizon path using (3.5) as the path value function, provide a lower bound on the value of the receding horizon path produced. This work expands the results of Chapters 3 and 4 to the case of multiple coordinated agents. The tests and subsequent guarantees provided in Chapter 4 are extended to the multi-agent case and presented in propositions 5.1 and 5.2.

We say that a joint path $\gamma'_n(s_i)$ is near-optimal if the value of an optimal path is given by $\mathcal{V}^*_n(s_i)$ and the value of $\gamma'_n(s_i)$ satisfies

$$\mathcal{V}'_n(s_i) \geq \mathcal{V}^*_n(s_i) - \epsilon \tag{5.1}$$

for some $\epsilon \geq 0$. Assuming that a a team of agents is capable of planning near-optimal paths within the planning horizon, proposition 5.1 holds.

**Proposition 5.1.** *Suppose the $l$-length joint receding horizon path $\boldsymbol{P}_l(s_0)$ is composed of $m$-step sequences along a sequence of $n$-length near-optimal joint paths that seek to maximize (3.5) satisfying (5.1). For every $n$-length near-optimal path*

$$\gamma'_n(s_i) = \{s_i, s'_{i+1}, \ldots, s'_{i+n}\} \tag{5.2}$$

*suppose that there exists a set of joint states $\{\tilde{s}_{i+n+1}, \ldots, \tilde{s}_{i+n+m}\}$ that are feasible from $s'_{i+n}$ such that*

$$B_l(s'_{i+n}) + \epsilon \leq \sum_{j=i+n+1}^{i+n+m} g(\tilde{s}_j) + B_l(\tilde{s}_{i+n+m}) \tag{5.3}$$

*then the RH path satisfies $J(\boldsymbol{P}_l(s_0)) \geq B_l(s_0)$.*

Noting that $\epsilon$ may be difficult to characterize and a set of states satisfying (5.3) may be difficult to find as $\epsilon$ grows large, Proposition 5.2 provides a test against the remainder of the path found at the previous planning step. As such, it is not required that paths be near-optimal within the planning horizon. Instead, paths within the planning horizon should have a reward no less than the remainder of the path planned at the previous step.

**Proposition 5.2.** *Suppose that at each planning step the joint path $\gamma'_n(s_i)$ satisfies*

$$\mathcal{V}'_n(s_i) \geq \mathcal{V}'_{n-m}(s_i) \tag{5.4}$$

*where $\mathcal{V}'_{n-m}(s_i)$ is the value of the $n-m$ steps not included in the execution horizon of $\gamma'_n(s_{i-m})$ and that*

$$\mathcal{V}'_n(s_0) \geq B_l(s_0) \tag{5.5}$$

*then the RH path satisfies*

$$J(\boldsymbol{P}_l) \geq B_l(s_0). \tag{5.6}$$

We note that propositions 5.1 and 5.2 build upon the corresponding results in Chapter 4 by directly addressing the multi-agent case. As such, the proofs for propositions 5.1 and 5.2 can be directly adapted from the corresponding proofs in Chapter 4, and are omitted herein. Also of note, any naive, readily available joint path may be used to provide a lower bound on the value-to-go as in Chapters 3 and 4.

## 5.3 Receding Horizon Search Path Planning with Sequentially Allocated Optimal Paths

The computational complexity associated with solving for optimal or near-optimal joint paths motivates the use of sequentially allocated optimal paths, which are jointly suboptimal but require computational complexity that increases linearly with the number of search agents instead of exponentially, which is generally the case. To demonstrate that this well known joint path planning method is easily adapted to our proposed framework, we present a variant of sequentially allocated optimal paths (Algorithm 1) that provably satisfies the requirements of proposition 5.2 (Proposition 5.3) and thus ensures a guaranteed lower bound on the value of joint paths produced while retaining the computational benefits associated with the method.

**Proposition 5.3.** *At every time step $i$, the joint path $\gamma_n(s_i)$ on line 8 of Algorithm 1 satisfies the hypotheses of Proposition 5.2.*

*Proof of Proposition 5.3.* The joint path $\gamma_n(s_0)$ for all agents is initially chosen as the first $n$-step path along the $l$-length naive path used to determine the value-to-go. Therefore at line 1 of Algorithm 1 we have $\mathcal{V}(\gamma_n(s_0)) = B_l(s_0)$. Agent 1 solves for single agent $n$-length path satisfying (5.8). We then have the joint path $\tilde{\gamma}_n^*(s_0)$ composed of $\gamma_n^{1*}(s_0^1)$ and $\gamma_n^{(1)}(s_0^{(1)})$. From lemma 5.4, we have that $\mathcal{V}(\tilde{\gamma}_n^*(s_0)) \geq \mathcal{V}(\gamma_n(s_0))$ and therefore $\mathcal{V}(\tilde{\gamma}_n^*(s_0)) \geq B_l(s_0)$. Agent 1

then replaces the corresponding single agent path in $\gamma_n(s_0)$ with the new path $\gamma_n^{k*}(s_0^k)$ such that $\gamma_n(s_0) = \tilde{\gamma}_n^*(s_0)$ and therefore $\mathcal{V}(\gamma_n(s_0)) \geq B_l(s_0)$. Repeating the process of solving for a path satisfying (5.8) and setting $\gamma_n(s_0) = \tilde{\gamma}_n^*(s_0)$ for each agent, we ensure that $\gamma_n(s_0)$ always satisfies $\mathcal{V}(\gamma_n(s_0)) \geq B_l(s_0)$ at the first planning step.

At step $i > 0$, let $\gamma_{n-1}(s_i)$ be the $(n-1)$-length path remaining of the joint path found at the previous step. Let $\gamma_n(s_i)$ be the $n$-length path constructed by appending the first step along the naive path used to determine the value-to-go from the end of $\gamma_{n-1}(s_i)$ to the end of $\gamma_{n-1}(s_i)$ (Algorithm 1 line 4). Then $\mathcal{V}(\gamma_n(s_i)) = \mathcal{V}(\gamma_{n-1}(s_i))$. Agent 1 solves for a single agent $n$-length path satisfying (5.8) and replaces the single agent path in $\gamma_n(s_i)$ associated with agent 1 such that $\mathcal{V}(\gamma_n(s_i)) \geq \mathcal{V}(\gamma_{n-1}(s_i))$. Repeating the process for all other agents in order, we ensure that $\mathcal{V}(\gamma_n(s_i)) \geq \mathcal{V}(\gamma_{n-1}(s_i))$ is always satisfied by lemma 5.4. □

Suppose that the lower bound on the value-to-go is defined using a readily available, naive joint path. We denote this naive joint path as $\bar{\gamma}_n(s_0)$. Then

$$J(\bar{\gamma}_n(s_0)) = B_l(s_0). \tag{5.7}$$

**Lemma 5.4.** *Given any joint path $\gamma_n(s_i)$, let $\tilde{\gamma}_n(s_i)$ be the joint path composed of the single agent path $\gamma_n^k(s_i^k)$ for agent $k$ and the joint path $\gamma_n^{(k)}(s_i^{(k)})$ for all agents excluding agent $k$. Let the optimal path for agent $k$ satisfy*

$$\gamma_n^{k*}(s_i^k) = \underset{\gamma_n^k(s_i^k) \in \Gamma_n^k(s_i^k)}{\arg\max} \ \mathcal{V}(\tilde{\gamma}_n(s_i)). \tag{5.8}$$

*Then the path $\tilde{\gamma}_n^*(s_i)$ composed of the optimal path for agent $k$ and the joint path $\gamma_n^{(k)}(s_i^{(k)})$*

*for all agents excluding agent $k$ satisfies*

$$\mathcal{V}(\tilde{\gamma}_n^*(s_i)) \geq \mathcal{V}(\gamma_n(s_i)). \tag{5.9}$$

*Proof.* Recall that $\gamma_n(s_i)$ is composed of some path $\gamma_n^k(s_i^k)$ for agent $k$ and a joint path $\gamma_n^{(k)}(s_i^{(k)})$ for all agents excluding agent $k$. Because $\tilde{\gamma}_n^*(s_i)$ is found by considering all possible paths for agent $k$, the case $\tilde{\gamma}_n(s_i) = \gamma_n(s_i)$ must be considered. Because $\gamma_n^k(s_i^k)$ does not necessarily maximize the reward attained, we conclude that the value of the joint path $\gamma_n(s_i)$ provides a lower bound on the value of the joint path $\tilde{\gamma}_n^*(s_i)$. □

**Algorithm 1:** Receding Horizon Search Path Planning with Sequentially Allocated Optimal Paths

```
1  γ_n(s_0) = γ̄_n(s_0);
2  while Search mission is not complete do
3      if i > 0 then
4          γ_n(s_i) = extend(γ_{n-1}(s_i));
5      end
6      for k = 1 : K do
7          Solve for γ_n^{k*}(s_i^k) given γ_n(s_i) excluding the single agent path for agent k;
8          γ_n(s_i) = γ̃_n^*(s_i);
9      end
10     transition from s_i to s_{i+1};
11 end
```

Algorithm 1 is executed as follows. At the start of a search mission the team of search agents is initialized with a $n$-length joint path defined as the first $n$ steps along the naive mission-length path used to define the lower bound on the value-to-go (Line 1). This may be accomplished by having each agent broadcast the first $n$-step segment of a naive single-agent path feasible from that agent's current state. Agent 1 solves for a single agent path that maximizes the joint reward of the joint path composed of agent 1's path with the naive joint path for all other agents (Line 7). Agent 1 communicates the new path to all other agents

replacing the corresponding single agent path in the previous joint path (Line 8). Agent 2, in turn, solves for a single agent path that maximizes the joint reward of the joint path composed of the single agent path found by agent 1, agent 2's path, and the naive joint path for all other agents 3 to $K$ (Line 7). Agent 2 then communicates the new path to all other agents (Line 8). Agents 3 to $K$ follow the same procedure in order. Once agent $K$'s path is selected, all agents transition to the next state (Line 10). The process of solving for an $n$-length joint path and taking a step or steps along that path is repeated until the mission is complete. After transitioning states, the first step along the naive path used to determine the lower bound on the value-to-go from the final step along the path found at the previous planning epoch is appended to the path found at the previous planning epoch to ensure that the naive path is $n$-steps long. This step may be executed by each agent in parallel under the assumption that paths are broadcast to all agents in the network.

## 5.4 Numerical Experiments

The search environment used in the numerical experiments presented in this chapter is the $150 \times 150$ search grid shown in Figure 5.1. The search environment was generated from data collected at sea during AUV trials in Boston Harbor and represents estimates of varying bottom types that are characterized by their acoustic reflectivity. In the context of sub-sea search using acoustic sensors, the bottom type impacts the sensor performance; environments that are too absorptive or reflective degrade the sensor's ability to search for objects of interest. In this chapter we generalize the performance into three distinct difficulties: easy, medium, and hard. This information is then wrapped into an expected reward for searching a cell based on the sensor performance, see [53] for additional details. In Figure 5.1 we show the expected reward for searching a cell once. The actions available to each agent are *go*

*straight*, *turn right*, and *turn left*. Side-scan sonar works best when a search agent tracks straight line paths. We assume that no valid measurements are acquired during turning maneuvers, and that turning maneuvers yield zero reward.

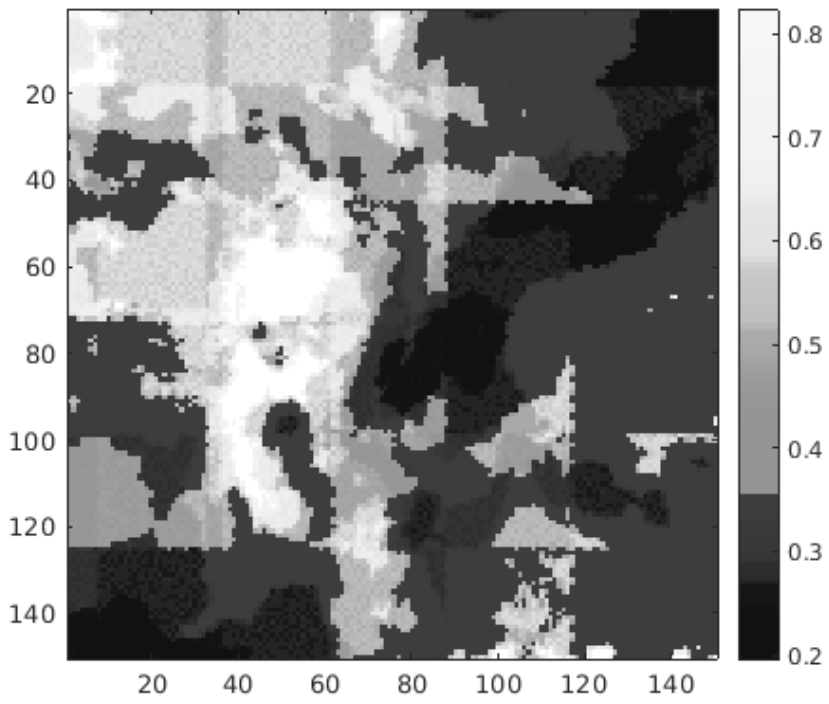



Figure 5.1: 150 × 150 search grid representing the benefit of search given a single visit to each cell.

We compare four receding horizon path planning methods: sequentially allocated optimal paths with terminal rewards (Algorithm 1) which satisfies the hypotheses of proposition 5.2, decentralized Monte Carlo Tree Search (dec-MCTS) with terminal rewards that seeks to maximize (3.5) directly and is motivated by proposition 5.1, sequentially allocated optimal paths with no terminal rewards, and dec-MCTS with no terminal rewards. In each case, the planner solves for paths within an $n$-length planning horizon. We specifically consider the dec-MCTS planner as it is robust to imperfect communication [5] whereas sequential path planning methods generally are not. In each case, the search agents have a planning

horizon of $n = 6$ and terminal rewards are defined using a set of naive lawnmower paths. Any single lawnmower path begins at a single agent's current state and terminates at the maximum mission length for the associated agent. The vertical or horizontal orientation of any lawnmower path is decided by the orientation of the agent's state where the lawnmower path begins. To plan sequentially allocated optimal paths, each agent enumerates all possible $n$-length paths for itself and selects a path that maximizes the reward of the joint path composed of the single agent path and all previously determined single agent paths or, in the case of Algorithm 1, all previously determined single agent paths and the naive $n$-length joint path composed of single agent lawnmower paths for all other agents. An example of a path planned using Algorithm 1 is shown in Figure 5.3a. An example of a path planned using dec-MCTS with terminal rewards is shown in Figure 5.3b. For the dec-MCTS planners, agents are permitted 5 seconds to plan at each step.

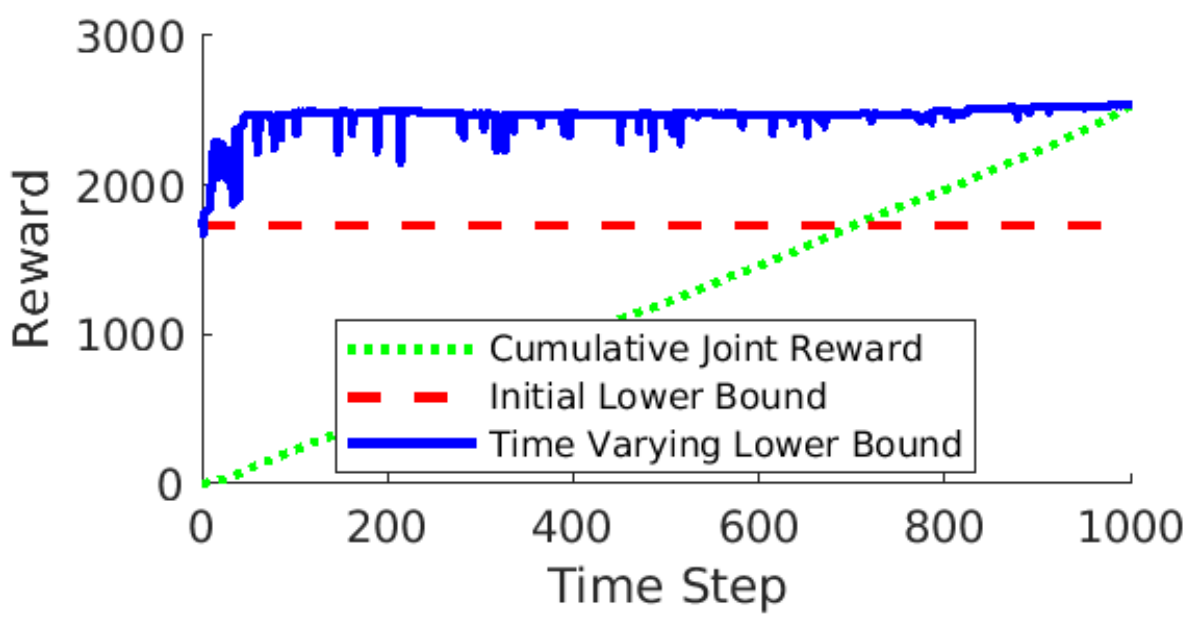



Figure 5.2: Cumulative reward and lower bounds for the path shown in Figure 5.3a

Figure 5.2 shows the reward associated with the path shown in Figure 5.3a. The dotted green line gives the cumulative reward attained by the search agents attained at any time

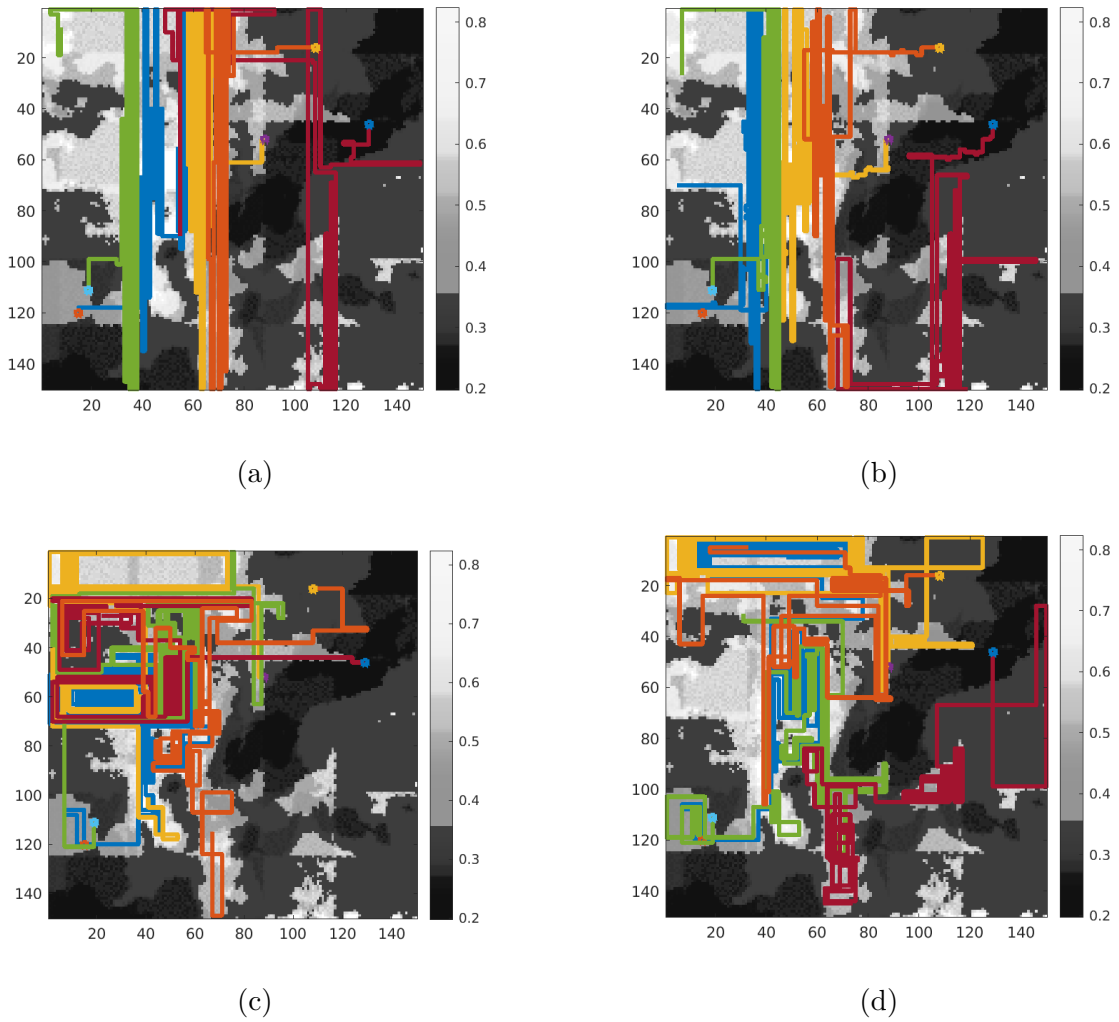



Figure 5.3: Examples of receding horizon paths produced by 5 agents using Algorithm 1 with terminal rewards (a), dec-MCTS with terminal rewards (b), sequentally allocated optimal paths with no terminal rewards (c), and dec-MCTS with no terminal rewards (d).

step. The initial lower bound $B_l(s_0)$ is shown as the red dashed line. The solid blue line gives a time varying lower bound at any given time step given as the cumulative reward plus a lower bound on the value-to-go from the joint state at a given time step. That is, at any time step $i$ the new lower bound $r_i$ is given by the equation $r_i = J(\boldsymbol{P}_i) + B_l(s_i)$. Notably, at any time step $i$ the lower bound on the value of the receding horizon path $\boldsymbol{P}_{l-i}$ is given by $B_l(s_i)$. As such, we may expect the total accumulated reward to not only exceed the initial lower bound $B_l(s_0)$ in expectation but also to satisfy $J(\boldsymbol{P}_l) \geq \max_i r_i$ for $i = 0, \ldots, l$ where

$l$ is the total mission length. Indeed, we see this in this single example shown in Figure 5.2 with results consistent for all receding horizon paths planned using Algorithm 1.

Results of missions planned using each planning method are shown in Figure 5.4. Paths were planned using each method with 1,3, and 5 agent teams assuming perfect communication. For dec-MCTS, messages contained a single path[1] despite perfect communication assumptions. To emphasise the value of using multiple agents, the sum of the mission lengths for all agents amounts to 5010 in every case. That is, for a team with only one agent the single agent traverses a receding horizon path of length 5010 whereas, in a team of 5 agents, each agent traverses only 1002 steps. Thus, Figure 5.4 shows that a larger team of search agents achieves similar results to a single agent performing a long search mission in a fraction of the time. In each case, 50 paths were planned with random initial joint states. We note that the methods that incorporate terminal rewards perform comparably to the methods that do not incorporate terminal rewards with the additional benefit of a guaranteed lower bound on performance.

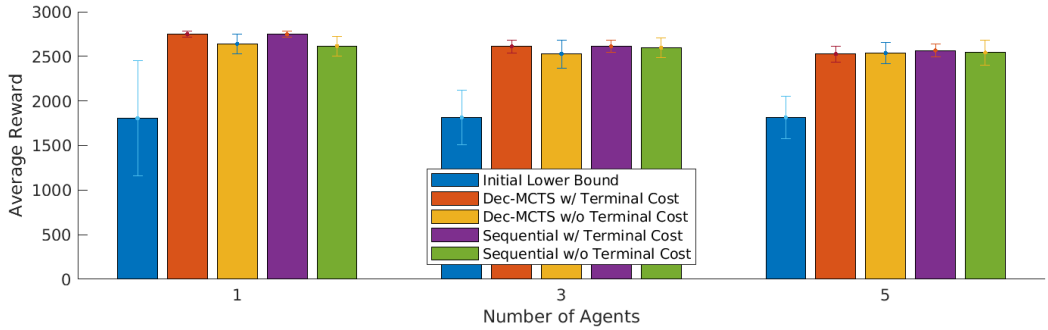



Figure 5.4: Average path values for all path planners given differing numbers of search agents.

[1]This is significant as large messages are not feasible when communicating via acoustic channels where reliable communications are typically limited to approx. 32 byte packets of data.

# Chapter 6

# Non-Submodular Maximization via the Greedy Algorithm and the Effects of Limited Information in Multi-Agent Execution

We provide theoretical bounds on the worst case performance of the greedy algorithm in seeking to maximize a normalized, monotone, but not necessarily submodular objective function under a general matroid constraint. We also provide worst case bounds on the performance of the greedy algorithm in the case that limited information is available at each planning step. We specifically consider limited information as a result of unreliable communications during distributed execution of the greedy algorithm. We utilize notions of curvature for normalized, monotone set functions to develop the bounds provided in this chapter. To demonstrate the value of the bounds provided in this work, we analyze a variant of the benefit of search objective function and show, using real-world data collected by an autonomous underwater vehicle, that theoretical approximation guarantees are achieved despite non-submodularity of the objective function.

## 6.1 Introduction

In general, the problem of planning search paths that seek to maximize a general objective function is NP-hard. One simple method of addressing the general infeasability of planning optimal paths for a team of search agents is to utilize the *greedy algorithm* [15] wherein an ordering is assigned to the set of search agents and each agent plans a path for itself while accounting for the paths of preceding agents. We seek to provide theoretical approximation guarantees for the greedy algorithm in seeking to maximize a non-submodular objective function where planning at each step of the greedy algorithm is potentially suboptimal. Furthermore, we consider the effect of unreliable communications during distributed execution of the greedy algorithm which limits the information available to search agents and seek to provide approximation guarantees for the greedy algorithm with limited information as well.

The greedy algorithm has received significant attention because of its practical simplicity. In addition to its ease of implementation, the greedy algorithm has been shown to yield results with high-quality approximation guarantees under a wide variety of constraints on the possible solutions that the greedy algorithm may produce.

It is shown in [22] that the greedy algorithm yields an optimal solution given a modular objective function. The well known 1/2 lower bound is presented in [23] for the case of a normalized, monotone, submodular objective function under a general matroid constraint. The authors of [15] introduce a notion of curvature for normalized, monotone, submodular objective functions and use this curvature $\alpha_c \in [0, 1]$ to provide an approximation guarantee of $1/(1 + \alpha_c)$ that is equivalent to the bounds presented in [22] and [23] at the limits of the curvature term (when $\alpha = 0$ or $\alpha = 1$) thus unifying the results of [22] and [23]. It is shown in [55] that under a uniform matroid constraint, the greedy algorithm yields an improved bound of $(1 - e^{-1}) \approx 0.63$ given a submodular objective function. In [15], curvature is again

used to further improve this bound to $(1 - e^{-\alpha})/\alpha$.

In an effort to retain the $(1 - e^{-1})$ approximation guarantee of [55] under a general matroid constraint, the authors of [12] present a randomized algorithm that retains the $(1 - e^{-1})$ approximation guarantee of [55] in expectation for submodular objective functions. This advancement has prompted the development of a host of randomized algorithms that provide desirable performance bounds [14, 43, 49, 61]. In this work, we consider only the greedy algorithm and the worst case effects of limited communication on the performance of the greedy algorithm.

The bounds of [23] and [55] have motivated the frequent use of submodular objective functions such as mutual information which is known to be submodular assuming measurements are conditionally independent [43]. Further consideration of approximation guarantees for the greedy algorithm in seeking to maximize a non-submodular objective function has begun only recently. We refer the reader to [49] for a more thorough discussion of distinct types of non-submodular functions and recent advancements. Important applications with corresponding non-submodular objective functions include experimental design, dictionary selection, and subset selection [49]. The work of [28] addresses visibility optimization in social media. In this work, we address robotic search.

**Contributions:** In the first contribution of this work, which appears in Theorem 6.6, we provide suboptimality guarantees for the greedy algorithm under a general matroid constraint where the objective function is normalized and monotone, but not necessarily submodular. We discuss how our novel bound generalizes and extends bounds provided in earlier works with the additional consideration of generalized curvature. In the second contribution of our work, which appears in Theorem 6.7, we provide suboptimality guarantees for the greedy algorithm with limited information extending the work of [29, 31]. Our extensions include the consideration of normalized, monotone, but not necessarily submodular objective functions.

Additionally, in each case we consider that the greedy selection step is executed with bounded suboptimality as in [64]. Finally, we illustrate the efficacy of our contributions with respect to a robotic search application.

Most closely related to the first contribution of this work are the results presented in [57] and [28] which show that under a general matroid constraint the greedy algorithm yields a constant approximation factor $(1-\beta)/(1+(1-\beta))$ for normalized, monotone, but not necessarily submodular objective functions where $\beta$ is equivalent to the inverse generalized curvature given in Definition 6.5 of our work. Our work extends the bounds of [28, 57] by further considering the generalized curvature given in Definition 6.4. Our contribution given as Theorem 6.6 has a form very similar to that of the bound presented in [3]. In contrast to [3] where the authors consider the sum of submodular and supermodular functions and utilize notions of curvature with respect to these functions, we consider a general non-submodular objective function. This is significant because a decomposition of a normalized monotone set function into a sum of submodular and supermodular functions is not guaranteed to exist as shown in Lemma 3.2 of [3]. Additionally, we consider that the greedy selection step is potentially suboptimal as in [64] further distinguishing our contribution.

Most closely related to the second contribution of this work are the results in [29] and [31] where a generalized greedy algorithm is presented that does not require complete information regarding preceding decisions. Notably, the generalized greedy algorithm in [31] enjoys a worst-case approximation guarantee of $\frac{1}{1+k^*(G)}$ under a partition matroid constraint assuming a normalized, monotone, submodular objective function. The term $k^*(G)$ represents the fractional clique cover number of the underlying communication graph $G$ and is discussed in detail in [31]. We extend the work of [31] by considering the curvature of the objective function which is assumed to be normalized and monotone, but not necessarily submodular.

## 6.2 Mathematical Preliminaries

Let $\mathcal{V}$ be a finite ground set and let $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ be a set function where $2^{\mathcal{V}}$ denotes the power set of $\mathcal{V}$. Generally, calligraphic font is used to denote sets. Exceptions should be clear from context. We consider a team of $N$ agents given as $\mathcal{A} = \{1, \ldots, N\}$. In executing the greedy algorithm, each agent $i$ will select an element $x_i$ from the ground set $\mathcal{V}$ in an effort to maximize the objective function $f$.

When analyzing suboptimality of the greedy algorithm, one often encounters matroids which are used to represent constraints on the solutions produced by the greedy algorithm. A matroid is defined as follows.

**Definition 6.1** (Matroid [63, Definition 39.1] )**.** Consider a finite ground set $\mathcal{V}$, and a non-empty collection of subsets of $\mathcal{V}$, denoted by $\mathcal{I}$. Then, the pair $(\mathcal{V}, \mathcal{I})$ is called a matroid if and only if the following conditions hold:

i ) for any set $\mathcal{X} \subseteq \mathcal{V}$ such that $\mathcal{X} \in \mathcal{I}$, and for any set $\mathcal{Z} \subseteq \mathcal{X}$, it holds $\mathcal{Z} \in \mathcal{I}$;

ii ) for any sets $\mathcal{X}, \mathcal{Z} \subseteq \mathcal{V}$ such that $\mathcal{X}, \mathcal{Z} \in \mathcal{I}$ and $|\mathcal{X}| < |\mathcal{Z}|$, it holds that there exists an element $z \in \mathcal{Z} \setminus \mathcal{X}$ such that $\mathcal{X} \cup \{z\} \in \mathcal{I}$.

Matroids are used to represent abstract dependence. The set $\mathcal{I}$ contains all independent sets of $\mathcal{V}$. A set $x \in \mathcal{I}$ is called maximal if there exists no $v \in \mathcal{V}$ such that $x \cup \{v\} \in \mathcal{I}$. For example, any subset of the columns of a matrix are either linearly independent or dependent. A matroid may be constructed by allowing the columns of the matrix to form the ground set $\mathcal{V}$ and every combination of linearly independent columns to form $\mathcal{I}$. An excellent discussion of matroids and their history is provided in [63]. Most importantly for our purposes, matroids provide a construct to represent constraints in planning. Two

notable special types of matroids are uniform matroids [6, 15, 55] and partition matroids [17, 18, 24, 31, 64].

To analyze suboptimality bounds for the greedy algorithm, we utilize the *marginal reward*.

**Definition 6.2** (Marginal Reward)**.** Given any sets $\mathcal{S}, \mathcal{Q} \subseteq \mathcal{V}$ the marginal contribution of $\mathcal{S}$ given $\mathcal{Q}$ is

$$\Delta(\mathcal{S}|\mathcal{Q}) \triangleq f(\mathcal{S} \cup \mathcal{Q}) - f(\mathcal{Q}). \tag{6.1}$$

We adopt notation for the marginal reward from [29] and [31]. It may be useful to consider the marginal reward as the discrete derivative of $f$ as discussed in [17].

Note that given an arbitrarily ordered set $\mathcal{S} \subseteq \mathcal{V}$, the reward attained by $\mathcal{S}$ may be represented as the sum of the marginal rewards for the elements of $\mathcal{S}$. For example, given $\mathcal{S} = \{s_1, \ldots, s_M\}$, we have

$$f(\mathcal{S}) = \sum_{i=1}^{M} \Delta(\{s_i\} | \cup_{j=1}^{i-1} \{s_j\}) \tag{6.2}$$

where $\cup_{j=1}^{i-1}\{s_j\}$ is the empty set when $i = 1$. For convenience, we use the subscript notation $s_{1:i-1} = \cup_{j=1}^{i-1}\{s_j\}$ to denote a subset of an ordered set. We also often use a comma in place of the union operator, e.g. $\Delta(\mathcal{S}|\mathcal{Q}, \mathcal{Z}) = \Delta(\mathcal{S}|\mathcal{Q} \cup \mathcal{Z})$. Lastly, when representing a single element in a set, we often drop the curly brace notation, e.g. $\Delta(\{s_i\}|\mathcal{Q}) = \Delta(s_i|\mathcal{Q})$.

By (6.2), given sets $\mathcal{S}, \mathcal{Q} \subseteq \mathcal{V}$ with $|\mathcal{S}| = M_{\mathcal{S}}$ and $|\mathcal{Q}| = M_{\mathcal{Q}}$ and arbitrary orderings $\mathcal{S} = \{s_1, \ldots, s_{M_{\mathcal{S}}}\}$ and $\mathcal{Q} = \{q_1, \ldots, q_{M_{\mathcal{Q}}}\}$ we have

$$f(\mathcal{S}, \mathcal{Q}) = f(\mathcal{S}) + \sum_{i=1}^{M_{\mathcal{Q}}} \Delta(q_i | q_{1:i-1}, \mathcal{S}) \tag{6.3}$$

$$f(\mathcal{S}, \mathcal{Q}) = f(\mathcal{Q}) + \sum_{i=1}^{M_{\mathcal{S}}} \Delta(s_i | s_{1:i-1}, \mathcal{Q}). \tag{6.4}$$

For the case of informative path planning, let $\mathcal{X}_i$ represents the set of paths available to agent $i$ and let $\mathcal{V} = \cup_{i\in\mathcal{A}}\mathcal{X}_i$. We define the constraint $\mathcal{I} = \{\mathcal{S} \subseteq \mathcal{V} : |\mathcal{S} \cap \mathcal{X}_i| \leq 1 \ \forall i \in \mathcal{A}\}$, which is a simple partition matroid. Any element of $\mathcal{I}$ gives a joint path for a set of agents $a \subseteq \mathcal{A}$ and maximal set in $\mathcal{I}$ gives a complete joint path. That is, for a maximal set $x \in \mathcal{I}$, $x$ satisfies $|x| = |\mathcal{A}|$. Meaning each agent in $\mathcal{A}$ contributes to the joint path $x$. Let $x, x^* \in \mathcal{I}$ be maximal independent sets in $\mathcal{I}$. Then replacing $\mathcal{S}$ and $\mathcal{Q}$ in (6.3) with $x$ and $x^*$ gives us

$$f(x, x^*) = f(x^*) + \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}, x^*) \tag{6.5}$$

$$f(x, x^*) = f(x) + \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) \tag{6.6}$$

which, in turn, gives

$$f(x^*) = f(x) + \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) - \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}, x^*). \tag{6.7}$$

While equality in (6.7) holds for any maximal $x, x^* \in \mathcal{I}$, $x^*$ is assumed to satisfy $x^* \in \underset{\mathcal{S}\in\mathcal{I},|\mathcal{S}|=N}{\arg\max} f(\mathcal{S})$ while $x$ is assumed to be a solution produced by the greedy algorithm.

In addition, we consider the case that limited information is available to an agent at its planning turn as in [29, 31]. More precisely, the standard greedy algorithm relies on the plans of agents 1 through $i-1$ to be available to agent $i$ when it plans. We consider that only a subset of those plans $\mathcal{N}_i \subseteq \{j \in \mathcal{A} : j < i\}$ is available to agent $i$ when planning. Thus, for informative path planning, we denote the incomplete joint plan available to agent $i$ with $x_{\mathcal{N}_i}$. As a note: consider that each element of a set $\mathcal{S} \subseteq \mathcal{V}$ is represented by a node in a graph $G$ and let the in-neighbors of each node $s_i$ be the set of nodes $\{s_j : 1 \leq j < i\}$ such that the $G$ is a directed acyclic graph. Equality in (6.2) is only guaranteed when $G$ is complete, i.e. $\mathcal{N}_i = \{j \in \mathcal{A} : j < i\}$. Specifically for a subset $\mathcal{S}$ of nodes, (6.2) holds when

the subgraph induced by $\mathcal{S}$ is complete, i.e. when $\mathcal{S}$ is a clique. We refer the reader to [29] for a detailed discussion of the underlying communication graph structure.

## 6.3 Properties of the Objective Function

While we have imposed no constraints on the objective function $f$ in Section 6.2, many useful results regarding the greedy algorithm rely on $f$ possessing several properties most common of which are the following:

i ) *Normalized:* $f(\emptyset) = 0$

ii ) *Monotone:* For $\mathcal{S} \subseteq \mathcal{Q} \subseteq \mathcal{V}$, $f(\mathcal{S}) \leq f(\mathcal{Q})$.

iii ) *Submodular:* For $\mathcal{S} \subseteq \mathcal{Q} \subset \mathcal{V}$ and $v \in \mathcal{V} \setminus \mathcal{Q}$, the following holds:

$$f(\mathcal{S} \cup \{v\}) - f(\mathcal{S}) \geq f(\mathcal{Q} \cup \{v\}) - f(\mathcal{Q}). \tag{6.8}$$

Note that property iii (submodularity) provides a relationship between marginal rewards and is equivalently written as $\Delta(v|\mathcal{S}) \geq \Delta(v|\mathcal{Q})$.

The authors of [15] introduce a notion of curvature of submodular set functions that they use to unify distinct suboptimality guarantees for the greedy algorithm. Their definition of curvature, denoted by $\alpha_c$, is defined as follows.

**Definition 6.3.** (Total Curvature [15]) Let $f$ be normalized, monotone, and submodular. Then the curvature of $f$ denoted by $\alpha_c$ is defined as

$$\alpha_c \triangleq \max_{v \in \mathcal{V}^*} \frac{\Delta(v|\emptyset) - \Delta(v|\mathcal{V} \setminus v)}{\Delta(v|\emptyset)} \tag{6.9}$$

where $\mathcal{V}^* \triangleq \{v \in \mathcal{V} : f(v) \geq 0\}$.

In our analysis, we utilize a more general notion of curvature given in [8].

**Definition 6.4** (Generalized Curvature [8, definition 2])**.** Consider a normalized, monotone function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}_{\geq 0}$. The generalized curvature is the smallest scalar $\alpha$ s.t.

$$\Delta(v|(\mathcal{S} \cup \mathcal{Q}) \setminus v) \geq (1 - \alpha)\Delta(v|\mathcal{S} \setminus v) \tag{6.10}$$

for all $\mathcal{S}, \mathcal{Q} \subseteq \mathcal{V}$ and $v \in \mathcal{S} \setminus \mathcal{Q}$.

Proposition 2 in [24] gives that, for submodular, monotone functions, $\alpha$ given in Definition 6.4 satisfies $\alpha \leq \alpha_c$ where $\alpha_c$ is given by Definition 6.3.

To analyze the suboptimality of the greedy algorithm given a non-submodular objective function, we utilize the definition of inverse generalized curvature presented in [8].

**Definition 6.5** (Inverse Generalized Curvature [8, definition 2])**.** Consider a normalized, monotone function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}_{\geq 0}$. The inverse generalized curvature is the smallest scalar $\beta$ s.t.

$$\Delta(v|\mathcal{S} \setminus v) \geq (1 - \beta)\Delta(v|(\mathcal{S} \setminus v) \cup \mathcal{Q}) \tag{6.11}$$

for all $\mathcal{S}, \mathcal{Q} \subseteq \mathcal{V}$ and $v \in \mathcal{S} \setminus \mathcal{Q}$. The function is submodular *iff* $\beta = 0$ and modular *iff* $\beta = \alpha = 0$.

In general $\alpha$ can be different from $\beta$.

## 6.4 Bounds for the Greedy Algorithm Given a Non-Submodular Objective Function

The main contributions of this work are given in Theorems 6.6 and 6.7. Throughout, we assume that the element of the solution contributed by agent $i$ or at the $i^{\text{th}}$ planning step of the greedy algorithm satisfies

$$x_i \in \{\bar{x}_i \in \mathcal{X}_i : \eta\Delta(\bar{x}_i|x_{\mathcal{N}_i}) \geq \max_{\hat{x}_i \in \mathcal{X}_i} \Delta(\hat{x}_i|x_{\mathcal{N}_i})\}. \tag{6.12}$$

An element $x_i$ satisfying (6.12) is called $\eta$-optimal.

We note that $\eta$-optimality at each planning step is generally not guaranteed in practice. However, the consideration of $\eta$-optimality here provides insight into the effects of using planning techniques, such as Monte Carlo Tree Search, that do not necessarily provide optimal solutions, but are likely nearly optimal. For example, each agent running Monte Carlo Tree search using the d-UCT proposed in [5] would allow agents to begin planning simultaneously and adapt plans as new information is gained until the planning window for each agent closes.

### 6.4.1 Suboptimality of the Greedy Algorithm given a Non-submodular Objective Function Under a General Matroid Constraint

**Theorem 6.6.** *Given a normalized, monotone objective function with $\alpha$ and $\beta$ given in Definitions 6.4 and 6.5, and assuming $\eta$-optimal planning, the greedy algorithm guarantees a solution $x$ satisfying*

$$\frac{f(x)}{f(x^*)} \geq \frac{1-\beta}{\eta + (1-\beta)\alpha}. \tag{6.13}$$

*Proof.*

$$f(x^*) = f(x) + \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) - \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}, x^*) \tag{6.14}$$

$$= \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}) + \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) - \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}, x^*) \tag{6.15}$$

$$= \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) + \sum_{i\in\mathcal{A}} [\Delta(x_i|x_{1:i-1}) - \Delta(x_i|x_{1:i-1}, x^*)] \tag{6.16}$$

$$\leq \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) + \alpha \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}) \tag{6.17}$$

where the inequality of (6.17) is by Definition 6.4.

Multiplying both sides of (6.17) by $1-\beta$ gives

$$(1-\beta)f(x^*) \leq (1-\beta) \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}^*, x) + (1-\beta)\alpha \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}) \tag{6.18}$$

$$\leq \sum_{i\in\mathcal{A}} \Delta(x_i^*|x_{1:i-1}) + (1-\beta)\alpha \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}) \tag{6.19}$$

$$\leq \eta \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}) + (1-\beta)\alpha \sum_{i\in\mathcal{A}} \Delta(x_i|x_{1:i-1}) \tag{6.20}$$

$$= \eta f(x) + (1-\beta)\alpha f(x) \tag{6.21}$$

where the second inequality is by Definition 6.5 and the third inequality is by $\eta$-optimality of the greedy selection. □

When $f$ is modular we have $\alpha = \beta = 0$. Thus, when $\eta = 1$ our bound reflects the optimality of the greedy algorithm proven in [22]. When $\beta = 0$ and $\eta = 1$, our bound improves the $1/(1+\alpha_c)$ bound of [15] by Proposition 2 of [24] which gives that $\alpha \leq \alpha_c$ where $\alpha$ is given by Definition 6.4. When $\beta = 0$ and $\alpha = 1$, our bound is equivalent to that of [64]. When $\alpha = 1$ and $\eta = 1$, our bound is equivalent to [28] and to the single matroid constrained case

of [57].

### 6.4.2 Suboptimality of the Greedy Algorithm with Limited Information given a Non-submodular Objective Function

Theorem 6.7 is inspired by distributed path planning in environments where communication is potentially unreliable. Generally, distributed execution of the greedy algorithm relies on each agent having access to the decisions of all preceding agents. Here we consider that unreliable communications have limited the information of each agent such that it must make a decision while having access to the decisions of only a subset of the preceding agents.

**Theorem 6.7.** *Given a normalized, monotone objective function with generalized curvature $\alpha$ and inverse generalized curvature $\beta$ given in Definitions 6.4 and 6.5, respectively, the greedy algorithm guarantees a solution $x$ satisfying*

$$\frac{f(x)}{f(x^*)} \geq \frac{(1-\beta)^2}{(1-\beta)^2 + (\alpha + \eta - 1 + \beta - \alpha\beta)k^*(G)} \tag{6.22}$$

*where $k^*(G)$ represents the fractional clique cover number of the underlying communication graph $G$.*

*Proof.*

$$f(x^*) = f(x) + \sum_{i \in \mathcal{A}} \Delta(x_i^* | x_{1:i-1}^*, x) - \sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}, x^*). \tag{6.23}$$

Multiplying both sides by $1 - \beta$ gives

$$(1-\beta)f(x^*) = (1-\beta)f(x) + (1-\beta)\sum_{i \in \mathcal{A}} \Delta(x_i^* | x_{1:i-1}^*, x) - (1-\beta)\sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}, x^*). \tag{6.24}$$

By Definition 6.5

$$(1-\beta)\sum_{i\in\mathcal{A}}\Delta(x_i^*|x_{1:i-1}^*,x)\leq\sum_{i\in\mathcal{A}}\Delta(x_i^*|x_{\mathcal{N}_i}). \tag{6.25}$$

and by $\eta$ optimality $\Delta(x_i^*|x_{\mathcal{N}_i})\leq\eta\Delta(x_i|x_{\mathcal{N}_i})$ such that

$$(1-\beta)f(x^*)\leq(1-\beta)f(x)+\eta\sum_{i\in\mathcal{A}}\Delta(x_i|x_{\mathcal{N}_i})-(1-\beta)\sum_{i\in\mathcal{A}}\Delta(x_i|x_{1:i-1},x^*). \tag{6.26}$$

Adding and subtracting $(1-\beta)\sum_{i\in\mathcal{A}}\Delta(x_i|x_{\mathcal{N}_i})$ to the right-hand side and combining summations and like terms gives

$$\begin{aligned}(1-\beta)f(x^*)\leq(1-\beta)f(x)&+(\eta-(1-\beta))\sum_{i\in\mathcal{A}}\Delta(x_i|x_{\mathcal{N}_i})\\&+(1-\beta)\sum_{i\in\mathcal{A}}\left[\Delta(x_i|x_{\mathcal{N}_i})-\Delta(x_i|x_{1:i-1},x^*)\right].\end{aligned} \tag{6.27}$$

By Definition 6.4

$$(1-\beta)f(x^*)\leq(1-\beta)f(x)+(\eta-(1-\beta))\sum_{i\in\mathcal{A}}\Delta(x_i|x_{\mathcal{N}_i})+(1-\beta)\alpha\sum_{i\in\mathcal{A}}\Delta(x_i|x_{\mathcal{N}_i}) \tag{6.28}$$

which simplifies to

$$(1-\beta)f(x^*)\leq(1-\beta)f(x)+(\alpha+\eta-1+\beta-\alpha\beta)\sum_{i\in\mathcal{A}}\Delta(x_i|x_{\mathcal{N}_i}). \tag{6.29}$$

The remainder of the proof is inspired by the proof of Theorem 1 in [31] to which we refer the reader for greater intuition regarding set of scalars $\{y_c\}_{c\in\mathcal{K}(G)}$.

Suppose that we have a set of scalars $\{y_c\}_{c\in\mathcal{K}(G)}$ where $\mathcal{K}(G)$ is the set of all cliques (fully

connected subsets of $\mathcal{A}$) such that $y_c \geq 0$ for all $c$ and $\sum_{c \in \mathcal{K}(G): i \in c} y_c \geq 1$ for all $i$. Then

$$\sum_{i \in \mathcal{A}} \Delta(x_i | x_{\mathcal{N}_i}) \leq \sum_{i \in \mathcal{A}} \Delta(x_i | x_{\mathcal{N}_i}) \left[ \sum_{c: i \in c} y_c \right] \tag{6.30}$$

$$= \sum_{i \in \mathcal{A}} \sum_{c: i \in c} y_c \Delta(x_i | x_{\mathcal{N}_i}) \tag{6.31}$$

$$= \sum_{c \in \mathcal{K}(G)} y_c \sum_{i \in c} \Delta(x_i | x_{\mathcal{N}_i}). \tag{6.32}$$

Therefore (6.29) and (6.32) give

$$(1-\beta) f(x^*) \leq (1-\beta) f(x) + (\alpha + \eta - 1 + \beta - \alpha\beta) \sum_{c \in \mathcal{K}(G)} y_c \sum_{i \in c} \Delta(x_i | x_{\mathcal{N}_i}). \tag{6.33}$$

Multiplying both sides by $1-\beta$ once more, we have

$$(1-\beta)^2 f(x^*) \leq (1-\beta)^2 f(x) + (\alpha + \eta - 1 + \beta - \alpha\beta) \sum_{c \in \mathcal{K}(G)} y_c (1-\beta) \sum_{i \in c} \Delta(x_i | x_{\mathcal{N}_i}) \tag{6.34}$$

where

$$\sum_{c \in \mathcal{K}(G)} y_c (1-\beta) \sum_{i \in c} \Delta(x_i | x_{\mathcal{N}_i}) \leq \sum_{c \in \mathcal{K}(G)} y_c \sum_{i \in c} \Delta(x_i | x_{\mathcal{N}_i \cap c}) \tag{6.35}$$

$$= \sum_{c \in \mathcal{K}(G)} y_c f(x_c) \tag{6.36}$$

$$\leq \sum_{c \in \mathcal{K}(G)} y_c f(x). \tag{6.37}$$

where (6.35) is by Definition 6.5, (6.36) is by Definition 6.2, and (6.37) is by the monotone

property of $f$. From (6.34) and (6.37) we have

$$(1-\beta)^2 f(x^*) \leq (1-\beta)^2 f(x) + (\alpha+\eta-1+\beta-\alpha\beta) \sum_{c\in\mathcal{K}(G)} y_c f(x) \tag{6.38}$$

The performance bound follows as

$$\frac{f(x)}{f(x^*)} \geq \frac{(1-\beta)^2}{(1-\beta)^2 + (\alpha+\eta-1+\beta-\alpha\beta)\sum_{c\in\mathcal{K}(G)} y_c} \tag{6.39}$$

To make the bound in (6.39) as tight as possible, one can solve the following optimization:

$$\min_{y} \sum_{c\in K(G)} y_c \tag{6.40}$$

subject to

$$\sum_{c\in K(G):i\in c} y_c \geq 1, \ \forall i \tag{6.41}$$

$$y_c \geq 0, \ \forall c. \tag{6.42}$$

The solution to this optimization problem is the fractional clique cover number of the communication graph $G$ and is denoted by $k^*(G)$. Thus our final result is given as

$$\frac{f(x)}{f(x^*)} \geq \frac{(1-\beta)^2}{(1-\beta)^2 + (\alpha+\eta-1+\beta-\alpha\beta)k^*(G)} \tag{6.43}$$

□

We note that when $k^* = 1$ the bound in (6.22) does not reduce to (6.13). For this reason, we believe a tighter bound is achievable in future work. Regardless, this result provides, to the

best of the authors knowledge, the first suboptimality guarantees for the greedy algorithm with limited information given a non-submodular objective function. Notably, when $\beta = 0$ and $\alpha = \eta = 1$, we recover the approximation bound of [31].

To lend intuition to the meaning of $k^*(G)$ in (6.22), Figure 6.1 shows the average clique cover number $k(G)$ for graphs with $N$ nodes and edges occurring randomly with probability given by the values on the horizontal axis. For each $N$ and edge probability $p$, 1000 random graphs were produced. The clique cover number satisfies $k^*(G) \leq k(G)$. Thus, the values shown in Figure 6.1 yield a conservative estimate of the true value of $k^*(G)$ on average. As intuition may suggest, edge probability (probability of successful communication) can have a significant effect on the quality of the bound of Theorem 6.7.

## 6.5 The Benefit of Search Objective Function

We consider a robotic search application where a team of robots seeks to cooperatively locate an unknown number of objects. We refer to the corresponding objective function as the *benefit of search*. The benefit of search gives the expected reduction in risk at a location as a result of obtaining $k$ measurements of the number of objects and of the environment type in that location.

We specifically consider the benefit of search as it addresses practical issues such as environmental influence on sensor performance, false alarms, and multiple visits to a single location where new measurements are not conditionally independent of previous measurements. These practical considerations come at the cost of the benefit of search being non-submodular such that classic approximation guarantees for submodular objective functions are not applicable. Thus, the benefit of search provides a meaningful example objective function to illustrate the value of Theorems 6.6 and 6.7.

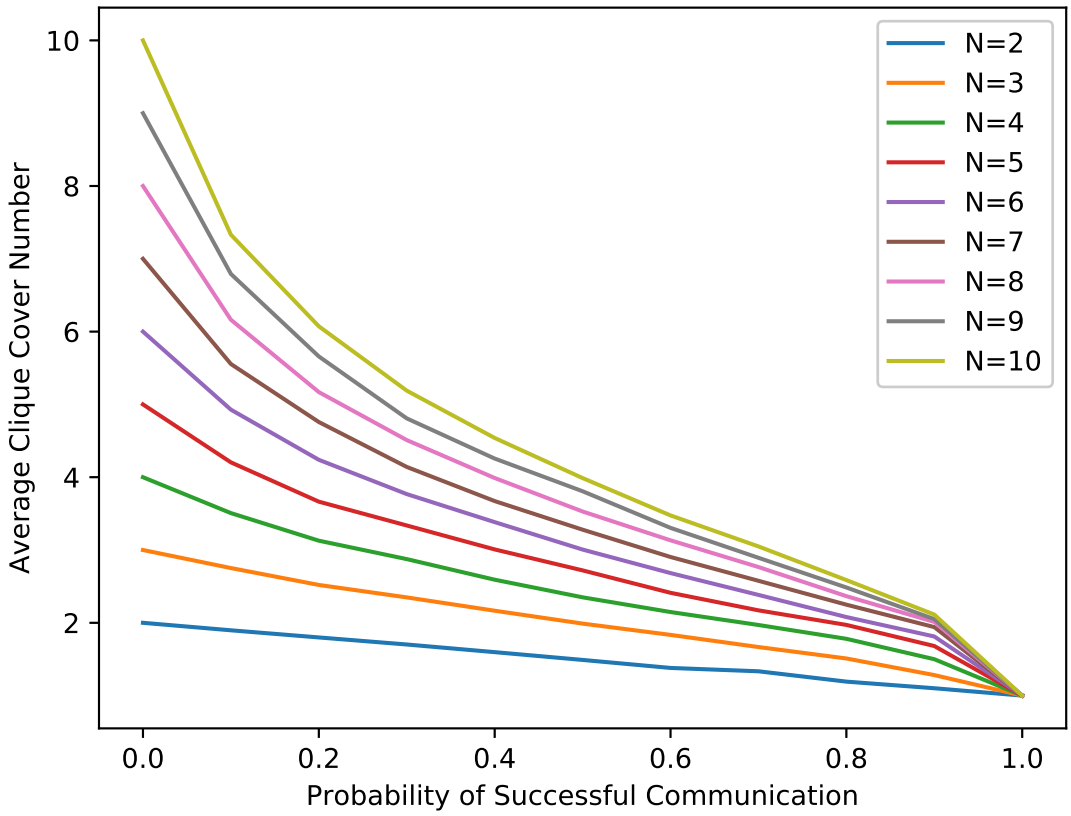



Figure 6.1: Average clique cover number of random graphs as a function of edge probability

Note that we provide a slightly different formulation of the benefit of search from that derived in [7, 53, 75]. Specifically, our derivation does not rely on a Bayesian optimal estimate of the environment type at a location facilitating the proof of Theorem 6.9 which gives that the benefit of search is monotone. Additionally, we show, in Theorem 6.8, that the benefit of search is normalized. Theorems 6.8 and 6.9 together ensure that the bounds of Theorems 6.6 and 6.7 apply to the benefit of search objective function.

We assume that a search area is composed of disjoint cells. Each cell has a distinct environment type and a number of targets of interest to a team of search agents. We consider, for now, a single cell of interest that we label $h_i$. Let $\mathcal{Z}_i$ be the set of target observations and

$\mathcal{Y}_i$ be the set of observations of the environment type within $h_i$. Our sensor model gives the probability of obtaining a measurement $z \in \mathcal{Z}_i$ given the true number of targets $t \in \mathcal{T}_i$ and environmental conditions $e \in \mathcal{E}_i$ and is given as

$$P(z|t,e) = \sum_{k=0}^{\min(t,z)} \binom{t}{k} D_e^k (1-D_e)^{t-k} (1-A_e) A_e^{z-k} \tag{6.44}$$

where $D_e$ gives the probability of detection and $A_e$ gives the probability of one or more false alarms. Note that both $D_e$ and $A_e$ depend on the true environment type $e$. We now drop the $i$ subscript denoting the cell $h_i$ as we will consider no other cells in the following derivations and definitions. We then have the Bayesian updates for the beliefs on the number of targets and the environmental conditions given by

$$P(t|z,e) = \frac{P(z|t,e)P(t|e)}{P(z|e)} \tag{6.45}$$

$$P(e|y) = \frac{P(y|e)P(e)}{P(y)} \tag{6.46}$$

We assume that the number of targets at a location is independent of the environment type. That is, $P(t|e) = P(t)$. The posterior belief on the number of targets conditioned on environmental measurements is thus given by

$$P(t|z,y) = \sum_{e \in \mathcal{E}} P(t|z,e)P(e|y). \tag{6.47}$$

Given a measurement $z$, the team of agents must estimate the true number of targets $t$ within the cell. We consider the case that overestimating the number of targets in the cell may be preferable to the alternative. As such, we impose a cost on the estimate $\delta(z) \in \mathcal{T}$

that handles this consideration.

$$L(t, \delta(z)) = c_i|t - \delta(z)| \quad \text{for } i \in \{1, 2\} \tag{6.48}$$

The posterior expected loss (risk) of computing the estimate $\delta(z)$ is then given by

$$\mathbb{E}[L(t, \delta(z))|z, y] = \sum_{t \in \mathcal{T}} P(t|z, y) L(t, \delta(z)). \tag{6.49}$$

and, given no measurements, the posterior expected loss given an estimate $\delta \in \mathcal{T}$ is given as

$$\mathbb{E}[L(t, \delta)] = \sum_{t \in \mathcal{T}} P(t) L(t, \delta) \tag{6.50}$$

with the Bayes estimate found as

$$\delta^* = \arg\min_{\delta \in \mathcal{T}} \mathbb{E}[L(t, \delta)]. \tag{6.51}$$

Note that (6.49) represents the primary deviation of our work from the benefit of search as described in [53]. Specifically, we compute risk using $P(t|z, y)$ instead of $P(t|z, e)$. As a result of this deviation, we are not required to find an optimal estimate of $e$ in order to compute anticipated risk.

The current risk is defined using the Bayes estimate and is given as

$$r(0) = \mathbb{E}[L(t, \delta^*)]. \tag{6.52}$$

The Bayes estimate of the number of targets given measurements $z$ and $y$ is given as

$$\delta^*(z) = \arg\min_{\delta(z)\in\mathcal{T}} \mathbb{E}[L(t, \delta(z))|z, y] \tag{6.53}$$

and the anticipated risk conditioned on measurements $z$ and $y$ is

$$r(z, y) = \mathbb{E}[L(t, \delta^*(z))|z, y]. \tag{6.54}$$

Naturally, we do not have measurements $z$ and $y$ when planning. Therefore, we find the expectation of (6.54) over the space of possible measurements to get the anticipated risk

$$\mathbb{E}[r(z, y)] = \sum_{z\in\mathcal{Z}} \sum_{y\in\mathcal{Y}} P(z, y)\mathbb{E}[L(t, \delta^*(z))|z, y]. \tag{6.55}$$

The joint probability $P(z, y)$ of obtaining measurements $z$ and $y$ is

$$P(z, y) = \sum_{t\in\mathcal{T}} \sum_{e\in\mathcal{E}} P(z|t, e)P(y|e)P(t)P(e). \tag{6.56}$$

Because we use new measurements to update our prior distributions $P(t)$ and $P(e)$, the conditional probability of a second set of measurements is

$$P(z_2, y_2|z_1, y_1) = \sum_{t\in\mathcal{T}} \sum_{e\in\mathcal{E}} P(z_2|t, e)P(y_2|e)P(t|z_1, y_1)P(e|y_1). \tag{6.57}$$

Note that $P(z_1, z_2, y_1, y_2) = P(z_2, y_2|z_1, y_1)P(z_1, y_1)$. Expanding upon this, we see that for $k$ measurements we have

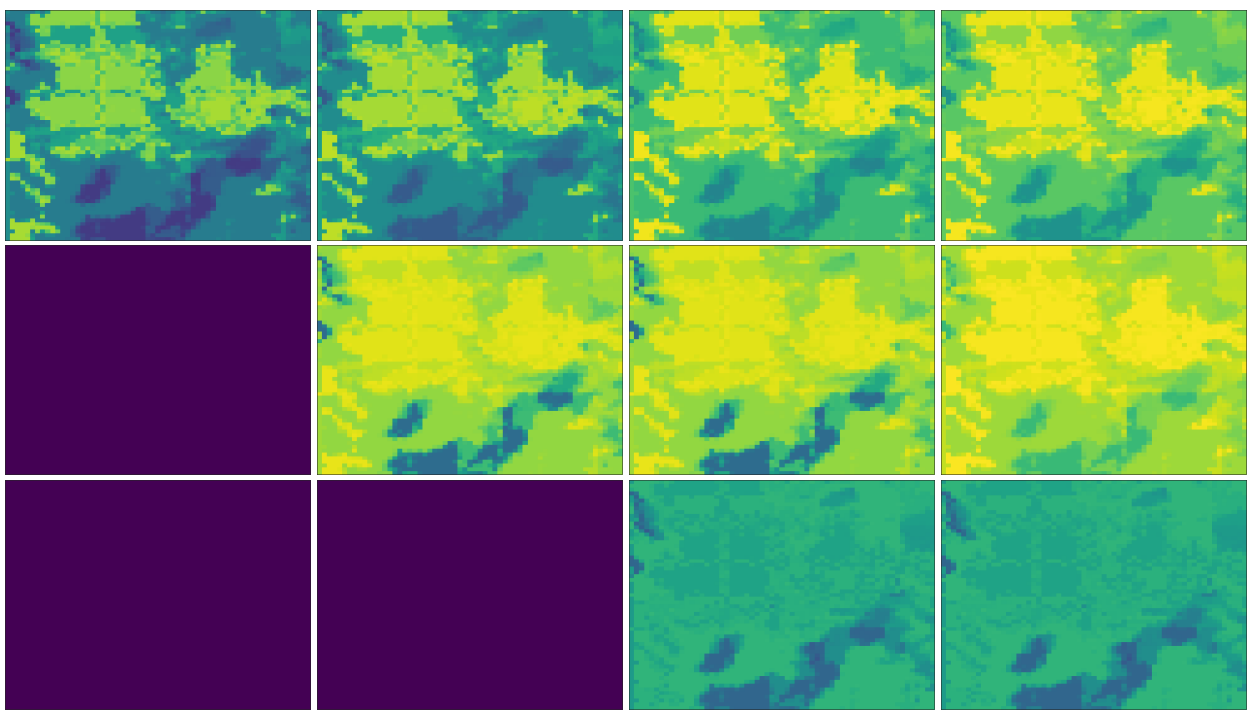
$$P(z, y) = \sum_{t\in\mathcal{T}} \sum_{e\in\mathcal{E}} P(z_k|t, e)P(y_k|e)P(t|z_{1:k-1}, y_{1:k-1})P(e|y_{1:k-1})P(z_{1:k-1}, y_{1:k-1}) \tag{6.58}$$

where $z = \{z_1, \ldots, z_k\}$ and $y = \{y_1, \ldots, y_k\}$ and $P(z, y) = P(z_1, \ldots, z_k, y_1, \ldots, y_k)$. The anticipated risk given $k$ measurements is then found as

$$r(k) = \sum_{z \in \mathcal{Z}^k} \sum_{y \in \mathcal{Y}^k} P(z, y) \mathbb{E}[L(t, \delta^*(z)) | z, y]. \tag{6.59}$$

We use equations (6.52) and (6.59) to define the benefit of searching cell $h_i$ $k$ times as

$$f_i(k) \triangleq r(0) - r(k). \tag{6.60}$$



$k = 1$ $k = 2$ $k = 3$ $k = 4$

Figure 6.2: The first row gives the normalized anticipated reward for $k$ visits to each cell. The second row gives the generalized curvature from Definition 6.4 for each cell assuming at most $k$ visits to each cell. The third row gives the inverse generalized curvature from Definition 6.5 for each cell assuming at most $k$ visits to each cell. All values are between 0 and 1 with darker regions indicating small values and lighter regions indicating large values.

## 6.6 Properties of the Benefit of Search

To show that the benefit of search is an appropriate objective function given the conditions of Theorems 6.6 and 6.7, we first show that the benefit of search is normalized and monotone. Furthermore, we characterize the parameters $\alpha$ and $\beta$ for each cell in a discrete map where the environment type prior probability distribution $P(e)$ comes from real-world data that was acquired by an autonomous underwater vehicle during a subsea survey of Boston Harbor.

**Theorem 6.8** (Normalized)**.** *The benefit of search as described in Section 6.5 is normalized.*

*Proof.* Clearly, if no new measurements are taken then the anticipated risk given no measurements is simply the current risk once again. Therefore,

$$f(0) = r(0) - r(0) = 0. \tag{6.61}$$

□

**Theorem 6.9** (Monotone)**.** *The benefit of search as described in Section 6.5 is monotone.*

*Proof.* To be monotone, it must be true that increasing the number of measurements increases the benefit of search. That is, we wish to show that $f(k) \geq f(k-1)$ for all $k \geq 1$. Naturally, because risk is a positive number, and the benefit of search is the difference in risk, we simply need to show that the risk is monotonically decreasing in order to prove that $f$ is monotone increasing.

Therefore, we seek to show that $r(k) \leq r(k-1)$. Let $z = \{z_1, \ldots, z_k\}$ and $y = \{y_1, \ldots, y_k\}$. Also, let $z^- = \{z_1, \ldots, z_{k-1}\}$ and $y^- = \{y_1, \ldots, y_{k-1}\}$. As before, we have

$$r(k) = \sum_{z \in \mathcal{Z}^k} \sum_{y \in \mathcal{Y}^k} P(z, y) \left[ \sum_{t \in \mathcal{T}} P(t|z, y) L(t, \delta^*(z)) \right] \tag{6.62}$$

where $\delta^*(z)$ minimizes $\sum_{t\in\mathcal{T}} P(t|z,y)L(t,\delta(z))$ such that $\delta^*(z^-)$ minimizing $\sum_{t\in\mathcal{T}} P(t|z^-,y^-)L(t,\delta(z^-))$ yields an upper bound

$$r(k) \leq \sum_{z\in\mathcal{Z}^k}\sum_{y\in\mathcal{Y}^k} P(z,y)\left[\sum_{t\in\mathcal{T}} P(t|z,y)L(t,\delta^*(z^-))\right] \tag{6.63}$$

$$= \sum_{z^-\in\mathcal{Z}^{k-1}}\sum_{y^-\in\mathcal{Y}^{k-1}}\sum_{z_k\in\mathcal{Z}}\sum_{y_k\in\mathcal{Y}} P(z_k,y_k|z^-,y^-)P(z^-,y^-)\left[\sum_{t\in\mathcal{T}} P(t|z,y)L(t,\delta^*(z^-))\right] \tag{6.64}$$

$$= \sum_{z^-\in\mathcal{Z}^{k-1}}\sum_{y^-\in\mathcal{Y}^{k-1}} P(z^-,y^-)\left[\sum_{t\in\mathcal{T}}\sum_{z_k\in\mathcal{Z}}\sum_{y_k\in\mathcal{Y}} P(z_k,y_k|z^-,y^-)P(t|z,y)L(t,\delta^*(z^-))\right] \tag{6.65}$$

$$= \sum_{z^-\in\mathcal{Z}^{k-1}}\sum_{y^-\in\mathcal{Y}^{k-1}} P(z^-,y^-)\left[\sum_{t\in\mathcal{T}}\sum_{z_k\in\mathcal{Z}}\sum_{y_k\in\mathcal{Y}} P(t,z_k,y_k|z^-,y^-)L(t,\delta^*(z^-))\right] \tag{6.66}$$

$$= \sum_{z^-\in\mathcal{Z}^{k-1}}\sum_{y^-\in\mathcal{Y}^{k-1}} P(z^-,y^-)\left[\sum_{t\in\mathcal{T}} P(t|z^-,y^-)L(t,\delta^*(z^-))\right] \tag{6.67}$$

$$= r(k-1) \tag{6.68}$$

□

In order to characterize the parameters $\alpha$ and $\beta$, we consider that there are 3 possible environment types. For simplicity, we assume that all search agents have the same sensors and therefore $P(z|t,e)$ and $P(y|e)$ is the same across all agents. Specifically, we use

$$P(y|e) = \begin{pmatrix} 0.82 & 0.09 & 0.09 \\ 0.08 & 0.84 & 0.08 \\ 0.06 & 0.06 & 0.88 \end{pmatrix} \tag{6.69}$$

where the $ij^{\text{th}}$ element of $P(y|e)$ gives the probability that the environmental sensor will

measure the environment type to be $y = j$ given the true environment type is $e = i$. and, in constructing $P(z|t, e)$, we use $D_e = (0.650.80.95)$ $A_e = (0.40.30.05)$

We assume $t \in \{0, 1, 2\}$ with prior probability distribution $P(t)$ chosen to be a truncated Poisson distribution meaning that most locations are considered unlikely to contain an object of interest. The cost of underestimating the number of targets is $c_1 = 3$ and the cost of overestimating the number of targets is $c_2 = 1$. The normalized anticipated reward as well as the generalized curvature $\alpha$ and inverse generalized curvature $\beta$ for each cell in the search area corresponding to a fixed maximum number of search passes $k$ are shown in Figure 6.2.

In the specific example shown in Figure 6.2, $\alpha$ increases significantly in the second column reflecting a relatively small increase in reward for visiting each cell a second time as compared to the reward gained upon a single search pass. The increased value of $\beta$ in the third column corresponds to an increased marginal reward upon a third visit to each cell as compared to the marginal reward attained upon a second visit to each cell. The maximum values of $\alpha$ and $\beta$ corresponding to each value of $k$ are given in Table 6.1.

Note that in Definitions 6.4 and 6.5, $\alpha$ and $\beta$ are characterized based on the ground set of a matroid. Because there is no limit on how many times a single cell may be visited, we suppose that the marginal gain for visiting a cell more than $k$ times is zero. This results in $\alpha = 1$ for each cell and $\beta$ is the maximum value over all cells for the maximum value of $k$ considered before fixing additional marginal gains to be zero. We can, therefore bound the value of a joint path planned using the greedy algorithm using $\alpha = 1$ and $\beta$ is the value given in Table 6.1 for the maximum value of $k$ considered. Thus, for $k = 4$ and $\eta = 1.25$, Theorem 6.6 guarantees an approximation ratio of $(1 - 0.6564)/(1.25 + (1 - 0.6564)) = 0.2156$. The approximation guarantee of Theorem 6.7 may be calculated likewise.

| | k = 1 | k = 2 | k = 3 | k = 4 |
|---|---|---|---|---|
| $\max_{h_i} \alpha$ | 0 | 0.9688 | 0.9688 | 0.9923 |
| $\max_{h_i} \beta$ | 0 | 0 | 0.6564 | 0.6564 |

Table 6.1: Maximum values of $\alpha$ and $\beta$ with respect to the maximum number of allowed search passes $k$.

## 6.7 Approximation Guarantees with respect to A Submodular Approximate Objective Function

This section is motivated by prior work wherein the objective function (the benefit of search) is changed so that, in planning, a reward is only nonzero upon a single visit to search a location [53]. The decision to alter the objective function eases the computational burden of computing the reward for multiple visits. However, in recognizing that the true benefit of searching a location multiple times is greater than that of searching a location a single time (See the top row of Figure 6.2), it may be beneficial to utilize the true benefit of searching the location multiple times (6.60) (at least for more than a single visit) to evaluate paths in planning. As discussed in Section 6.6, the benefit of search assuming multiple visits is a non-submodular objective function whereas artificially setting the marginal reward for visiting a location more than once to zero ensures that the approximate benefit of search is submodular. Thus, the approximate benefit of search provides theoretical guarantees for the greedy algorithm. Given that the true benefit of search yields higher rewards for multiple visits to cells, we address the question. If planning is accomplished using the true benefit of search, does the result achieved by the greedy algorithm have any theoretical lower bound guarantees with respect to the maximum achievable reward given the approximate benefit of search?

**Theorem 6.10.** *Let $f : 2^{\mathcal{V}} \mapsto \mathbb{R}_{\geq 0}$ be a normalized, monotone, potentially non-submodular function and let $\tilde{f} : 2^{\mathcal{V}} \mapsto \mathbb{R}_{\geq 0}$ be a submodular, non-decreasing, normalized function. Let $\tilde{\Delta}$*

*represent the marginal reward evaluated using the function* $\tilde{f}$*. If* $\Delta(v|\mathcal{S}) \geq \tilde{\Delta}(v|\mathcal{S})$ *for all* $v \in \mathcal{V}$ *and* $\mathcal{S} \subseteq \mathcal{V} \setminus \{v\}$ *then*

$$\frac{f(x)}{\tilde{f}(x^*)} \geq \frac{1}{2}. \tag{6.70}$$

*Proof.* We begin with equation (6.7) where $x^*$ is a solution maximizing $\tilde{f}$ and $x$ is a solution produced by the greedy algorithm using the objective function $f$.

$$\begin{aligned}
\tilde{f}(x^*) &\leq \tilde{f}(x) + \sum_{i \in \mathcal{A}} \tilde{\Delta}(x_i^* | x_{1:i-1}^*, x) && (6.71)\\
&\leq \tilde{f}(x) + \sum_{i \in \mathcal{A}} \tilde{\Delta}(x_i^* | x_{1:i-1}) && (6.72)\\
&\leq \tilde{f}(x) + \sum_{i \in \mathcal{A}} \Delta(x_i^* | x_{1:i-1}) && (6.73)\\
&\leq \tilde{f}(x) + \sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}) && (6.74)\\
&= \sum_{i \in \mathcal{A}} \tilde{\Delta}(x_i | x_{1:i-1}) + \sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}) && (6.75)\\
&\leq \sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}) + \sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}) && (6.76)\\
&= 2 \sum_{i \in \mathcal{A}} \Delta(x_i | x_{1:i-1}) && (6.77)\\
&= 2 f(x) && (6.78)
\end{aligned}$$

The first inequality is by monotonicity and removing the final term of equation (6.7), the second inequality is by submodularity of $\tilde{f}$, the third inequality is by hypothesis, the fourth inequality is by optimality of the greedy selection using $f$, and the fifth inequality is again by hypothesis. □

It may be shown using the same procedure that the solution produced by the greedy algorithm using $f$ is guaranteed to achieve at least 1/2 the reward achieved by the greedy

algorithm using $\tilde{f}$. This may be validated by a simple example. That is, the greedy algorithm using $f$ is not guaranteed to yield a larger result than the greedy algorithm using $\tilde{f}$ although the maximum achievable reward may be higher when using $f$. However, this result holds regardless of the submodularity of $f$. Thus, if $\frac{1}{2}\tilde{f}(x^*)$ is a satisfactory lower bound, then there is little risk in using the greedy algorithm with $f$ regardless of the submodularity of $f$.

## 6.8 Conclusions

We present worst case performance bounds for the greedy algorithm in seeking to maximize a normalized, monotone, but not necessarily submodular objective function under a general matroid constraint. We further provide bounds on the performance of the greedy algorithm assuming limited information at each planning step. We demonstrate that a variant of the benefit of search objective function is normalized and monotone, but not submodular. We characterize the curvature of the benefit of search using real-world data collected by an autonomous underwater vehicle. We show that theoretical approximation guarantees are achievable despite non-submodularity of the objective function. Additionally, we show that a solution produced by the greedy algorithm in seeking to maximize a normalized, monotone, non-submodular objective function is guaranteed an approximation bound of 1/2 with respect to the value of a solution maximizing an approximate submodular objective function.

# Chapter 7

# Experiments in Underwater Feature Tracking with Performance Guarantees Using a Small AUV

We present the results of experiments performed using a small autonomous underwater vehicle to determine the location of an isobath within a bounded area. The primary contribution of this work is to implement and integrate several recent developments real-time planning for environmental mapping, and to demonstrate their utility in a challenging practical example. We model the bathymetry within the operational area using a Gaussian process and propose a reward function that represents the task of mapping a desired isobath. As is common in applications where plans must be continually updated based on real-time sensor measurements, we adopt a receding horizon framework where the vehicle continually computes near-optimal paths. The sequence of paths does not, in general, inherit the optimality properties of each individual path. Our real-time planning implementation incorporates recent results that lead to performance guarantees for receding-horizon planning.

## 7.1 Introduction

Determining the location of a depth contour or isobath is an important element of quickly finding navigable paths in waterways. The conventional approach to underwater feature mapping relies on performing an exhaustive search over an area where a human operator believes features of interest are located. Path planning methods that seek to maximize information gain by planning sequences of locations that maximize a specific reward function are available to attempt to track underwater features more quickly than exhaustive search methods. However, many of these methods are susceptible to poor performance caused by myopic planning horizons when using receding horizon methods or local maxima in the reward function when using gradient-based optimization approaches. The primary contribution of this paper is the presentation of the results of experiments in depth contour estimation using path planning techniques with performance guarantees. We integrate several recent developments for real-time isobath localization, all of which can be easily modified for similar real-time mapping applications: (1) use of a sparse Gaussian process for representing the bathymetry, which leads to a continuous representation of the environment along with rigorous assessment of uncertainty, and (2) use of a so-called terminal reward in the optimization problem that generates each near-optimal path, which ensures that the cumulative reward attained by the AUV from a sequence of near-optimal paths is no worse than a desired lower bound. An important contribution of this work is to describe specific implementation solutions needed to integrate and implement these ideas for real-time operation on an AUV.

Isobath localization, like front tracking, can be formulated as an informative path planning (IPP) problem [52]. The goal of the IPP problem is to find a path that maximizes an information utility function while not exceeding a budget constraint on the cost of the path. In general, the computational complexity associated with informative path planning grows exponentially with the length of the path. Heuristic planning methods may be used to

Figure 7.1: 690 vehicles being loaded for transportation to the operational area.

produce reasonable results [21, 25, 78]. However, it is desirable to plan paths that are as nearly optimal as possible. Despite a host of techniques developed to provide approximate or suboptimal solutions [38, 62, 65, 76], receding horizon methods are often used regardless of the planning techniques used within the shorter planning horizon [1, 2, 70].

In Chapters 3 and 4, the authors showed how to append a specific terminal reward to the short horizon optimization problem used for each path segment such that the reward of the overall receding horizon path is guaranteed to exceed a desirable lower bound. This result was demonstrated for the explicit reward function associated with a robotic search problem in a discrete environment through numerical experiments. In this work, we demonstrate the utility of adding a terminal reward to a different class of reward function based on level set estimation using Gaussian process models. We report on the results of challenging real-world experiments accomplished using a real autonomous underwater vehicle (AUV) performing on-line planning using live data.

Throughout this work, we seek to rapidly map the 15 meter isobath within a bounded area of Claytor lake in Dublin, VA, USA. The specific region of Claytor lake where the experiments of this work are performed is shown in Figure 7.4 with the solid red lines indicating the boundaries of the operational area. The AUV used for these experiments is the Virginia Tech 690 vehicle shown in Figure 7.1.

## 7.2 Gaussian Process Regression

In this section we provide a brief review of Gaussian process regression. We refer the reader to [72, Chapter 2] for a more thorough discussion of the topic.

A Gaussian process is a collection of random variables, any finite number of which have a

joint Gaussian distribution [72, Definition 2.1]. A GP is completely specified by its mean and covariance functions $m(p)$ and $k(p, p')$ respectively and is written

$$f(p) \sim \mathcal{GP}(m(p), k(p, p')) \tag{7.1}$$

where $f(p)$ is a real process. In this work, we assume that the bathymetry or depth of Claytor lake within the operational area may be modeled using a GP.

A sample or datum $d = (p, z)$ is an input-measurement pair with $p \in \mathscr{P}$ and $z \in \mathbb{R}$. Let $\mathcal{D}_t$ be the set of all samples acquired up to time $t$. For any input $p_i$, an associated measurement $z_i$ with

$$z_i = f(p_i) + \epsilon_i, \text{ for } i = 1, \ldots, |\mathcal{D}_t| \tag{7.2}$$

where $|\cdot|$ denotes cardinality and $\epsilon_i \sim \mathcal{N}(0, \sigma_n^2)$ is drawn from a zero mean Gaussian distribution with standard deviation $\sigma_n$.

Given the data set $\mathcal{D}_t$, one may wish to predict $f(p^*)$ for an arbitrary input $p^*$. From [72, Chapter 2.7], the posterior predictive distribution of $f(p^*)$ conditioned on $\mathcal{D}_t$ is Gaussian and specified by the mean $\mu_{p^*|\mathcal{D}_t}$ and variance $\sigma^2_{p^*|\mathcal{D}_t}$ given as

$$\mu_{p^*|\mathcal{D}_t} = m(p^*) + k(p^*)^\top (K_t + \sigma_n^2 I)^{-1} (Z_t - m(P_t)) \tag{7.3}$$

$$\sigma^2_{p^*|\mathcal{D}_t} = k(p^*, p^*) - k(p^*)^\top (K_t + \sigma_n^2 I)^{-1} k(p^*). \tag{7.4}$$

$Z_t = [z_1, \ldots, z_{|\mathcal{D}_t|}]^\top$ is the vector of all measurements contained in $\mathcal{D}_t$ and $P_t = \{p_1, \ldots, p_{|\mathcal{D}_t|}\}$ is the set of associated inputs. The prior mean vector is $[m(P_t)]_i = m(p_i)$, $[k(p^*)]_i = k(p^*, p_i)$, $[K_t]_{i,j} = k(p_i, p_j)$, and $I_t$ is the $|\mathcal{D}_t| \times |\mathcal{D}_t|$ identity matrix.

When computing the predictive distribution at $p^*$, the matrix inversion $(K_t + \sigma_n^2 I)^{-1}$ can become intractable as $|\mathcal{D}_t|$ grows large. To address this complexity, we utilize a distance

metric $d(p, p')$ on $\mathscr{P}$ to reduce the number of samples retained in $\mathcal{D}_t$ and make predictions using local Gaussian processes. To reduce the number of samples retained in $\mathcal{D}_t$, a new sample $d_c = (p_c, z_c)$ is only added to $\mathcal{D}_t$ if $d(p_c, p_i) \geq \delta_f$ for all $p_i \in \mathcal{D}_t$. For a predictive point $p^*$, the associated data is limited to a locality around $p^*$ giving the local data set as $\mathcal{D}_{p^*,t} = \{d_i \in \mathcal{D}_t | d(p_i, p^*) \leq \delta_c\}$. The local predictive mean and variance are given by equations (7.3) and (7.4) using $\mathcal{D}_{p^*,t}$ in the place of $\mathcal{D}_t$.

Throughout this work, we define the input space $\mathscr{P}$ to be two dimensional euclidean space ($\mathbb{R}^2$) with distance metric $d(p, p') = \|p - p'\|$ being the euclidean distance. We use the squared exponential kernel function

$$k(p, p') = \sigma_f^2 \exp\left\{\frac{\|p - p'\|^2}{2l_c^2}\right\} \tag{7.5}$$

as the prior for the covariance function. The prior mean function used in this work is presented in Section 7.5.

## 7.3 Objective Function

We wish to determine the level set $\mathcal{L}(l^*) = \{p \in \mathscr{P} | f(p) = l^*\}$. To direct the selection of locations that should be measured in order to obtain information regarding this level set, we require an objective function that encodes uncertainty regarding whether $p^*$ is an element of $\mathcal{L}(l^*)$. To provide such a function, we use a measure of ambiguity presented in [30] which is closely related to the straddle heuristic presented in [11].

**Definition 7.1** (Ambiguity)**.** The ambiguity associated with a point $p^*$ with respect to a level $l$ is

$$a_{p^*|\mathcal{D}_t} = \beta\sigma_{p^*|\mathcal{D}_t} - |\mu_{p^*|\mathcal{D}_t} - l|. \tag{7.6}$$

Concretely, if $\beta = 2$ and $a_{p^*|\mathcal{D}_t} < 0$, then the model indicates that the probability that $f(p^*)$ falls on the opposite side of $l$ as $\mu_{p^*|\mathcal{D}_t}$ is less than 2.5 percent.

Similar to [30] where batches of points are selected for sampling, we seek to select a path or a sequence of points to sample. Considering the ambiguity of sets of points requires some slight modifications to account for locations that will be sampled while no samples have yet been acquired.

**Definition 7.2** (Anticipated Ambiguity)**.** Let $\mathcal{P}^* = \{p_0^*, \ldots, p_n^*\}$ and $\mathcal{P}_i^* = \{p_0^*, \ldots, p_{i-1}^*\}$ with $\mathcal{P}_0^* = \emptyset$. Then the anticipated ambiguity of $p_i^* \in \mathcal{P}^*$, with respect to a level $l$ is

$$a'_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^*} = \beta\sigma_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^*} - |\mu_{p^*|\mathcal{D}_t} - l| \tag{7.7}$$

where $\sigma_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^*}$ is the posterior predictive standard deviation computed using equation (7.4) where $\mathcal{P}_t$ is augmented with $\mathcal{P}_i^*$.

Having defined the anticipated ambiguity for a set of points, we define our reward function to be the anticipated reduction in ambiguity at a point with respect to a level $l$.

$$r_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^*} = \max(a'_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^*}, 0) - \max(a'_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^* \cup p_i^*}, 0). \tag{7.8}$$

Note that $a'_{p_i^*|\mathcal{D}_t,\mathcal{P}_0^*} = a_{p_i^*|\mathcal{D}_t}$. Also note that without the inclusion of the max function, the reward simply becomes $\beta[\sigma_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^*} - \sigma_{p_i^*|\mathcal{D}_t,\mathcal{P}_i^* \cup p_i^*}]$, which is uninformative regarding $\mathcal{L}(l)$. Intuitively, the reward places the most value on points $p^*$ where $f(p^*)$ is expected to be close to $l$ and the anticipated standard deviation at $p^*$ is expected to be reduced significantly. That is, $f(p^*)$ may be close to $l$, but if the anticipated reduction in variance is low, there is little to be gained by sampling that location.

## 7.4 Virginia Tech 690 AUV

Experiments were performed using the Virginia Tech 690 AUV shown in Figure 7.1. Specifications for the 690 are given in Table 7.1. Significantly, the 690 vehicle is equipped with a Doppler velocity log (DVL, TELEDYNE MARINE Pathfinder) with four acoustic beams at a 30 degree janus angle. An example AUV illustrating the DVL placement on the 690 is shown in Figure 7.2 with DVL beams shown in red. The DVL records a measurement of the distance from the DVL sensor to the lake bed along each beam at a rate of 10Hz. The 690 vehicle is able to estimate its location using DVL-aided inertial navigation. The navigation estimate is fused with the DVL range measurements to infer the depth of the lake bed. We use all four range measurements in our implementation. Throughout this chapter, vehicle state, and sensor measurements are converted into a North, East, Down (NED) coordinate frame. The origin of the NED coordinate system is shown as the green point in Figure 7.4.

| **PARAMETER** | **SPECIFICATION** |
|---|---|
| Length | 2.2 m |
| Displacement | 42.9 kg |
| Diameter | 17.5 cm |
| Endurance | 24 hrs @ 2 m/s |
| Depth | 500 meters |
| Communications | 900MHz RF modem, Wi-Fi, acoustic communication, satellite communication |
| Navigation | DVL + IMU, acoustic ranging |
| Mission Sensor | PingDSP 3D bathymetric sonar |

Table 7.1: Specifications of the Virginia Tech 690 AUV.

Control and navigation of the 690 vehicle are accomplished using ROS-based autonomy software running on an ODROID-XU4 single board computer. Path planning for these experiments is accomplished separately on an NVIDA Jetson TX2. Line segments connecting waypoints in the paths computed onboard the Jetson are delivered to the ROS-based autonomy software on the ODROID via a TCP interface. The autonomy software running

Figure 7.2: Simulated DVL beams shown in red.

on the ODROID drives the 690 vehicle to converge to the input line segment and reports when the end of the line segment has been reached. DVL sensor measurements and vehicle navigation data are transmitted from the ODROID to the Jetson over the same interface. Notably, executing path planning on a separate computer and sending line segments over a well-defined interface, ensures that the path planning algorithm does not interfere with critical autonomy functionality.

## 7.5 Field Trials

The prior mean $m(p)$ is defined using a line $\mathscr{L}$. Let $d(p, \mathscr{L})$ be the minimum euclidean distance from $p$ to $\mathscr{L}$. Then the prior mean at a point $p$ is given as

$$m(p) = \max(d_{max}(1 - [d(p, \mathscr{L})/c]^2), 0). \tag{7.9}$$

Intuitively, the line $\mathscr{L}$ gives the anticipated deepest set of points in the the lake with the anticpated depth decreasing proportionally to the square of the distance from the line until a depth of zero is reached. The coefficients $c$ and $d_{max}$ are chosen to be 300 and 25 meters respectively. The specific line chosen for these trials is shown as the cyan dashed line in Figure 7.4. Figure 7.3 shows the 15 meter isobath of the prior mean function within the bounded search area. This choice of prior mean function provides a naive initial estimate of the location of the 15 meter isobath which facilitates quickly finding the true isobath location, but is not meant to represent the bathymetry accurately beyond an initial naive estimate. The parameters of the squared exponential prior covariance function in equation (7.5) are chosen to be $\sigma_f = 5.0$, $l_c = 40.0$ meters, and the sensor measurement noise $\sigma_n = 2.5$ meters. The data retention threshold $\delta_f$ and local GP diameter $\delta_c$ presented in Section 7.2 are chosen to be $\delta_f = 6$ meters and $\delta_c = 40$ meters or $0.15 l_c$ and $l_c$ respectively.

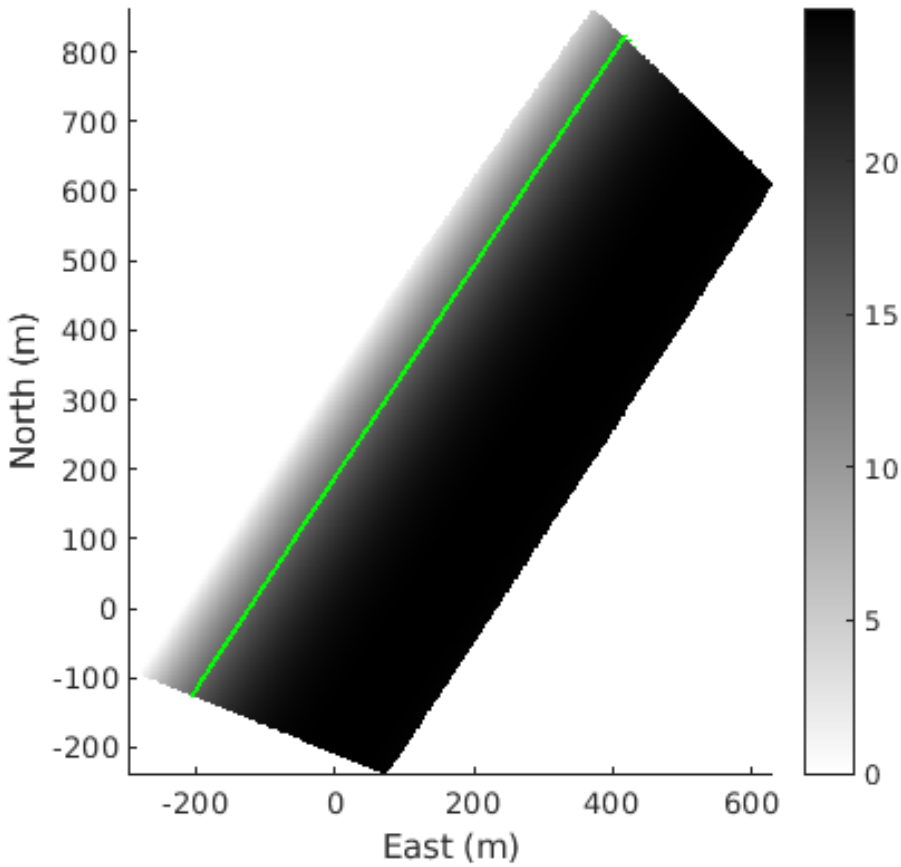



Figure 7.3: Prior mean depth map with 15 meter isobath shown as a green line.

### 7.5.1 Feasible Paths

Path planning for the vehicle is performed in the NED frame at a fixed depth of 2 meters. In planning, the vehicle state is given as $s = \{x_h, p_n, p_e\}$ where $x_h$ gives the heading in radians and $p_n$ and $p_e$ are the North and East components of the position of the state in meters from the NED origin. The set of next states available from any current state are defined by the

following discrete dynamics

$$x_h[t+1] = x_h[t] + a \tag{7.10}$$

$$p_n[t+1] = p_n[t] + d_m \cos(x_h[t] + a) \tag{7.11}$$

$$p_e[t+1] = p_e[t] + d_m \sin(x_h[t] + a) \tag{7.12}$$

where $a \in \mathcal{A}$ is an input angle in radians and $d_m$ is the desired euclidean distance in meters between path states. Throughout these experiments, $d_m = 30$ meters and $\mathcal{A} = \{-90, -45, -15, 0.0, 15, 45, 90\}$ with the values of $A$ given in degrees for convenience. Let $p[t] = [p_n[t], p_e[t]]^\top$. To traverse a path, the 690 vehicle tracks a line from $p[t]$ to $p[t+1]$ until it reaches $p[t+1]$ at which point it attempts to track the line from $p[t+1]$ to $p[t+2]$. By construction, the heading of the vehicle upon reaching a point in its path should be close to the desired heading $x_h$.

### 7.5.2 Terminal Rewards

The value of a path using equation (3.5) is found by appending a pair of lawnmower paths to the end of each short $n$-length path and selecting the lawnmower path that maximizes the reward of the augmented path. Lawnmower paths are given as a sequence of straight lines parallel to the heading of the final state of the short $n$-length path and continuing outward on a single side at fixed intervals in the direction of the first turn required to remain within the operational area. Notably, the value of the lawnmower path with the highest expected reward provides a lower bound on the performance of the receding horizon path. One may reasonably expect that constantly maximizing the expected reward would generate a better solution than the naive lawnmower path, however, the formal guarantee only provides that the receding horizon path will perform no worse than the naive lawnmower path. In practice,

the receding horizon path generally produces a significantly improved solution.

### 7.5.3 Computation of Reward Attained

Note that equation (7.8) assumes that a single measurement will be recorded precisely at the location contained in each path state when, in reality, measurements are taken periodically by the DVL at a rate of 10Hz with a subset of those measurements being retained according to the methods described in Section 7.2. In an attempt to ensure that several measurements of the lake depth are taken near each point of interest, we consider that a location has been sampled 3 seconds after the vehicle reaches the associated waypoint. The realized reduction in ambiguity is computed as the difference between the ambiguity at the point when first beginning the transition to the point and the ambiguity at the point 3 seconds after reaching the point. The choice of 3 seconds is based on the 690 vehicles average velocity of approximately 1.6 m/s.

### 7.5.4 Receding Horizon Path Planning and Execution

In this work, we utilize a basic implementation of Monte Carlo Tree Search (MCTS) [10, 39, 40] with minor modifications to ensure that the lower bound requirements of Proposition 4.2 are satisfied. First, the reward term of the UCT is normalized using the value of the highest-reward rollout path found for the current planning epoch and is represented as $\tilde{\mathcal{V}}_l(s_k)$. Second, instead of back-propagating the average reward of a Monte Carlo simulation, we back-propagate the maximum $\mathcal{V}_l(s_k)$ such that

$$\mathbb{UCT} = \frac{\mathcal{V}_l(s_k)}{\tilde{\mathcal{V}}_l(s_k)} + C\sqrt{\frac{\ln(N)}{n}}. \tag{7.13}$$

Lastly, at the beginning of each planning epoch, the recomputed value of the remainder of the best path found in the previous planning epoch is compared against the value of the highest-reward lawnmower path starting from the root state. The path with the highest reward is saved as the initial highest-reward rollout path for the planning epoch such that the MCTS planner is guaranteed to satisfy the critera of Proposition 4.2 at each step. Furthermore, the comparison of the remainder of the path at each planning step against the best lawnmower path from the same initial state ensures that the guarantee of Proposition 4.2 is extended to

$$J(\tilde{\gamma}_l(s_k)) \geq B_l(s_k) \tag{7.14}$$

at each step $k$ such that the theoretical lower bound for the receding horizon path produced is

$$J(\tilde{\gamma}_l(s_0)) \geq \underset{k \in \{1,\ldots,l\}}{\arg\max}[J(\tilde{\gamma}_k(s_0)) + B_l(s_k)] \tag{7.15}$$

as discussed in Section 5.4.

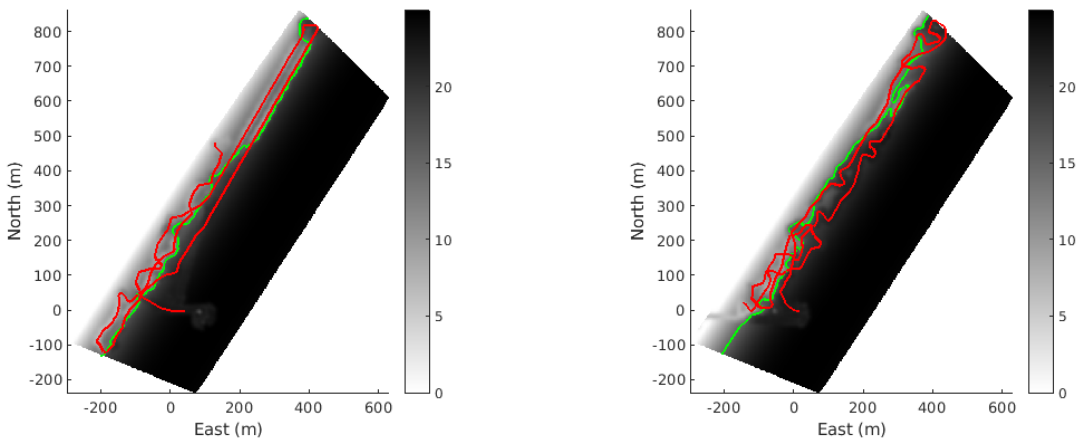
Defining a step to be a line segment connecting consecutive states in a path. The planning horizon is 5 steps, the execution horizon is a single step, and the total mission length $l$ is 100 steps. For comparison, one path was planned using the proposed receding horizon method with terminal rewards and another receding horizon was planned without the addition of terminal rewards. The receding horizon path produced using the proposed method is shown as the solid red line in Figure 7.5a. The receding horizon path produced without the use of terminal rewards is similarly shown in Figure 7.5b.

## 7.6 Results and Conclusions

The results of planning and executing receding horizon paths are shown in Figures 7.5 and 7.6. Figure 7.5 shows the posterior depth maps produced using equations (7.3) and (7.4) and the data sets acquired while traversing the receding horizon paths according to the methods discussed in Section 7.2. Figure 7.5 also shows the 15 meter isobath in green. The cumulative reward for the receding horizon path produced without the use of terminal rewards is shown as the dashed blue line in Figure 7.6 with the cumulative reward for the receding horizon path produced with terminal rewards shown as the solid green line. The solid black line shows the value of the lower bound given in equation 7.15 which is the maximum value of the solid red line. The red line gives the sum of the current cumulative reward attained by the receding horizon path up to step $k$ plus the lower bound on the value-to-go.

Notably, Figure 7.6 shows that both receding horizon paths performed significantly better than the value of the initial lower bound $B_l(s_0)$ with the receding horizon path produced using the proposed methods performing slightly better than the receding horizon path that did not include terminal rewards. While there is no guarantee that paths produced using the proposed methods will perform better or worse than receding horizon paths produced using other methods, these results do not indicate any degradation in performance for including terminal rewards defined by naive heuristic paths. Furthermore, the results of these experiments appear to further confirm the results of Proposition 4.2.

Figure 7.4: Claytor lake operational area.




(a) (b)

Figure 7.5: The posterior predictive depth map within the bounded search region given the measurements acquired while executing the receding horizon path shown in red produced using terminal rewards (a) and the receding horizon path produced without the addition of terminal rewards (b). The 15 meter isobath is shown in green.

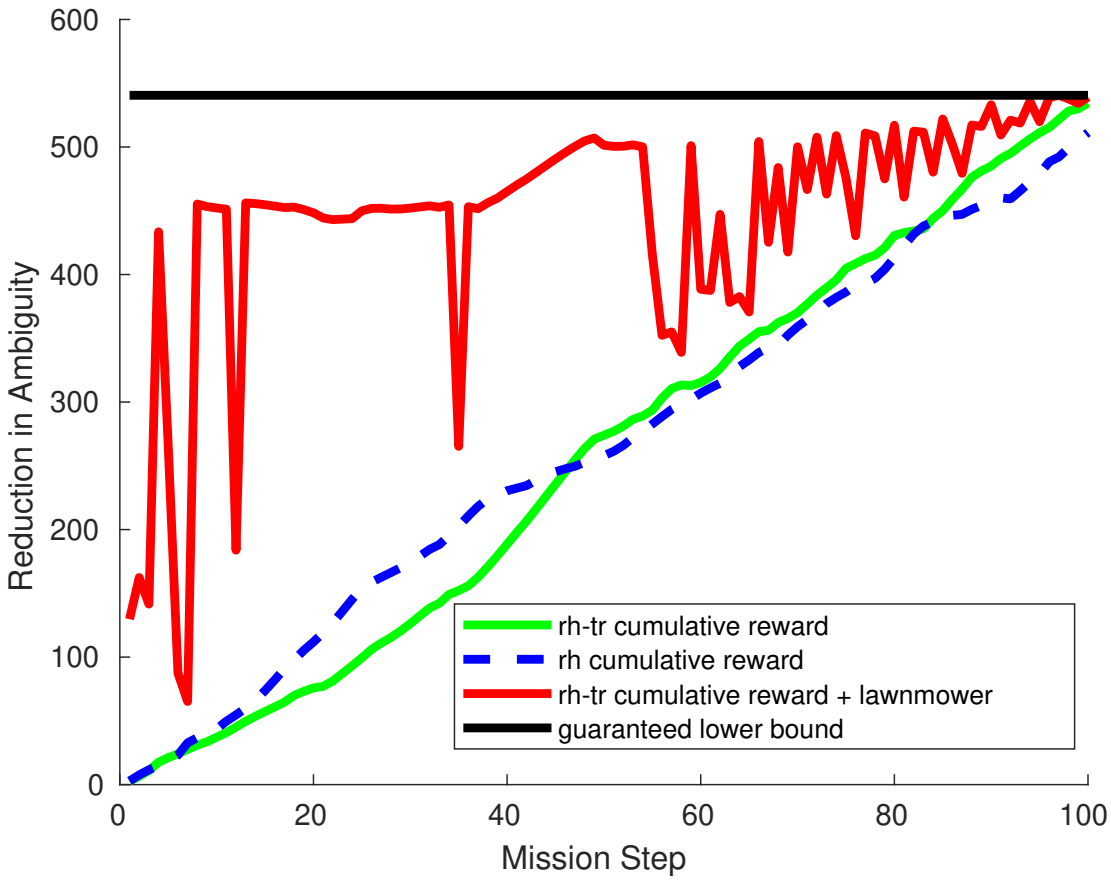



Figure 7.6: Cumulative rewards for the RH path and RH path constructed with terminal rewards are shown in dashed blue and solid green respectively. The cumulative reward for the RH path constructed with terminal rewards plus the lower bound on the value-to-go is shown as the solid red line. The guaranteed lower bound is shown as the solid black line.

# Chapter 8

# Experiments in Collaborative Underwater Feature Tracking with Teams of Small AUVs

We present the results of experiments performed using a team of small autonomous underwater vehicle to determine the location of an isobath within a bounded area. The primary contributions of this chapter are the development of a novel objective function for level set estimation (LSE) that utilizes the rigorous assesment of uncertainty provided by Gaussian processes as well as the presentation of the results of experiments in isobath localization with teams of real AUVs using path planning techniques designed to leverage theoretical performance guarantees. It is shown through experimental results addressing a challenging practical problem that desirable performance guarantees may be obtained by a team of agents performing a real-time search mission despite slow and intermittent communications and limited computational resources.

## 8.1 Introduction

The mapping of spatial phenomenon using mobile sensor platforms is an important topic in robotics, spanning a wide variety of applications from monitoring solar radiation [60]

to mapping structures associated with ocean bottom methane sources [13]. In the field of underwater robotics, autonomous underwater vehicles (AUV) can be employed for a variety of tasks such as mine hunting, collection of oceanographic data, and mapping of the ocean floor. The conventional approach to underwater feature mapping relies on performing an exhaustive search over an area where a human operator believes features of interest are located. This is normally accomplished with a single, expensive AUV system carrying a suite of sensors. In this work, we seek to address the problem of deploying a team consisting of multiple AUVs that are able to collaboratively search an area of interest in a decentralized manner, allowing the area to be searched much more quickly and efficiently.

We address the level set estimation problem where a team of agents collaboratively build a map of the environment and determine the regions where a critical threshold is passed. We specifically consider the problem of isobath localization or, in other words, the localization of a depth contour within a body of water. While our theoretical contributions could be applied to other problems, isobath localization provides a simple way for us to demonstrate our work with a team of AUVs deployed in the field. A motivating application of isobath localization is in determining regions within a waterway that are safely navigable. The primary contributions of this work are i) the development of a novel objective function for level set estimation (LSE) that utilizes a rigorous assesment of uncertainty provided by Gaussian processes, and ii) the experimentation in isobath localization with a team of AUVs using path planning techniques designed to leverage theoretical performance guarantees.

Level set estimation is the process of using noisy observations of the function $f$ defined on the input space $\mathscr{P}$ to determine the region(s) where $f$ exceeds some threshold [71]. Level set estimation has applications in geospatial data [66], bionformatics [74], and environmental studies [30, 33, 68]. This work is distinct from the existing literature on level set estimation in the following ways: (1) the level set is estimated using the decision-theoretic *benefit of search*

objective function presented in this work instead of a more common measure of classification ambiguity [30, 33], and (2) to the best of the authors' knowledge, this work represents the first use of the path planning methods proposed in Chapter 5 in real-world experiments for a team of agents. The path planning method used in Chapter 7 is related, but is only applied to the single agent level set estimation problem.

For path planning, we formulate the isobath localization problem as an informative path planning (IPP) problem. Despite a host of techniques developed to provide approximate or suboptimal solutions to the informative path planning problem [38, 62, 65, 76], receding horizon methods are often used regardless of the planning techniques used within the shorter planning horizon [1, 2, 70]. However, the sequence of path segments in a receding horizon implementation do not inherit the near-optimality of each individual segment. The theoretical performance guarantees that we seek to leverage in this work rely on the use of a so-called terminal reward in the path planning optimization problem at each step in the construction of a receding horizon path. It is shown in Chapter 5 that the appropriate use of terminal rewards ensures that the reward attained by traversing sections of nearly-optimal paths is no worse than a desired lower bound. It is further shown that any naive solution to the mission-length path planning problem may be used to determine terminal rewards and set the value of the worst-case performance bound. In this work, a naive lawn mower heuristic is used in conjunction with the greedy algorithm (see Chapter 6) to leverage the results of Chapter 5. The lawn mower heuristic is specifically chosen because it is computationally efficient, commonly used in subsea search applications, and generally produces reasonable results [25].

The remainder of this chapter is organized as follows. The benefit of search objective function for level set estimation is developed in Section 8.2. Path evaluation and assumptions made to leverage lower bound guarantees are presented in Section 8.3. A discussion of the practical

implementation of Gaussian process regression used in field trials is presented in Section 8.4. Experimental procedure, special considerations, and results are presented in Section 8.5. Throughout this work we use Gaussian process definitions and notation presented in Chapter 7.

## 8.2 Search Objective for Level Set Estimation

We assume that incorrectly classifying a location as being above or below a threshold of interest incurs a loss. For example, falsely labeling a location as deep enough for ship navigation could lead to a collision while falsely labeling a location as too shallow for ship navigation could lead a vessel to travel further than necessary. Mathematically, we represent these costs incurred by poor estimation using a loss function $\mathcal{L}$.

We model $f$ using a Guassian process as in equation (7.1). In the context of isobath localization, $f(p)$ gives the true depth of the waterway at the input location $p$. For convenience, we use the notation

$$f_p \triangleq f(p). \tag{8.1}$$

Let $l \in \mathbb{R}$ be the level of interest. The indicator function

$$g(f_p, l) = \begin{cases} 0 \text{ if } f_p < l \\ 1 \text{ if } f_p \geq l \end{cases} \tag{8.2}$$

classifies the input location $p$ as being safe ($f_p \geq l$) or unsafe ($f_p < l$) for navigation. Let $\delta \in \{0, 1\}$ be an estimate of $g(f_p, l)$. We define our loss function for level set estimation as

$$\mathscr{L}(f_p, \delta) = c_i |g(f_p, l) - \delta| \tag{8.3}$$

where $c_i > 0$ and

$$c_i = \begin{cases} c_1 \text{ if } g(f_p, l) < \delta \\ c_2 \text{ if } g(f_p, l) \geq \delta. \end{cases} \tag{8.4}$$

Intuitively, $c_1$ gives the cost associated with estimating a location to be safe when it is actually unsafe while $c_2$ gives the cost of estimating a location to be unsafe when it is actually safe.

### 8.2.1 Bayes' Risk and the Benefit Of Search

Recall that the true value of $f_p$ is not known. Therefore, the estimate $\delta$ which minimizes (8.3) cannot be computed directly. However, because we assume $f$ is modeled by a Gaussian process, a probability distribution for $f_p$ is known. Let $\mathcal{D}_t$ be a data set and $\delta$ be an estimate of (8.2), the expected loss or *risk* at an input point $p \in \mathscr{P}$ is

$$\mathbb{E}[\mathscr{L}(f_p, \delta)|\mathcal{D}_t] = \int_{\mathbb{R}} \pi(f_p|\mathcal{D}_t)\mathscr{L}(f_p, \delta)\, df_p \tag{8.5}$$

$$= \int_{\mathbb{R}} \pi(f_p|\mathcal{D}_t)c_i|g(f_p, l) - \delta|df_p \tag{8.6}$$

$$= \int_{-\infty}^{l} \pi(f_p|\mathcal{D}_t)c_1|0 - \delta|df_p + \int_{l}^{\infty} \pi(f_p|\mathcal{D}_t)c_2|1 - \delta|\, df_p \tag{8.7}$$

$$= \begin{cases} c_2 \int_l^{\infty} \pi(f_p|\mathcal{D}_t)df_p \text{ if } \delta = 0 \\ c_1 \int_{-\infty}^{l} \pi(f_p|\mathcal{D}_t)df_p \text{ if } \delta = 1. \end{cases} \tag{8.8}$$

where $\pi(f_p|\mathcal{D}_t)$ is the Gaussian probability distribution of $f_p$ conditioned on the data set $\mathcal{D}_t$ according to equations (7.3) and (7.4). Equations (8.6) and (8.7) simply expand the notation and partition the integral into the regions corresponding to the different regions where $f_p < l$ and $f_p \geq l$.

The Bayes' estimate is the estimate that minimizes the *risk* in (8.8) and is denoted

$$\delta^*(\mathcal{D}_t) = \underset{\delta \in \{0,1\}}{\arg\min} \mathbb{E}[\mathscr{L}(f_p, \delta)|\mathcal{D}_t]. \tag{8.9}$$

The *Bayes' risk* is the expected loss given the Bayes' estimate

$$\mathbb{E}[\mathscr{L}(f_p, \delta^*(\mathcal{D}_t))|\mathcal{D}_t] = \min\left(c_1 \int_{-\infty}^{l} \pi(f_p|\mathcal{D}_t) df_p, c_2 \int_{l}^{\infty} \pi(f_p|\mathcal{D}_t) df_p\right). \tag{8.10}$$

To represent the Bayes' risk, we define

$$r(\mathcal{D}_t) = \mathbb{E}[\mathscr{L}(f_p, \delta^*(\mathcal{D}_t))|\mathcal{D}_t]. \tag{8.11}$$

**Definition 8.1** (Benefit of Search)**.** Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be data sets. The *benefit of search* is the reduction in Bayes' risk due to obtaining additional data and is written

$$\mathscr{B}(\mathcal{D}_2|\mathcal{D}_1) = r(\mathcal{D}_1) - r(\mathcal{D}_1 \cup \mathcal{D}_2). \tag{8.12}$$

Consider that a data set $\mathcal{D}_t$ is available at time $t$. Let $\mathcal{D}_{t+1:t+n}$ be the data set acquired in the $n$ time increments following time $t$. And let $\mathcal{D}_{t+n} = \mathcal{D}_t \cup \mathcal{D}_{t+1:t+n}$ be the combined data set. In planning at time $t$, the input locations ($\mathcal{P}_{t+1:t+n}$) corresponding to the future data set $\mathcal{D}_{t+1:t+n}$ are known, but the measurements ($Z_{t+1:t+n}$) are not yet available. In planning, the benefit of search is computed using the expectation of the Bayes' risk

$$\mathbb{E}_{Z_{t+1:t+n}}[r(\mathcal{D}_{t+n})] = \int_{\mathbb{R}^{n_\mathcal{V}}} \int_{\mathbb{R}} \pi(f_p|\mathcal{D}_{t+n}) \mathscr{L}(f_p, \delta^*(\mathcal{D}_{t+n})) df_p dZ_{t+1:t+n} \tag{8.13}$$

where $n_\mathcal{V}$ denotes the cardinality of the measurement set $\mathcal{D}_{t+1:t+n}$ and, therefore, the dimensionality of the vector $Z_{t+1:t+n}$.

**Theorem 8.2.** *The benefit of search in* (8.12) *is normalized and monotone in expectation.*

A proof of Theorem 8.2 is given in Appendix 8.6.1. Significantly, the normalized and monotone properties indicate that the benefit of search is an appropriate objective function for use with the greedy algorithm described in Chapter 6 which is used for cooperative path planning at each step in the construction of the receding horizon paths presented in Section 8.5. In all of the experiments presented in Section 8.5, the benefit of search objective function is used for collaborative path planning.

### 8.2.2 Computing Expected Bayes' Risk

While the expected Bayes' risk in (8.13) may be approximated using Monte Carlo methods, a closed form approximation is desirable. However, no closed-form solution to (8.13) is known for this problem. Therefore, we propose an alternative approach to determine the expected Bayes' risk. We address the computation of the expected Bayes' risk generally assuming a Gaussian process model. The core insight of our proposed approach is that measured values influence the posterior predictive mean in (7.3) but have no effect on the posterior predictive variance in (7.4). Therefore, we consider a data set $\mathcal{Q}$ with $\mathcal{Q} = \mathcal{S} \cup \mathcal{V}$ for which the input locations $\mathcal{P}_\mathcal{Q}$ are known, but the measurement vector $Z_\mathcal{V}$ associated with the partial data set $\mathcal{V}$ is not yet available.

In order to determine the expected Bayes' risk, we first determine the marginal probability of $\mu$ ($\pi(\mu_{p|\mathcal{Q}}|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})$). Given the data set $\mathcal{Q}$, the posterior predictive distribution of $f_p$ is Gaussian with mean $\mu_{p|\mathcal{Q}}$ and variance $\sigma^2_{p|\mathcal{Q}}$. Following the process outlined in Appendix 8.7.1, the marginal probability of $\mu_{p|\mathcal{Q}}$ ($\pi(\mu_{p|\mathcal{Q}}|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})$) is found to be a Gaussian distribution

with parameters

$$\mu_\mu = \mu_{p|\mathcal{S}} \tag{8.14}$$

$$\sigma_\mu^2 = \sigma_{p|S}^2 - \sigma_{p|Q}^2. \tag{8.15}$$

Thus, the distribution for $\mu_{p|\mathcal{Q}}$ is centered around the posterior predictive mean given the set of available measurements with variance being the reduction in variance due to obtaining the data set $\mathcal{V}$ for which the measurement vector $Z_\mathcal{V}$ is not available. Notably, in the limit as $\sigma_{p|S}^2 - \sigma_{p|Q}^2$ approaches zero, the distribution converges to a Dirac delta function centered at $\mu_{p|\mathcal{S}}$. In other words, the posterior predictive mean given the set of future measurements is essentially deterministic at locations where the anticipated reduction in variance is negligible.

The expectation of the Bayes' risk with respect to the marginal distribution of the posterior predictive mean may now be found as

$$\mathbb{E}_\mu[r(\mathcal{Q})] = \int_\mathbb{R} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \int_\mathbb{R} \pi(f_p|\mu, \sigma_{p|\mathcal{Q}}^2)\mathscr{L}(f_p, \delta^*(\mu)) df_p d\mu \tag{8.16}$$

$$= \int_\mathbb{R} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \min\left(c_1\pi(f_p < l|\mu, \sigma_{p|\mathcal{Q}}^2), c_2\pi(f_p \geq l|\mu, \sigma_{p|\mathcal{Q}}^2)\right) d\mu \tag{8.17}$$

where $\delta^*(\mu)$ denotes the Bayes' estimate satisfying

$$\delta^*(\mu) = \arg\min_\delta \int_\mathbb{R} \pi(f_p|\mu, \sigma_{p|\mathcal{Q}}^2)\mathscr{L}(f_p, \delta)\, df_p. \tag{8.18}$$

To handle the min function in (8.17), we define the space

$$\Theta_\mu = \{\mu \in \mathbb{R} | c_1\pi(f_p < l|\mu, \sigma_{p|\mathcal{Q}}^2) \leq c_2\pi(f_p \geq l|\mu, \sigma_{p|\mathcal{Q}}^2)\} \tag{8.19}$$

or equivalently

$$\Theta_\mu = \left\{ \mu \in \mathbb{R} | \pi(f_p < l | \mu, \sigma^2_{p|\mathcal{Q}}) \leq \frac{c_2}{c_1 + c_2} \right\}. \tag{8.20}$$

Because $\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})$ is simply the CDF of a Gaussian distribution, we have

$$\Theta_\mu = (-\infty, \mu^*] \tag{8.21}$$

where $\mu^*$ satisfies the equation

$$\pi(f_p < l|\mu^*, \sigma^2_{p|\mathcal{Q}}) = \frac{c_2}{c_1 + c_2}. \tag{8.22}$$

As is shown in Appendix 8.6.3, solving for $\mu^*$ gives

$$\mu^* = l - \text{erf}^{-1}\left(\frac{c_2 - c_1}{c_1 + c_2}\right)\sigma_{p|\mathcal{Q}}\sqrt{2}. \tag{8.23}$$

The expected value of the Bayes' risk conditioned on the posterior predictive mean may now be computed as

$$\mathbb{E}_\mu[r(\mathcal{Q})] = c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})\pi(f_p \geq l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu + c_1 \int_{\mu^*}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu \tag{8.24}$$

which evaluates to the following as shown in Appendix 8.6.4

$$\begin{aligned}\mathbb{E}_\mu[r(\mathcal{Q})] = &\frac{(c_2 - c_1)}{4}\left[1 + \text{erf}\left(\frac{\mu^* - \mu_\mu}{\sigma_\mu\sqrt{2}}\right)\right] + \frac{c_1}{2} + \frac{c_1}{2}\text{erf}\left(\frac{l - \mu_\mu}{\sqrt{2\sigma^2_{p|\mathcal{Q}} + 2\sigma^2_\mu}}\right) \\ &- \frac{(c_1 + c_2)}{2}\int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})\text{erf}\left(\frac{l - \mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)d\mu.\end{aligned} \tag{8.25}$$

To the best of the authors' knowledge, the solution to the integral in the final term of (8.25) is unknown and represents a topic of current research. We use the recently proposed *exponential-quadratic approximation* [59] to provide an approximate solution given in (8.114) and (8.115) in Appendix 8.6.5.

## 8.3 Path Planning

Given a team of $K$ agents, an $n$-length joint path $\gamma_n(s_i) = \{s_i, \ldots, s_{i+n}\}$ beginning at joint state $s_i$ is a feasible sequence of joint states. A joint state is a collection of individual agent states and is denoted $s_i = \{s_i^1, \ldots, s_i^K\}$. A sequence is feasible if there exists a joint action $a$ such that the team of agents will transition from $s_i$ to $s_{i+1}$ by executing the action $a$.

We assume that a state $s_i^k$ for agent $k$ corresponds to an input location $p_i^k \in \mathscr{P}$ and a joint state $s_i$ corresponds to a set of input locations $p_i = \{p_i^1, \ldots, p_i^K\}$. Let $\mathcal{P}_n(s_i)$ be the set of input locations corresponding to the $n$-length path $\gamma_n(s_i)$. We assume that, in traversing $\gamma_n(s_i)$, the team of agents obtains a vector of measurements $Z_n(s_i)$ corresponding to the input locations $\mathcal{P}_n(s_i)$ resulting in a data set $\mathcal{D}_n(s_i) = (\mathcal{P}_n(s_i), Z_n(s_i))$.

Recall that the benefit of search is defined as a point-wise operation. Therefore, the reward associated with a path $\gamma_n(s_i)$ is a function of the locations at which the benefit of search is evaluated as well as the data set $\mathcal{D}_n(s_i)$. Additionally, the reward associated with a path (the reward for obtaining a data set) depends on previously obtained data $\mathcal{D}_{i-1}(s_0)$. Notice that the notation $\mathcal{D}_{i-1}(s_0)$ implies the existence of a path $\gamma_{i-1}(s_0)$ preceding $\gamma_n(s_i)$ such that $\gamma_{i+n}(s_0) = \gamma_{i-1}(s_0) \cup \gamma_n(s_i)$.

We define the reward associated with a path $\gamma_n(s_i)$ to be the benefit of search at the locations

along $\gamma_n(s_i)$ or

$$J(\gamma_n(s_i)) = \sum_{p_j^k}^{\mathcal{P}_n(s_i)} \mathscr{B}_j^k(\mathcal{D}_n(s_i)|\mathcal{D}_{i-1}(s_0)) \tag{8.26}$$

where $\mathscr{B}_j^k$ denotes the benefit of search defined in equation (8.12) evaluated at the input location $p_j^k$. To avoid cumbersome notation, we define

$$J_n(s_i) \triangleq J(\gamma_n(s_i)). \tag{8.27}$$

Let $\Gamma_n(s_0)$ be the set of all possible $n$-length paths beginning at the joint state $s_0$. Generally, the goal of path planning is to solve for a path $\gamma_n^*(s_0)$ satisfying

$$\gamma_n^*(s_0) = \underset{\gamma_n(s_0)\in\Gamma_n(s_0)}{\arg\max}\ J_n(s_i). \tag{8.28}$$

### 8.3.1 Augmented Receding Horizon Path Planning

In Chapter 5, it is shown that the appropriate use of a terminal reward when constructing receding horizon paths can guarantee a lower bound on the value of receding horizon paths produced. In this work, we utilize naive heuristic paths in receding horizon path planning in an attempt to leverage these lower bound guarantees. The remainder of this subsection describes the assumptions that must be made to directly obtain the lower bound guarantees of Chapter 5.

Let $\hat{\gamma}_{l-i-n-1}(s_{i+n+1})$ be a deterministic naive solution to the $l$-length path planning problem beginning at the joint state $s_i + n + 1$ which is feasible from the final state $s_{i+n}$ in $\gamma_n(s_i)$. Then $\bar{\gamma}_n(s_i) = \gamma_n(s_i) \cup \hat{\gamma}_{l-i-n-1}(s_{i+n+1})$ represents a solution to the $l$-length path planning problem beginning from the joint state $s_i$. We define the reward of a naively augmented

path as

$$\bar{J}(\gamma_n(s_i)) = J(\bar{\gamma}_n(s_i)) = J(\gamma_n(s_i) \cup \hat{\gamma}_{l-i-n-1}(s_{i+n+1})) \tag{8.29}$$

and adopt the simplified notation

$$\bar{J}_n(s_i) \triangleq \bar{J}(\gamma_n(s_i)). \tag{8.30}$$

Consider the augmented path $\bar{\gamma}_n(s_i) = \{s_i, \ldots, s_{i+n}, \hat{s}_{i+n+1}, \ldots, \hat{s}_l\}$. Using the property that for data sets $\mathcal{S}$ and $\mathcal{V}$, $\mathscr{B}(\mathcal{S} \cup \mathcal{V}|\emptyset) = \mathscr{B}(\mathcal{V}|\mathcal{S}) + \mathscr{B}(\mathcal{S}|\emptyset)$, the value of the $m$-length path $\gamma_m(s_i) \subset \bar{\gamma}_n(s_i)$ is

$$J_m(s_i) = \bar{J}_n(s_i) - \sum_{p_j^k}^{\mathcal{P}_{l-m-1}(s_{i+m+1})} \mathscr{B}_j^k(\bar{\mathcal{D}}_{n-m-1}(s_{i+m+1}) \cup \mathcal{D}_m(s_i)|\mathcal{D}_{i-1}(s_0)) \tag{8.31}$$

$$- \sum_{p_j^k}^{\mathcal{P}_m(s_i)} \mathscr{B}_j^k(\bar{\mathcal{D}}_{n-m-1}(s_{i+m+1})|\mathcal{D}_{i+m}(s_0)). \tag{8.32}$$

The first term on the right-hand side of (8.31) gives the reward for the augmented path $\bar{\gamma}_n(s_i)$. The second term gives the total benefit of search along the locations in $\bar{\gamma}_{n-m-1}(s_{i+m+1}) = \bar{\gamma}_n(s_i) \setminus \gamma_m(s_i)$. The third term on the right-hand side of (8.31) gives the additional reduction in risk at the locations in $\gamma_m(s_i)$ due to the data set acquired while traversing $\bar{\gamma}_{n-m-1}(s_{i+m+1})$.

Considering the third term on the right-hand side of (8.31), any additional reduction in risk due to the relatively small data set $\bar{\mathcal{D}}_{n-m-1}(s_{i+m+1})$ is likely negligible in practice. Therefore, we assume that

$$J_m(s_i) \approx \bar{J}_n(s_i) - \sum_{p_j^k}^{\mathcal{P}_{l-m-1}(s_{i+m+1})} \mathscr{B}_j^k(\bar{\mathcal{D}}_{n-m-1}(s_{i+m+1}) \cup \mathcal{D}_m(s_i)|\mathcal{D}_{i-1}(s_0)). \tag{8.33}$$

Similarly, we assume that the second term on the right-hand side of (8.33) may be appropriately approximated as

$$\begin{aligned}&\sum_{p_j^k}^{\mathcal{P}_{l-m-1}(s_{i+m+1})} \mathscr{B}_j^k(\bar{\mathcal{D}}_{n-m-1}(s_{i+m+1}) \cup \mathcal{D}_m(s_i)|\mathcal{D}_{i-1}(s_0)) \\ \approx &\sum_{p_j^k}^{\mathcal{P}_{l-m-1}(s_{i+m+1})} \mathscr{B}_j^k(\bar{\mathcal{D}}_{n-m-1}(s_{i+m+1})|\mathcal{D}_m(s_i) \cup \mathcal{D}_{i-1}(s_0))\end{aligned} \tag{8.34}$$

meaning that the measurements obtained while traversing $\gamma_m(s_i)$ have little effect on the benefit obtained while traversing the locations along $\bar{\gamma}_{n-m-1}(s_{i+m+1})$. Given these assumptions,

$$J_m(s_i) \approx \bar{J}_n(s_i) - \bar{J}_{n-m-1}(s_{i+m+1}) \tag{8.35}$$

where the second term on the right-hand side of (8.35) gives the reward for the augmented path $\bar{\gamma}_{n-m-1}(s_{i+m+1})$.

Given (8.35), Proposition 5.2 guarantees that the value of a receding horizon path produced using augmented paths will be bounded below by $J(\hat{\gamma}_l(s_0))$ under the condition that the $n$-length augmented path $\bar{\gamma}_n(s_i)$ satisfies

$$\bar{J}_n(s_i) \geq \bar{J}_{n-m-1}(s_i) \tag{8.36}$$

at each planning step. $\bar{\gamma}_{n-m-1}(s_i)$ is the remainder of the $n$-length augmented path planned at the previous step.

The condition in (8.36) is difficult to guarantee in practice, as discussed in Section 8.5.1. However, it is shown in Chapter 5 that a joint receding horizon path produced using the Dec-MCTS algorithm [5] (a decentralized variant of MCTS) performs comparably to a method specifically designed to satisfy (8.36) despite no explicit guarantees that (8.36) is satisfied

at each planning step. In this paper, we use the greedy algorithm which is similarly able to achieve the lower bound guarantees of Proposition 5.2 despite no guarantees that (8.36) is satisfied as is shown in Section 8.5.2.

## 8.4 Online Gaussian Process Regression

Recall that the matrix inversion required for GP regression has time complexity $\mathcal{O}(|\mathcal{D}_t|^3)$ where $|\mathcal{D}_t|$ is the cardinality of the data set or the number of samples. Therefore, it is desirable to select a subset of $\mathcal{D}_t$ to perform regression in an online manner. In this section, we provide intuition and justification for the use of the heuristic methods proposed in Section 7.2 of Chapter 7 where a datum is only used for regression if it is sufficiently close to the location of interest and a newly acquired datum is only retained if it is sufficiently far from all existing data. We note that the use of these heuristics does not influence the benefit of search objective function presented in this work so far as the Gaussian process method is used consistently in implementation. Throughout this work we use Gaussian process definitions and notation presented in Chapter 7.

Consider a single datum $d = (p, z)$ and assume the squared exponential kernel given in equation (7.5). Given $\epsilon \in [0, 1]$, the set of locations $p'$ about an input location $p$ satisfying the inequality

$$\exp\left\{\frac{\|p - p'\|^2}{2l_c^2}\right\} \geq \epsilon \tag{8.37}$$

is closed and bounded by the circle of radius $d_\epsilon$ centered at the point $p$ where

$$d_\epsilon = \sqrt{-2l_c \ln \epsilon}. \tag{8.38}$$

If $\epsilon$ is the decimal representation of a percentage value, then $d_\epsilon$ may be considered to be the

distance at which a measurement asserts $\epsilon$ percent of its influence on the point $p'$ in terms of the posterior predictive mean of equation (7.3) assuming regression is performed using the single datum $d$.

Notably $d_\epsilon$ scales with the square root of $l_c$ meaning that, for large values of the characteristic length ($l_c$), $d_\epsilon$ is relatively small. Therefore, when performing regression on the point $p^*$, we use the data subset

$$\hat{\mathcal{D}}_t = \{(p, z) \in \mathcal{D}_t : |p - p^*| \leq d_\epsilon\}. \tag{8.39}$$

Let $\delta_f \geq 0$ be the minimum allowable distance between data proposed in Chapter 7. According to the heuristic proposed in Section 7.2 of Chapter 7, a candidate datum $d_c = (p_c, z_c)$ is only added to the existing data set $\mathcal{D}_t$ if $||p_c - p_i|| \geq \delta_f$ for all $p_i \in \mathcal{D}_t$ with $|| \cdot ||$ being the Euclidean norm.

**Theorem 8.3.** *Enforcing a lower bound on the distance between data ( $||p_i - p_j|| \geq \delta_f$ for all $p_i, p_j \in \mathcal{D}_t$) ensures an upper bound on the condition number of the covariance matrix $K_t$ defined in Section 7.2.*

Although, not explicitly explored in this work, theorem 8.3 implies that one might select a desired upper bound on the condition number $\mathscr{K}(K_t)$ and set a minimum distance between data points $\delta_f$ based upon that bound. It should be noted, however, that there is a trade-off as the prediction error is decreased when kernel centers (samples) are closer together.

## 8.5 Field Trials

In order to demonstrate our augmented receding horizon path planning method in the field, we construct an example problem with the objective of localizing the 15 meter isobath within a bounded area of Claytor Lake in Dublin, VA, USA. The area of interest is shown in Figure

8.2, outlined by a white line. Within this area, we define a prior map of the lake's bathymetry according to (7.9) where the blue line marks the center line.

In order to localize the isobath, teams of Virginia Tech 690 AUVs (shown in Figure 8.1) were deployed and programmed to autonomously produce receding horizon paths that seek to maximize the path value function given in (8.29) indirectly using equation (8.49) and utilizing the benefit of search objective function described in equation (8.12). Each 690 AUV is equipped with a suite of navigation sensors including a KVH 1750 inertial measurement unit (IMU), Teledyne Pathfinder Doppler velocity log (DVL), pressure sensor, WHOI micromodem 2 acoustic modem, and GPS receiver. The inertial navigation system (INS) on the vehicle uses an unscented Kalman filter on manifolds (UKF-M) [9] to fuse the data from each sensor into an estimate of the vehicle's attitude, velocity, and position as described in [45]. Additionally, each AUV is able to use its acoustic modem to communicate with other AUVs, although with severely limited bandwidth. Details regarding the 690 AUV are provided in Section 7.4.

Field trials were conducted using teams of 2 and 3 Virginia Tech 690 AUVs. For each team size, two receding horizon path planning methods are tested. One method utilizes a naive heuristic to augment short horizon paths while the other does not. The naive heuristic paths used in these field trials are simple lawnmower paths examples of which are shown in Figure 8.4. All receding horizon paths produced are 100 steps long. The planning horizon $n$ for the method that includes naive heuristic paths is 3 steps. The planning horizon for the method that does not utilize augmented paths is 10 steps.

Figure 8.1: 690 AUVs

The benefit of search gives the reduction in Bayes' risk due to obtaining a set of depth measurements at locations throughout the operational area. The depth of interest ($l$ in (8.2)) is 15 meters. The costs of (8.4) are equal in these experiments ($c_1 = c_2 = 10$). Therefore, the locations with the highest Bayes' risk throughout the operational area are associated with the expected location of the 15 meter isobath.

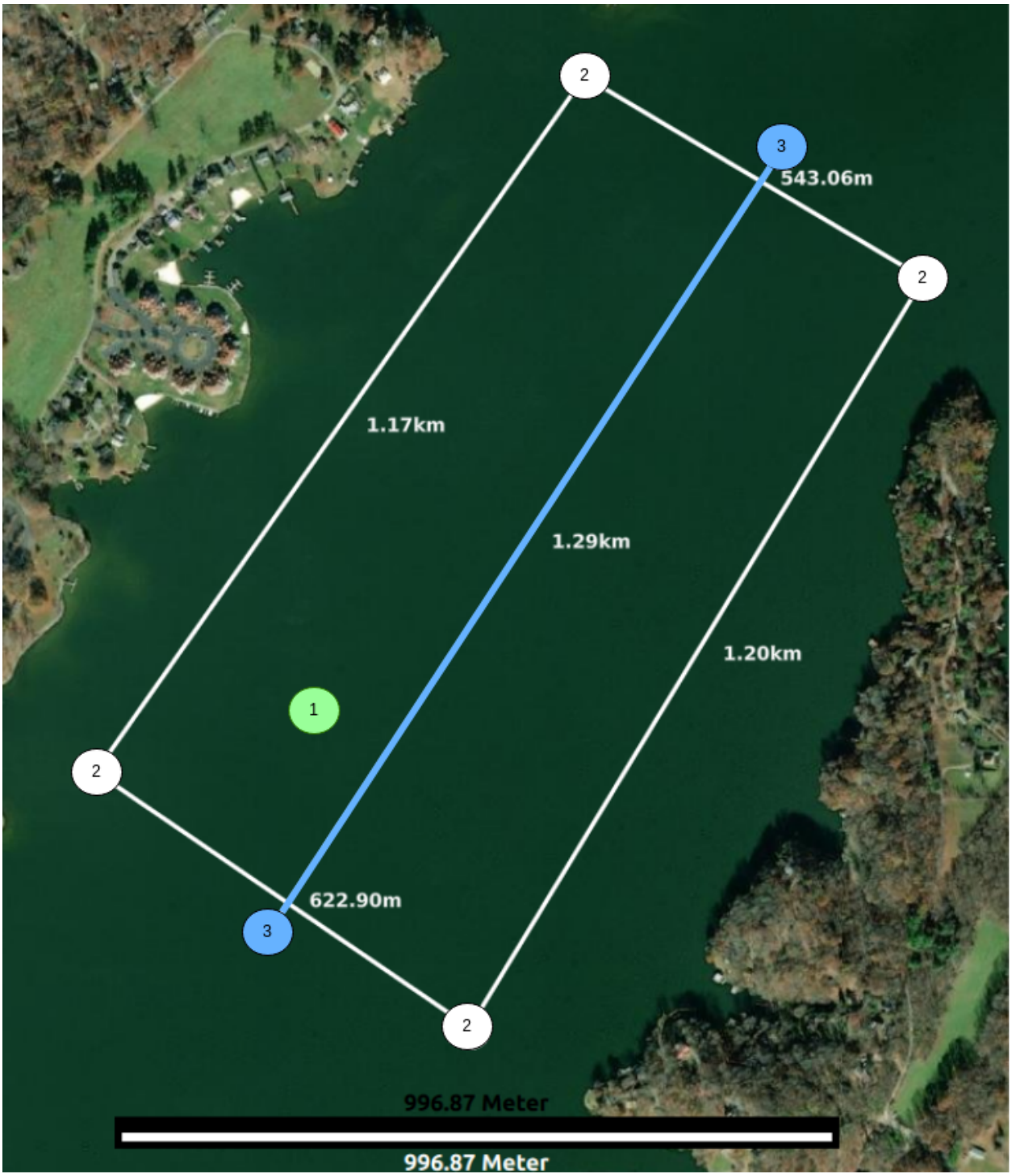



Figure 8.2: The mission start waypoint and local tangent frame origin are shown as the green circle marked with the number 1. The boundaries of the operational area are shown by the white lines with the number 2 in the circles where boundary lines intersect. The line defining the deepest portion of the prior mean is shown as the blue line with the number 3 in blue circles at each end of the line. The values shown next to the lines give the length of each line segment.

Path planning for each vehicle is performed in a local tangent frame with its origin marked by a green dot with the number 1 in Figure 8.2. Vehicle position in the local tangent frame is described by its Northing, Easting, and downward position from the origin in meters. To prevent collision between agents, each agent operates at its own fixed depth of 3, 4, and 5 meters for Agents 1, 2, and 3 respectively.

In planning, the vehicle state is given as $s = \{x_h, p_n, p_e\}$ where $x_h$ gives the heading in radians and $p_n$ and $p_e$ are the North and East components of the position of the state in meters from the local tangent frame origin. Additionally, we account for vehicle dynamics in the form of a limited turn radius. The set of next states available from any current state are based on a Dubins' type motion model. Specifically, given the turning radius $r$, a maximum rotation about a minimum turning circle (Either of the circles shown in Figure 8.3), and a desired change in heading $a \in (-\pi, \pi]$, define

$$d = r(\theta_{max} + |a|) \tag{8.40}$$

$$\Delta_x = \text{sign}(a)[r - r\cos(|a|) + d\sin(|a|)] \tag{8.41}$$

$$\Delta_y = r\sin(|a|) + d\cos(|a|). \tag{8.42}$$

where, for our scenario,

$$r = 15 \tag{8.43}$$

$$\theta_{max} = 90 \tag{8.44}$$

$$a \in \mathcal{A} = \{-90, -30, -20, -10, 5, 0, 5, 10, 20, 30, 90\}. \tag{8.45}$$

where the values of $\theta_{max}$ and $\mathcal{A}$ are given in degrees for convenience. The discrete dynamics

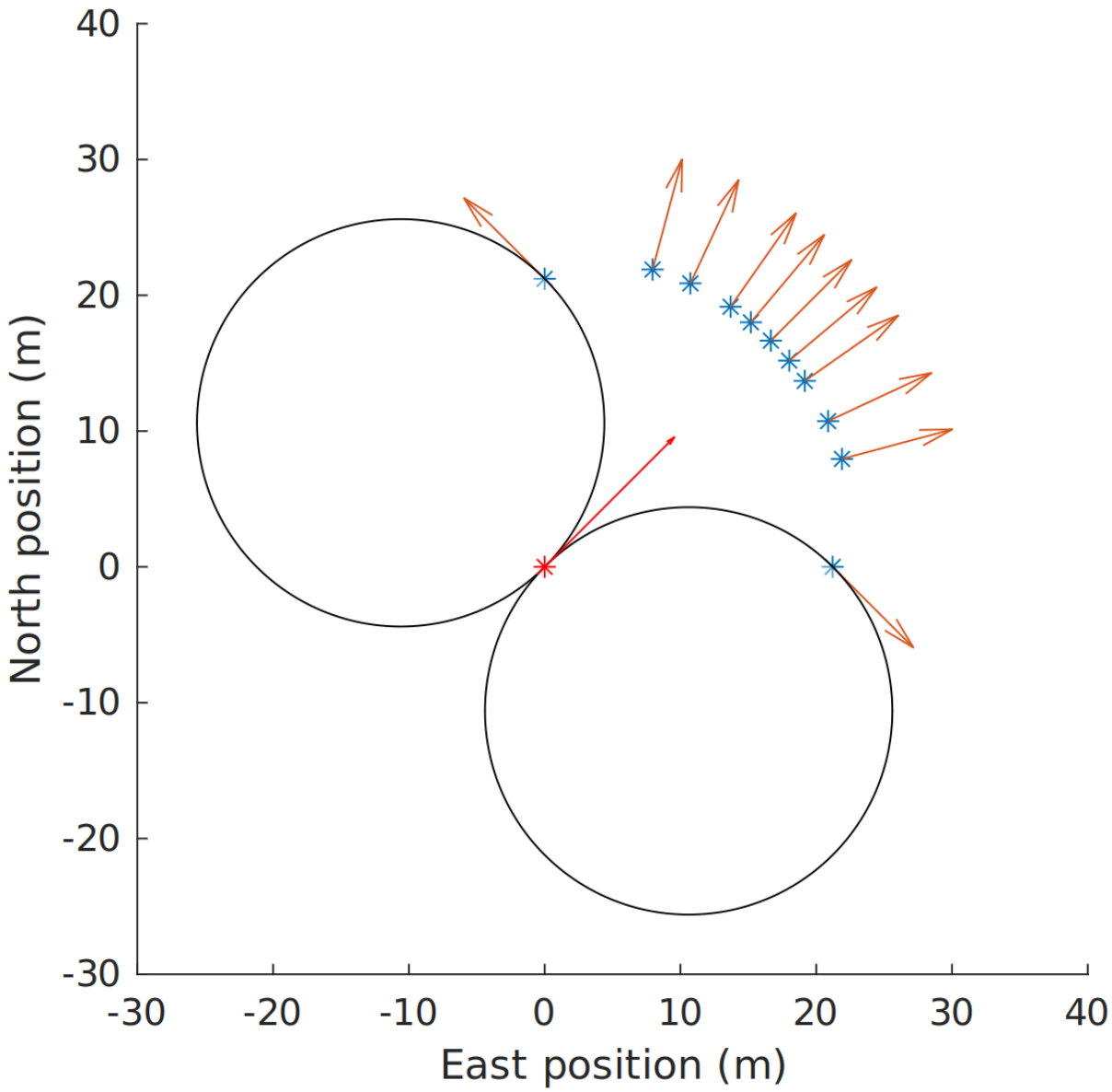



Figure 8.3: Visual representation of discrete dynamics. Initial state is shown as the red asterisk with attached arrow. Minimum turning circles are shown at either side of the initial state. Each blue asterisk with its attached arrow represents the result of commanding a change in heading from the initial state.

defining the next state for an agent are given as

$$x_h[t+1] = x_h[t] + a \tag{8.46}$$

$$p_n[t+1] = \cos(-x_h[t])\Delta_x - \sin(-x_h[t])\Delta_y \tag{8.47}$$

$$p_e[t+1] = \sin(-x_h[t])\Delta_x + \cos(-x_h[t])\Delta_y. \tag{8.48}$$

A visualisation of the next states available to an agent is shown in Figure 8.3. The red asterisk gives the initial position of the agent with the associated arrow giving the initial heading. The next states resulting from each $a \in \mathcal{A}$ are shown as blue asterisks with the attached arrows giving the associated headings.

Joint path planning is accomplished via the greedy algorithm wherein an ordering is assigned to the team of agents and each agent plans a path for itself considering only the paths of other agents that precede its position in the ordering. The use of the greedy algorithm is inspired by the greedy type algorithm proposed in Section 5.3. Additionally, because the benefit of search is both normalized and monotone, the greedy algorithm is expected to yield a high quality solution given the theoretical lower bounds presented in Chapter 6 despite the benefit of search having no proven submodularity properties.

Each agent plans paths for itself using the Monte-Carlo tree search method described in Section 7.5.4 with each agent seeking to maximize the additional reward that it contributes to the reward obtained by the paths of preceding agents per the greedy algorithm. Explicitly, at each planning step, agent $k$ seeks to maximize

$$J(\bar{\gamma}_n^k(s_i)) = \sum_{p_j^k}^{\bar{\mathcal{P}}_n^k(s_i)} \mathscr{B}_j^k(\bar{\mathcal{D}}_n^k(s_i)|\mathcal{D}_{i-1}(s_0) \cup \bar{\mathcal{D}}_n^{1:k-1}(s_i)) \tag{8.49}$$

where $\bar{\mathcal{P}}_n^k(s_i)$ is the set of locations corresponding to the augmented path $\bar{\gamma}_n^k(s_i)$ of the

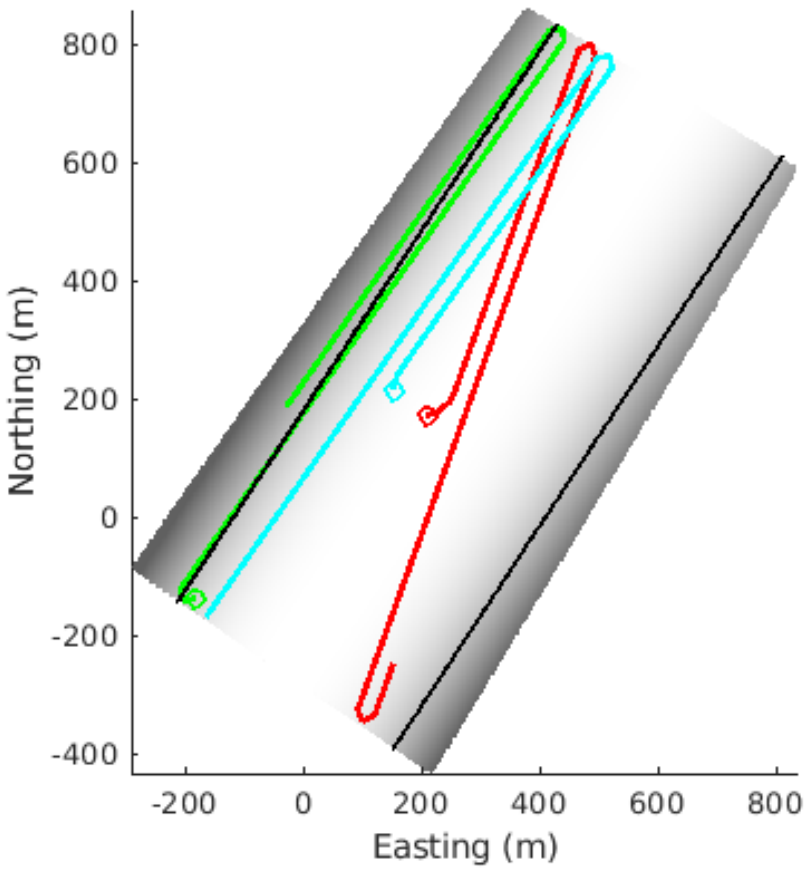



Figure 8.4: Short horizon paths with appended lawn mower paths. Circles indicates the path start locations.

$k^{\text{th}}$ agent, $\bar{\mathcal{D}}_n^k(s_i)$ is the data set acquired by the $k^{\text{th}}$ agent while traversing $\bar{\gamma}_n^k(s_i)$, and $\bar{\mathcal{D}}_n^{1:k-1}(s_i)$ is the data set acquired by the set of agents preceding agent $k$ in the greedy ordering while traversing the augmented joint path $\bar{\gamma}_n^{1:k-1}(s_i)$. In planning, the data sets $\bar{\mathcal{D}}_n^k(s_i)$ and $\bar{\mathcal{D}}_n^{1:k-1}(s_i)$ are not yet available and therefore the benefit of search must be computed in expectation using the expected Bayes' risk in equation (8.25). Figure 8.4 shows the best augmented path known to the team of agents at a point in time during the construction of the receding horizon path depicted in Figure 8.8a.

In order to facilitate communication between vehicles underwater using each vehicle's acoustic modem, a compact packet structure for communication of path an depth measurement

data was designed. A time division multiple access (TDMA) scheme is used to prevent acoustic interference between vehicles while transmitting data. Each agent is periodically allotted a 10 second window of time to transmit a single packet of data. The acoustic packets consist of 252 bytes. The first 14 bytes in a packet describe the initial state in the current best path known to the agent. The initial state bytes are followed by a sequence of single bytes describing the actions taken to construct the path of the desired length. A terminal byte describes the end of the action sequence. When using naive paths to augment the short horizon paths, the terminal byte takes a value describing the deterministic heuristic used to produce the naive path. In this way, long paths may be communicated quite compactly. Following the terminal byte for the path, depth measurements are appended to the packet as groups of 12 bytes representing the north and east position of the measurement in the local tangent frame as well as the at that position depth. Given the packet size constraints, approximately 18 depth samples are communicated in a packet depending on the length of the short horizon path. To select which measurements to communicate, each agent maintains a history of which measurements have been communicated and which have not. From the time-ordered list of samples that have not been communicated, the agent selects every $n^{\text{th}}$ sample where $n$ is selected according to the number of samples that may be transmitted in a single packet.

### 8.5.1 Theory vs. Practice

It is important to note that the actual implementation of these experiments differs from some of the assumptions underpinning the theoretical guarantees of Proposition 5.2. Disparities include the following:

1. **Asynchronous path traversal**: One agent may traverse a mission length path faster

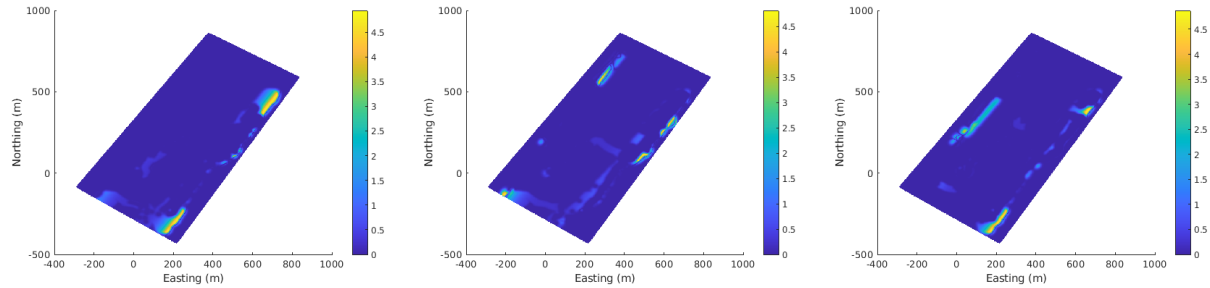
than another, one agent may reach the start waypoint before another agent, etc...

2. **The greedy algorithm used in receding horizon path planning is not guaranteed to satisfy the condition of** (8.36): While the greedy algorithm as implemented in these field trials is not explicitly guaranteed to satisfy the condition of (8.36), the greedy algorithm requires very little communication and is guaranteed to produce near optimal solutions when using an appropriate objective function as discussed in Chapter 6. As such, given the results of near-optimal planning methods demonstrated in Chapters 4 and 5, the authors believe the greedy algorithm to be an appropriate planning method for these experiments.

3. **Local approximations of the global objective function**: Each agent only has access to a local approximation of the global objective function which accounts for all measurements obtained by all agents. Each local approximation accounts for all measurements obtained by the corresponding agent, but the measurements obtained by all other agents are only available as they are communicated via acoustic channels. An example of the error between the posterior Bayes' risk for each agents vs the true posterior risk given all measurements from all agents is shown in Figure 8.5. Notably, given the limited communication bandwidth and the probability of data not being received at all, the error in each local approximation is small.

4. **Slow communication prohibits ideal coordination in path planning**: The use of the TDMA protocol allows each agent to transmit path planning data only once per TDMA cycle. Thus, a 10 second window ensures that each agent is able to broadcast only once every 30 seconds given a team of 3 agents. In this time, an agent may have an entirely different best path for planning.

5. **Real samples do not precisely match planned sample locations**: The Pathfinder




(a) (b) (c)

Figure 8.5: Panels (a), (b), and (c) show the difference between the posterior Bayes' risk given the measurements available to agents 1, 2, and 3 respectively and the posterior Bayes' risk given all measurements shown in Figure 8.8a. Note that all measurements not obtained by the agent itself must be received via acoustic communications.

DVL uses 4 beams to obtain depth measurements at a rate of 5Hz. It is assumed that some of the depth measurements retained according to the methods described in Section 8.4 are close to the planned sample locations although it is highly unlikely that measurements are obtained precisely at the planned sample locations.

Fortunately, these disparities between theory and practice do not appear to have an appreciable negative effect on the results of these experiments with respect to the performance guarantees of Proposition 5.2.

### 8.5.2 Results of Field Trials

The results for experiments conducted using 2 Virginia Tech 690 vehicles are presented in Figures 8.6 and 8.7 while results for experiments conducted using 3 690 vehicles are presented in Figures 8.8 and 8.9. In Figures 8.6 and 8.8, the prior Bayes' risk in the search environment is shown in panels (a) and (b). Panel (c) shows the posterior Bayes' risk given all measurements obtained by all agents while traversing the joint receding horizon path shown in panel (a). Panel (d) shows the posterior Bayes' risk given all measurements obtained by all agents while traversing the joint receding horizon path shown in panel (b). The paths

shown in panel (a) were constructed using appended naive heuristic paths. The paths shown in panel (b) were constructed without the use of heuristic paths. For each path, all agents begin at the location (0,0).

Figures 8.7 and 8.9 show the total reward accumulated by the team of search agents for each planning method. The dashed blue line gives the total reward accumulated by the joint receding horizon path produced without the use of naive heuristic paths. The solid green line gives the total reward accumulated by the joint receding horizon path produced using of naive heuristic paths. The dashed red line gives the time-varying lower bound given in (7.15) while the solid black line gives the maximum value of the time-varying lower bound.

Figures 8.7 and 8.9 show that both receding horizon paths perform significantly better than the value of the initial lower bound or the value of the dashed red line at $t = 0$. Additionally, the receding horizon path produced using naive heuristic paths performs no worse than the receding horizon path that did not include terminal rewards. While there is no guarantee that paths produced using the proposed method will perform better or worse than receding horizon paths produced using other methods, these results do not indicate any degradation in performance as compared to the receding horizon paths produced in a more traditional manner. Furthermore, the results of these experiments appear to demonstrate that the lower bound guarantees of 5.2 may be achieved despite non-idealities such as those described in Section 8.5.1.

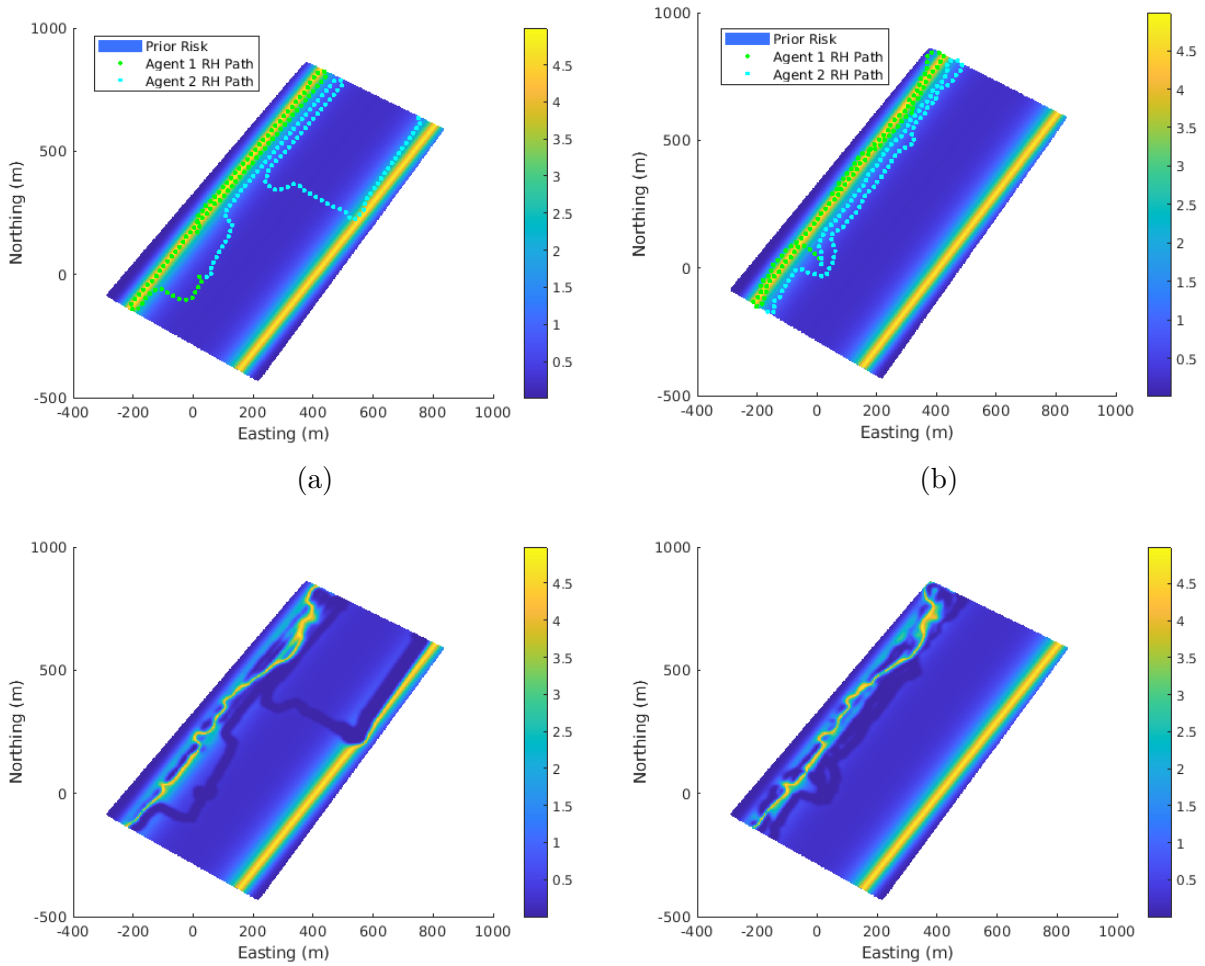



Figure 8.6: Paths and Bayes' risk for experiments accomplished using 2 vehicles. Panels (a) and (b) show the prior Bayes' risk while panels (c) and (d) show the posterior Bayes' risk.

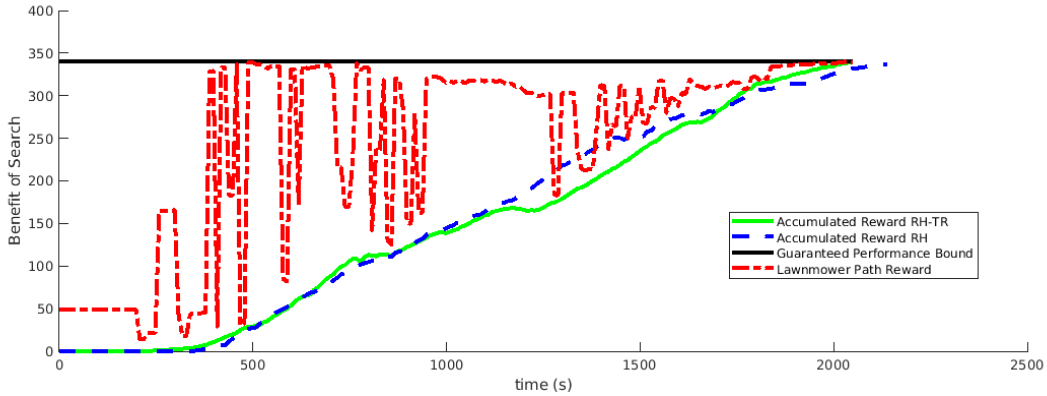



Figure 8.7: Results of receding horizon missions conducted with 2 vehicles.

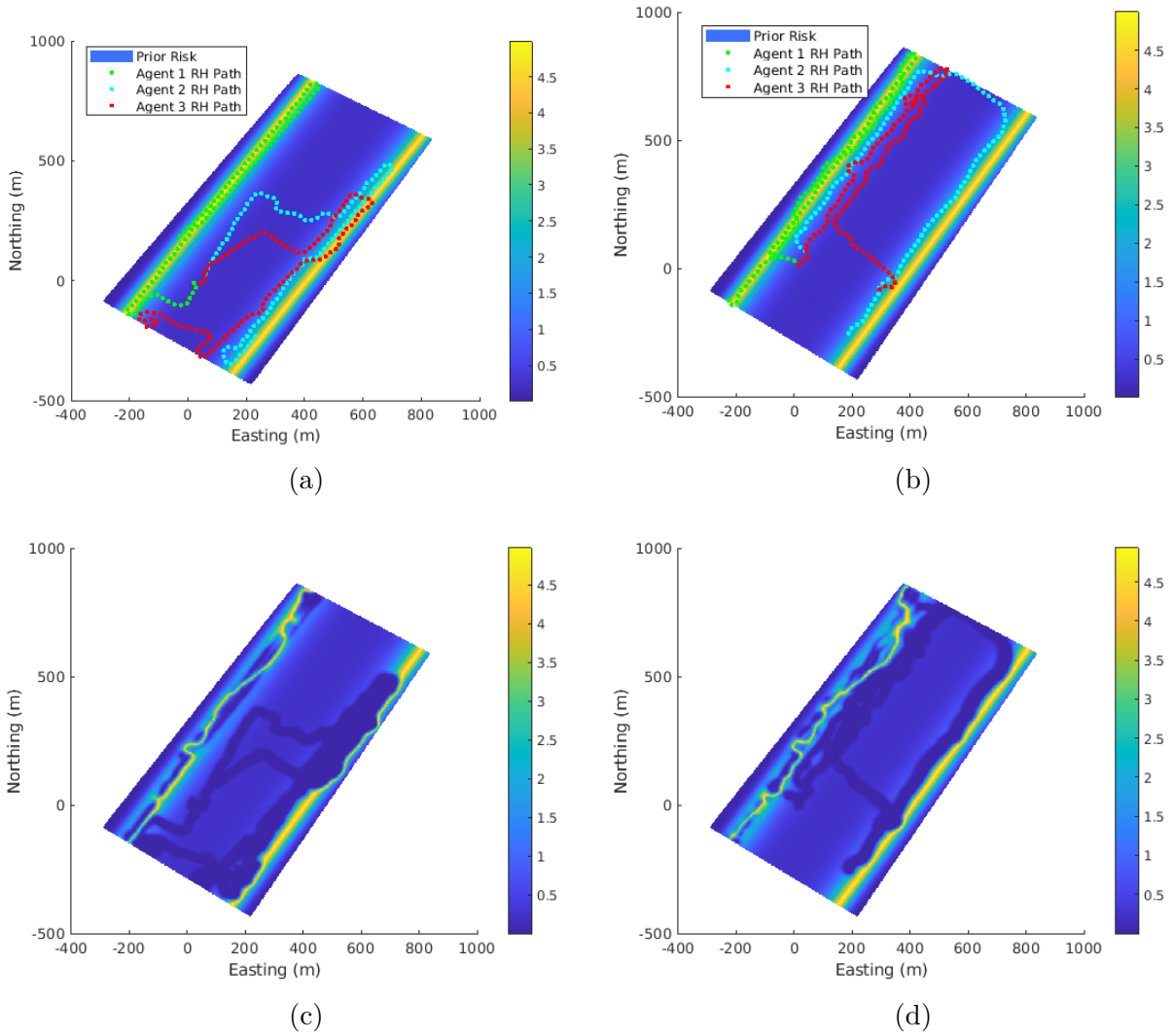



Figure 8.8: Paths and Bayes' risk for experiments accomplished using 3 vehicles. Panels (a) and (b) show the prior Bayes' risk while panels (c) and (d) show the posterior Bayes' risk.

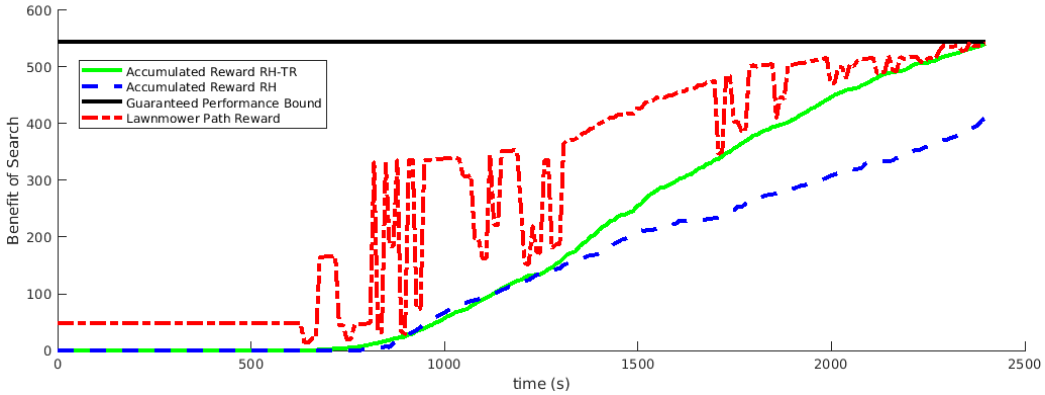



Figure 8.9: Results of receding horizon missions conducted with 3 vehicles.

## 8.6 Proofs

### 8.6.1 Proof of Lemma 8.2

1. **Normalized:**

*Proof.* It is trivial to see that the benefit of search is normalized because given any data set $\mathcal{D}$

$$\mathscr{B}(\emptyset|\mathcal{D}) = r(\mathcal{D}) - r(\emptyset \cup \mathcal{D}) = 0. \tag{8.50}$$

□

2. **Monotone:**

*Proof.* Let $\mathcal{S}, \mathcal{V}$ be data sets with $\mathcal{V} = \mathcal{H} \cup \mathcal{K}$. Let $Z_{\mathcal{S}}, Z_{\mathcal{V}}, Z_{\mathcal{H}}$ and $Z_{\mathcal{K}}$ be the measurement vectors corresponding to $\mathcal{S}, \mathcal{V}, \mathcal{H}$, and $\mathcal{K}$ respectively. To be monotone, it must be true that

$$\mathbb{E}_{Z_{\mathcal{V}}}\left[\mathscr{B}(\mathcal{V}|\mathcal{S})\right] \geq \mathbb{E}_{Z_{\mathcal{H}}}\left[\mathscr{B}(\mathcal{H}|\mathcal{S})\right] \tag{8.51}$$

or equivalently

$$\mathbb{E}_{Z_{\mathcal{V}}}\left[r(\mathcal{V} \cup \mathcal{S})\right] \leq \mathbb{E}_{Z_{\mathcal{H}}}\left[r(\mathcal{H} \cup \mathcal{S})\right]. \tag{8.52}$$

$$\mathbb{E}_{Z_\mathcal{V}}\left[r(\mathcal{V}\cup\mathcal{S})\right] \tag{8.53}$$

$$= \int_{\mathbb{R}^{|\mathcal{H}|}}\int_{\mathbb{R}^{|\mathcal{K}|}} \pi(Z_\mathcal{H}, Z_\mathcal{K}|Z_\mathcal{S}) \int_\mathbb{R} \pi(f_p|Z_\mathcal{H}, Z_\mathcal{K}, Z_\mathcal{S})\mathscr{L}(f_p, \delta^*(\mathcal{H},\mathcal{K},\mathcal{S}))df_p dZ_\mathcal{V} \tag{8.54}$$

$$\leq \int_{\mathbb{R}^{|\mathcal{H}|}}\int_{\mathbb{R}^{|\mathcal{K}|}} \pi(Z_\mathcal{H}, Z_\mathcal{K}|Z_\mathcal{S}) \int_\mathbb{R} \pi(f_p|Z_\mathcal{H}, Z_\mathcal{K}, Z_\mathcal{S})\mathscr{L}(f_p, \delta^*(\mathcal{H},\mathcal{S}))df_p dZ_\mathcal{H} dZ_\mathcal{K} \tag{8.55}$$

$$= \int_{\mathbb{R}^{|\mathcal{H}|}}\int_{\mathbb{R}^{|\mathcal{K}|}} \pi(Z_\mathcal{H}, Z_\mathcal{K}|Z_\mathcal{S}) \int_\mathbb{R} \pi(f_p|Z_\mathcal{H}, Z_\mathcal{K}, Z_\mathcal{S})dZ_\mathcal{K}\mathscr{L}(f_p, \delta^*(\mathcal{H},\mathcal{S}))df_p dZ_\mathcal{H} \tag{8.56}$$

$$= \int_{\mathbb{R}^{|\mathcal{H}|}}\int_{\mathbb{R}^{|\mathcal{K}|}} \pi(Z_\mathcal{K}|Z_\mathcal{H}, Z_\mathcal{S})\pi(Z_\mathcal{H}|Z_\mathcal{S}) \int_\mathbb{R} \pi(f_p|Z_\mathcal{H}, Z_\mathcal{K}, Z_\mathcal{S})dZ_\mathcal{K}\mathscr{L}(f_p, \delta^*(\mathcal{H},\mathcal{S}))df_p dZ_\mathcal{H} \tag{8.57}$$

$$= \int_{\mathbb{R}^{|\mathcal{H}|}} \pi(Z_\mathcal{H}|Z_\mathcal{S}) \int_\mathbb{R}\int_{\mathbb{R}^{|\mathcal{K}|}} \pi(f_p, Z_\mathcal{K}|Z_\mathcal{H}, Z_\mathcal{S})dZ_\mathcal{K}\mathscr{L}(f_p, \delta^*(\mathcal{H},\mathcal{S}))df_p dZ_\mathcal{H} \tag{8.58}$$

$$= \int_{\mathbb{R}^{|\mathcal{H}|}} \pi(Z_\mathcal{H}|Z_\mathcal{S}) \int_\mathbb{R} \pi(f_p|Z_\mathcal{H}, Z_\mathcal{S})\mathscr{L}(f_p, \delta^*(\mathcal{H},\mathcal{S}))df_p dZ_\mathcal{H} \tag{8.59}$$

$$= \mathbb{E}_{Z_\mathcal{H}}\left[r(\mathcal{H}\cup\mathcal{S})\right] \tag{8.60}$$

where the inequality in (8.55) is because $\delta^*(\mathcal{H},\mathcal{S})$ is not the minimizing estimate for the augmented data set. □

### 8.6.2 Proof of Theorem 8.3

*Proof.* Let $k_t$ be the real, positive definite, symmetric matrix defined as $[K_t]_{i,j} = k(p_i, p_j)$ where $k(\cdot,\cdot)$ is the squared exponential kernel given in equation (7.5). Recall that using the Euclidean norm $||\cdot||$, the condition number of $K_t$ is

$$\mathscr{K}(K_t) = ||K_t||\ ||K_t^{-1}||. \tag{8.61}$$

Because $K_t$ is real-valued, symmetric, and positive definite the eigenvalues of $K_t$ are real and positive. Furthermore,

$$\mathscr{K}(K_t) = \frac{\lambda_{\max}(K_t)}{\lambda_{\min}(K_t)} \tag{8.62}$$

where $\lambda_{\min}(K_t)$ and $\lambda_{\max}(K_t)$ are the minimum and maximum eigenvalues of $K_t$. Additionally, note that the squared exponential kernel given in equation (7.5) ensures that $k(p_i, p_i) = k(p_j, p_j) \in \mathbb{R}$ for any points $p_i, p_j \in \mathscr{P}$. Define

$$\bar{k} \triangleq k(p_i, p_i). \tag{8.63}$$

Enforcing a minimum distance between data points ensures that there exists some $\delta \in \mathbb{R}$ satisfying

$$\delta \geq (n-1)k(p_i, p_j) \quad \forall\ p_i, p_j \in \mathcal{D}_t \tag{8.64}$$

where $n = |\mathcal{D}_t|$. Therefore

$$\sum_{j \neq i} k(p_i, p_j) \leq \delta. \tag{8.65}$$

By Gershgorin circle theorem [35], the eigenvalues of $K_t$ are contained in

$$\{c \in \mathbb{R} : |c - \bar{k}| \leq \delta\} \tag{8.66}$$

such that

$$\lambda_{\max}(K_t) \leq \bar{k} + \delta \tag{8.67}$$

$$\lambda_{\min}(K_t) \geq \bar{k} - \delta. \tag{8.68}$$

Therefore

$$\mathscr{K}(K_t) \leq \frac{\bar{k} + \delta}{\bar{k} - \delta}. \tag{8.69}$$

□

### 8.6.3 Derivation of $\mu^*$

Using the definition of the Gaussian CDF,

$$\pi(f_p < l|\mu^*, \sigma^2_{p|\mathcal{Q}}) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{l-\mu^*}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)\right]. \tag{8.70}$$

Setting the right-hand side to $\frac{c_2}{c_1+c_2}$ gives

$$\frac{1}{2}\left[1 + \text{erf}\left(\frac{l-\mu^*}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)\right] = \frac{c_2}{c_1+c_2} \tag{8.71}$$

$$1 + \text{erf}\left(\frac{l-\mu^*}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) = \frac{2c_2}{c_1+c_2} \tag{8.72}$$

$$\text{erf}\left(\frac{l-\mu^*}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) = \frac{c_2-c_1}{c_1+c_2} \tag{8.73}$$

$$\frac{l-\mu^*}{\sigma_{p|\mathcal{Q}}\sqrt{2}} = \text{erf}^{-1}(\frac{c_2-c_1}{c_1+c_2}) \tag{8.74}$$

$$l-\mu^* = \text{erf}^{-1}(\frac{c_2-c_1}{c_1+c_2})\sigma_{p|\mathcal{Q}}\sqrt{2} \tag{8.75}$$

$$-\mu^* = \text{erf}^{-1}(\frac{c_2-c_1}{c_1+c_2})\sigma_{p|\mathcal{Q}}\sqrt{2} - l \tag{8.76}$$

$$\mu^* = l - \text{erf}^{-1}(\frac{c_2-c_1}{c_1+c_2})\sigma_{p|\mathcal{Q}}\sqrt{2} \tag{8.77}$$

### 8.6.4 Computing Expected Risk 1

$$\mathbb{E}_{\mu}[r(\mathcal{Q})] \tag{8.78}$$

$$= c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})\pi(f_p \geq l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu + c_1 \int_{\mu^*}^{\infty} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu$$

$$= c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) \left(1 - \pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})\right) d\mu + c_1 \int_{\mu^*}^{\infty} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu \tag{8.79}$$

$$= c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})d\mu + c_1 \int_{-\infty}^{\infty} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu \tag{8.80}$$

$$- (c_1 + c_2) \int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}})d\mu \tag{8.81}$$

The CDF of the Gaussian distribution may be found as

$$\pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}}) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{l - \mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)\right] \tag{8.82}$$

such that

$$\mathbb{E}_\mu[r(\mathcal{Q})] \tag{8.83}$$

$$= c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu + c_1 \int_{-\infty}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}}) d\mu \tag{8.84}$$

$$- (c_1 + c_2) \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \pi(f_p < l|\mu, \sigma^2_{p|\mathcal{Q}}) d\mu$$

$$= c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu + \frac{c_1}{2} \int_{-\infty}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \left[1 + \text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)\right] d\mu \tag{8.85}$$

$$- \frac{(c_1 + c_2)}{2} \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \left[1 + \text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)\right] d\mu$$

$$= c_2 \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu \tag{8.86}$$

$$+ \frac{c_1}{2} \int_{-\infty}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu + \frac{c_1}{2} \int_{-\infty}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu$$

$$- \frac{(c_1 + c_2)}{2} \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu - \frac{(c_1 + c_2)}{2} \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu \tag{8.87}$$

$$= \frac{(c_2 - c_1)}{2} \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu \tag{8.88}$$

$$+ \frac{c_1}{2} \int_{-\infty}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu + \frac{c_1}{2} \int_{-\infty}^{\infty} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu$$

$$- \frac{(c_1 + c_2)}{2} \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) \text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu.$$

Evaluating the terms of (8.88) yields the following. The integral of the first term of (8.88) evaluates to

$$\int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu = \frac{1}{2}\left[1 + \text{erf}\left(\frac{\mu^* - \mu_\mu}{\sigma_\mu\sqrt{2}}\right)\right] \tag{8.89}$$

where $\mu_\mu$ and $\sigma_\mu$ are the mean and standard deviation of the distribution $\pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})$. The

integral of the second term is

$$\int_{-\infty}^{\infty} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) d\mu = 1 \tag{8.90}$$

It is given in [56] that

$$\int_{-\infty}^{\infty} \operatorname{erf}(ax+b) \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) dx = \operatorname{erf} \frac{a\mu + b}{\sqrt{1+2a^2\sigma^2}}, \qquad a, b, \mu, \sigma \in \mathbb{R} \tag{8.91}$$

meaning that the integral in the third term of (8.88) evaluates to

$$\int_{-\infty}^{\infty} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) \operatorname{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu = \operatorname{erf}\left(\frac{l-\mu_\mu}{\sqrt{2\sigma_{p|\mathcal{Q}}^2 + 2\sigma_\mu^2}}\right). \tag{8.92}$$

We may now write the expected risk as

$$\begin{aligned}\mathbb{E}_\mu[r(\mathcal{Q})] = \frac{(c_2-c_1)}{4}\left[1+\operatorname{erf}\left(\frac{\mu^*-\mu_\mu}{\sigma_\mu\sqrt{2}}\right)\right] + \frac{c_1}{2} + \frac{c_1}{2}\operatorname{erf}\left(\frac{l-\mu_\mu}{\sqrt{2\sigma_{p|\mathcal{Q}}^2+2\sigma_\mu^2}}\right) \\ - \frac{(c_1+c_2)}{2}\int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) \operatorname{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu.\end{aligned} \tag{8.93}$$

We now need to solve for

$$\int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) \operatorname{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu. \tag{8.94}$$

An approximate solution to the integral of the fourth term in (8.88) is provided in Appendix 8.6.5.

### 8.6.5 Computing Expected Risk 2

To the best of the authors' knowledge, an indefinite solution to the integral in the fourth term of (8.88) has not been found despite the definite integral over $\mathbb{R}$ being known [56]. However, using an approximation of the non-elementary error function allows us to evaluate the integral. Notably, the form of the approximation to the error function is important as the integral of the product of the approximate error function with the Gaussian probability density function is non-trivial even in the simplest cases [41, 56]. However, an appropriate approximation to the error function termed an *exponential-quadratic approximation* was proposed in [59] as

$$\operatorname{erf}(x) \approx 1 - \sum_{i=1}^{4} c_i \exp(-a_i x^2 + 2b_i x) \quad \text{for } x \in \mathbb{R}_{\geq 0} \tag{8.95}$$

that has a form conducive to integration as will be seen shortly. The coefficients $a_i, b_i$ and $c_i$ may be found in [59]. Throughout Appendix 8.6.5, $c_i$ refers to the constant defined in [59] and not the cost discussed throughout the rest of this work.

Because erf is an odd function, we have that $-f(x) = f(-x)$, so for $x \in \mathbb{R}_{<0}$, and

$$\operatorname{erf}(x) \approx \begin{cases} \sum_{i=1}^{4} c_i \exp(-a_i x^2 - 2b_i x) - 1 & \text{for } x \in \mathbb{R}_{<0} \\ 1 - \sum_{i=1}^{4} c_i \exp(-a_i x^2 + 2b_i x) & \text{for } x \in \mathbb{R}_{\geq 0} \end{cases} . \tag{8.96}$$

For $x = \frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}$ this becomes

$$\operatorname{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \approx \begin{cases} \sum_{i=1}^{4} c_i \exp\left(\frac{-a_i l^2 + 2a_i l\mu - a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i\mu - 2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) - 1 & \text{for } \mu \geq l \\ 1 - \sum_{i=1}^{4} c_i \exp\left(\frac{-a_i l^2 + 2a_i l\mu - a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i l - 2b_i\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) & \text{for } \mu < l \end{cases} \tag{8.97}$$

We now have the product

$$\pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}})\mathrm{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \tag{8.98}$$

$$\approx \begin{cases} \pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}})\sum_{i=1}^{4} c_i \exp\left(\frac{-a_i l^2+2a_i l\mu-a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i\mu-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right)-\pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}}) & \text{for } \mu \geq l \\ \pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}})-\pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}})\sum_{i=1}^{4} c_i \exp\left(\frac{-a_i l^2+2a_i l\mu-a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i l-2b_i\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) & \text{for } \mu < l \end{cases}$$

Note that

$$\pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}})\sum_{i=1}^{4} c_i \exp\left(\frac{-a_i l^2+2a_i l\mu-a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i\mu-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \tag{8.99}$$

$$=\sum_{i=1}^{4} c_i\pi(\mu|Z_{\mathcal{S}},\mathcal{P}_{\mathcal{Q}}) \exp\left(\frac{-a_i l^2+2a_i l\mu-a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i\mu-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \tag{8.100}$$

$$=\sum_{i=1}^{4} c_i \left(\frac{1}{\sigma_\mu\sqrt{2\pi}}\exp\left(\frac{-\mu^2+2\mu\mu_\mu-\mu_\mu^2}{2\sigma_\mu^2}\right)\right)\exp\left(\frac{-a_i l^2+2a_i l\mu-a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i\mu-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \tag{8.101}$$

$$=\sum_{i=1}^{4} \frac{c_i}{\sigma_\mu\sqrt{2\pi}}\exp\left(\Theta_{2,i}\right) \tag{8.102}$$

where

$$\Theta_{2,i}=\frac{-\mu^2+2\mu\mu_\mu-\mu_\mu^2}{2\sigma_\mu^2}+\frac{-a_i l^2+2a_i l\mu-a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i\mu-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.103}$$

$$=\frac{-\mu^2}{2\sigma_\mu^2}+\frac{2\mu\mu_\mu}{2\sigma_\mu^2}-\frac{\mu_\mu^2}{2\sigma_\mu^2}+\frac{-a_i l^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2a_i l\mu}{2\sigma_{p|\mathcal{Q}}^2}-\frac{a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}-\frac{2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.104}$$

$$=-\mu^2\left(\frac{1}{2\sigma_\mu^2}+\frac{a_i}{2\sigma_{p|\mathcal{Q}}^2}\right)+\mu\left(\frac{2\mu_\mu}{2\sigma_\mu^2}+\frac{2a_i l}{2\sigma_{p|\mathcal{Q}}^2}+\frac{2b_i}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \tag{8.105}$$

$$+\left(\frac{-\mu_\mu^2}{2\sigma_\mu^2}+\frac{-a_i l^2}{2\sigma_{p|\mathcal{Q}}^2}+\frac{-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right).$$

Similarly, we define

$$\Theta_{1,i} = \frac{-\mu^2 + 2\mu\mu_\mu - \mu_\mu^2}{2\sigma_\mu^2} + \frac{-a_i l^2 + 2a_i l\mu - a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i l - 2b_i\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.106}$$

$$= \frac{-\mu^2}{2\sigma_\mu^2} + \frac{2\mu\mu_\mu}{2\sigma_\mu^2} - \frac{\mu_\mu^2}{2\sigma_\mu^2} + \frac{-a_i l^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2a_i l\mu}{2\sigma_{p|\mathcal{Q}}^2} - \frac{a_i\mu^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}} - \frac{2b_i\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.107}$$

$$= -\mu^2\left(\frac{1}{2\sigma_\mu^2} + \frac{a_i}{2\sigma_{p|\mathcal{Q}}^2}\right) + \mu\left(\frac{2\mu_\mu}{2\sigma_\mu^2} + \frac{2a_i l}{2\sigma_{p|\mathcal{Q}}^2} - \frac{2b_i}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) \tag{8.108}$$

$$+ \left(\frac{-\mu_\mu^2}{2\sigma_\mu^2} + \frac{-a_i l^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right).$$

Finally, we define

$$\alpha_i = \sqrt{\frac{1}{2\sigma_\mu^2} + \frac{a_i}{2\sigma_{p|\mathcal{Q}}^2}} \tag{8.109}$$

$$\beta_{1,i} = \frac{2\mu_\mu}{2\sigma_\mu^2} + \frac{2a_i l}{2\sigma_{p|\mathcal{Q}}^2} - \frac{2b_i}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.110}$$

$$\beta_{2,i} = \frac{2\mu_\mu}{2\sigma_\mu^2} + \frac{2a_i l}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.111}$$

$$\gamma_{1,i} = \frac{-\mu_\mu^2}{2\sigma_\mu^2} + \frac{-a_i l^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.112}$$

$$\gamma_{2,i} = \frac{-\mu_\mu^2}{2\sigma_\mu^2} + \frac{-a_i l^2}{2\sigma_{p|\mathcal{Q}}^2} + \frac{-2b_i l}{\sigma_{p|\mathcal{Q}}\sqrt{2}} \tag{8.113}$$

We may now approximate the integral $\int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})\text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu$ as

$$\int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q})\text{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu \tag{8.114}$$

$$\approx \int_{-\infty}^{\mu^*} \pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) d\mu - \sum_{i=1}^{4} \frac{c_i}{\sigma_\mu\sqrt{2\pi}} \int_{-\infty}^{\mu^*} \exp\left(\Theta_{1,i}\right) d\mu \quad \text{for } \mu^* < l$$

and

$$\int_{-\infty}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) \mathrm{erf}\left(\frac{l-\mu}{\sigma_{p|\mathcal{Q}}\sqrt{2}}\right) d\mu \tag{8.115}$$
$$\approx \int_{-\infty}^{l} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) d\mu - \sum_{i=1}^{4} \frac{c_i}{\sigma_\mu\sqrt{2\pi}} \int_{-\infty}^{l} \exp\left(\Theta_{1,i}\right) d\mu$$
$$+ \sum_{i=1}^{4} \frac{c_i}{\sigma_\mu\sqrt{2\pi}} \int_{l}^{\mu^*} \exp\left(\Theta_{2,i}\right) d\mu - \int_{l}^{\mu^*} \pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) d\mu \quad \text{for } \mu^* \geq l$$

With $\Theta$ of the form $-\alpha_i^2\mu^2 + \beta_i\mu + \gamma_i$, the solution to the integral is provided in [41] as

$$\int \exp\left(-\alpha_i^2\mu^2 + \beta_i\mu + \gamma_i\right) d\mu = \frac{\sqrt{\pi}}{2\alpha_i} \exp\left(\frac{\beta_i^2}{4\alpha_i^2} + \gamma_i\right) \mathrm{erf}\left(\alpha_i\mu - \frac{\beta_i}{2\alpha_i}\right). \tag{8.116}$$

## 8.7 Useful Identities

**Lemma 8.4** (Matrix Complete the Square)**.**

$$x'Mx - 2y^\top x = (x - M^{-1}y)^\top M(x - M^{-1}y) - y^\top M^{-1}y \tag{8.117}$$

**Lemma 8.5** (Woodbury Matrix Identity)**.** *For matrices $A, C, U, V$ being $n \times n, k \times k, n \times k$, and $k \times n$ respectively,*

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \tag{8.118}$$

### 8.7.1 Marginalization of Product of Gaussians

Using the notation described in Section 8.2.2, we wish to solve for $\pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}})$. For arbitrary sets of input points $\mathcal{A} = \{a_1, \cdots, a_{n_{\mathcal{A}}}\}$ and $\mathcal{B} = \{b_1, \cdots, b_{n_{\mathcal{B}}}\}$ with cardinalities $|\mathcal{A}| = n_{\mathcal{A}}$ and $|\mathcal{B}| = n_{\mathcal{B}}$ respectively. We use the notation

$$m(\mathcal{A}) = \begin{bmatrix} m(a_1) & \cdots & m(a_{n_{\mathcal{A}}}) \end{bmatrix}^\top \tag{8.119}$$

and

$$k(\mathcal{A}, \mathcal{B}) = \begin{bmatrix} k(a_1, b_1) & \cdots & k(a_1, b_{n_{\mathcal{B}}}) \\ \vdots & \ddots & \vdots \\ k(a_{n_{\mathcal{A}}}, b_1) & \cdots & k(a_{n_{\mathcal{A}}}, b_{n_{\mathcal{B}}}) \end{bmatrix}. \tag{8.120}$$

For data sets $\mathcal{C}$ and $\mathcal{D}$ with associated input points $\mathcal{P}_{\mathcal{C}}$ and $\mathcal{P}_{\mathcal{D}}$ respectively, we also use the notation

$$m(\mathcal{P}_{\mathcal{C}}) = m_{\mathcal{C}} \tag{8.121}$$

$$k(\mathcal{P}_{\mathcal{C}}, \mathcal{P}_{\mathcal{D}}) = k_{\mathcal{C},\mathcal{D}} \tag{8.122}$$

Throughout our analysis, we consider the realization $\mu$ of the posterior predictive mean to be a sample from a probability distribution. We assume the probability distribution for $\mu$ to be Dirac delta function centered at the point $\mu_{p|\mathcal{Q}}$ which is a function of $Z_{\mathcal{V}}$. Notably, a Dirac delta function is the limit of a Gaussian distribution as the variance approaches zero. Therefore, for analysis, we assume that the distribution over the posterior predictive mean

$\mu$ is Gaussian with parameters

$$\mu_\mu = m_p + k_{p,\mathcal{Q}}(k_{\mathcal{Q},\mathcal{Q}} + \sigma_n^2 I)^{-1} \begin{bmatrix} Z_\mathcal{S} - m_\mathcal{S} \\ Z_\mathcal{V} - m_\mathcal{V} \end{bmatrix} \tag{8.123}$$

$$\sigma_\mu^2 = \epsilon. \tag{8.124}$$

for an arbitrarily small $\epsilon > 0$ and find the limit of the posterior distribution as $\epsilon$ approaches zero. For convenience, we treat $\pi(\mu|Z_\mathcal{Q})$ as multivariate and let $\Sigma_\mu = \sigma_\mu^2$. Recall that the measurement vector $Z_\mathcal{V}$ is not available in planning, so we wish to find the marginal probability of the posterior predictive mean

$$\pi(\mu|Z_\mathcal{S}, \mathcal{P}_\mathcal{Q}) = \int_{\mathbb{R}^{|\mathcal{V}|}} \pi(Z_\mathcal{V}|\mu_{Z_\mathcal{V}}, \Sigma_{Z_\mathcal{V}})\pi(\mu|\mu_{p|\mathcal{Q}}, \epsilon) dZ_\mathcal{V}. \tag{8.125}$$

Note that we explicitly include the locations $\mathcal{P}_\mathcal{Q}$ of the future measurements in our notation for the marginal probability of the posterior predictive mean. The probability of a future set of measurements $\pi(Z_\mathcal{V}|Z_\mathcal{S})$ is multivariate Gaussian with parameters

$$\mu_{Z_\mathcal{V}} = m_\mathcal{V} + k_{\mathcal{V},\mathcal{S}}(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)^{-1}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.126}$$

$$\Sigma_{Z_\mathcal{V}} = k_{\mathcal{V},\mathcal{V}} - k_{\mathcal{V},\mathcal{S}}(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)^{-1} k_{\mathcal{S},\mathcal{V}} + \sigma_n^2 I \tag{8.127}$$

Note that

$$k_{\mathcal{Q},\mathcal{Q}} = \begin{pmatrix} k_{\mathcal{S},\mathcal{S}} & k_{\mathcal{S},\mathcal{V}} \\ k_{\mathcal{V},\mathcal{S}} & k_{\mathcal{V},\mathcal{V}} \end{pmatrix}. \tag{8.128}$$

We then have

$$k_{\mathcal{Q},\mathcal{Q}} + \sigma_n^2 I = \begin{pmatrix} k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I & k_{\mathcal{S},\mathcal{V}} \\ k_{\mathcal{V},\mathcal{S}} & k_{\mathcal{V},\mathcal{V}} + \sigma_n^2 I \end{pmatrix} \tag{8.129}$$

where the dimensions of the $\sigma_n^2 I$ matrixes should be clear from context. Furthermore, we assume that $k_{\mathcal{S},\mathcal{S}}$ and $k_{\mathcal{V},\mathcal{V}}$ are invertible such that we may apply the property that if $A$ and $B$ are inverible then

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & 0 \\ 0 & (D - CA^{-1}B)^{-1} \end{bmatrix} \begin{bmatrix} I & -BD^{-1} \\ -CA^{-1} & I \end{bmatrix} \tag{8.130}$$

$$= \begin{bmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix}. \tag{8.131}$$

Therefore we have

$$(k_{\mathcal{Q},\mathcal{Q}} + \sigma_n^2 I)^{-1} \begin{bmatrix} Z_{\mathcal{S}} - m_{\mathcal{S}} \\ Z_{\mathcal{V}} - m_{\mathcal{V}} \end{bmatrix} = \Phi_{\mathcal{S}}(Z_{\mathcal{S}} - m_{\mathcal{S}}) + \Phi_{\mathcal{V}}(Z_{\mathcal{V}} - m_{\mathcal{V}}) \tag{8.132}$$

where

$$\Phi_{\mathcal{S}} = \begin{bmatrix} (k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I - k_{\mathcal{S},\mathcal{V}}(k_{\mathcal{V},\mathcal{V}} + \sigma_n^2 I)^{-1} k_{\mathcal{V},\mathcal{S}})^{-1} \\ -(k_{\mathcal{V},\mathcal{V}} + \sigma_n^2 I - k_{\mathcal{V},\mathcal{S}}(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)^{-1} k_{\mathcal{S},\mathcal{V}})^{-1} k_{\mathcal{V},\mathcal{S}}(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)^{-1} \end{bmatrix} \tag{8.133}$$

$$= \begin{bmatrix} \phi_{\mathcal{S},\mathcal{S}} \\ \phi_{\mathcal{S},\mathcal{V}} \end{bmatrix} \tag{8.134}$$

and

$$\Phi_{\mathcal{V}} = \begin{bmatrix} -(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I - k_{\mathcal{S},\mathcal{V}}(k_{\mathcal{V},\mathcal{V}} + \sigma_n^2 I)^{-1} k_{\mathcal{V},\mathcal{S}})^{-1} k_{\mathcal{S},\mathcal{V}}(k_{\mathcal{V},\mathcal{V}} + \sigma_n^2 I)^{-1} \\ (k_{\mathcal{V},\mathcal{V}} + \sigma_n^2 I - k_{\mathcal{V},\mathcal{S}}(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)^{-1} k_{\mathcal{S},\mathcal{V}})^{-1} \end{bmatrix} \tag{8.135}$$

$$= \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} \tag{8.136}$$

We may now rewrite the mean of $\pi(\mu|Z_{\mathcal{S}}, Z_{\mathcal{V}})$ as

$$\mu_\mu = m_p + k_{p,\mathcal{Q}}\Phi_{\mathcal{S}}(Z_{\mathcal{S}} - m_{\mathcal{S}}) + k_{p,\mathcal{Q}}\Phi_{\mathcal{V}}(Z_{\mathcal{V}} - m_{\mathcal{V}}). \tag{8.137}$$

Let $A = \mu - m_p - k_{p,\mathcal{Q}}\Phi_{\mathcal{S}}(Z_{\mathcal{S}} - m_{\mathcal{S}}) + k_{p,\mathcal{Q}}\Phi_{\mathcal{V}}m_{\mathcal{V}}$ and $B = k_{p,\mathcal{Q}}\Phi_{\mathcal{V}}$. Then

$$\pi(\mu|Z_{\mathcal{S}}, Z_{\mathcal{V}}) \propto \exp\left[-\frac{1}{2}(A - BZ_{\mathcal{V}})^\top \Sigma_\mu^{-1}(A - BZ_{\mathcal{V}})\right]. \tag{8.138}$$

We may write $\pi(Z_{\mathcal{V}}|Z_{\mathcal{S}})$ in the same way as

$$\pi(Z_{\mathcal{V}}|Z_{\mathcal{S}}) \propto \exp\left[-\frac{1}{2}(Z_{\mathcal{V}} - \mu_{Z_{\mathcal{V}}})^\top \Sigma_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \mu_{Z_{\mathcal{V}}})\right]. \tag{8.139}$$

Then

$$\begin{aligned} &\pi(\mu|Z_{\mathcal{S}}, Z_{\mathcal{V}})\pi(Z_{\mathcal{V}}|Z_{\mathcal{S}}) \qquad (8.140)\\ &\propto \exp\left[-\frac{1}{2}(A - BZ_{\mathcal{V}})^\top \Sigma_\mu^{-1}(A - BZ_{\mathcal{V}}) - \frac{1}{2}(Z_{\mathcal{V}} - \mu_{Z_{\mathcal{V}}})^\top \Sigma_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \mu_{Z_{\mathcal{V}}})\right] \end{aligned}$$

or

$$\pi(\mu|Z_{\mathcal{S}}, Z_{\mathcal{V}})\pi(Z_{\mathcal{V}}|Z_{\mathcal{S}}) \propto \exp\left[-\frac{1}{2}\Gamma\right]. \tag{8.141}$$

Expanding and simplifying terms in $\Gamma$ gives

$$\Gamma = (A - BZ_{\mathcal{V}})^\top \Sigma_\mu^{-1}(A - BZ_{\mathcal{V}}) + (Z_{\mathcal{V}} - \mu_{Z_{\mathcal{V}}})^\top \Sigma_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \mu_{Z_{\mathcal{V}}}) \tag{8.142}$$

$$= A^\top \Sigma_\mu^{-1} A + Z_{\mathcal{V}}^\top(\Sigma_{Z_{\mathcal{V}}}^{-1} + B^\top \Sigma_\mu^{-1} B)Z_{\mathcal{V}} - 2(A^\top \Sigma_\mu^{-1} B + \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1})Z_{\mathcal{V}} + \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1}\mu_{Z_{\mathcal{V}}} \tag{8.143}$$

Using Lemma 8.4 with

$$x_1 = Z_{\mathcal{V}} \tag{8.144}$$

$$M_1 = \Sigma_{Z_{\mathcal{V}}}^{-1} + B^\top \Sigma_\mu^{-1} B \tag{8.145}$$

$$y_1^\top = A^\top \Sigma_\mu^{-1} B + \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1} \tag{8.146}$$

$$y_1 = B^\top \Sigma_\mu^{-1} A + \Sigma_{Z_{\mathcal{V}}}^{-1} \mu_{Z_{\mathcal{V}}} \tag{8.147}$$

we have

$$\Gamma = A^\top \Sigma_\mu^{-1} A \tag{8.148}$$

$$+ \left(Z_{\mathcal{V}} - M_1^{-1} y_1\right)^\top M_1 \left(Z_{\mathcal{V}} - M_1^{-1} y_1\right) \tag{8.149}$$

$$- (A^\top \Sigma_\mu^{-1} B + \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1})(\Sigma_{Z_{\mathcal{V}}}^{-1} + B^\top \Sigma_\mu^{-1} B)^{-1} (B^\top \Sigma_\mu^{-1} A + \Sigma_{Z_{\mathcal{V}}}^{-1} \mu_{Z_{\mathcal{V}}}) \tag{8.150}$$

$$+ \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1} \mu_{Z_{\mathcal{V}}} \tag{8.151}$$

$$\tag{8.152}$$

which further reduces to

$$\Gamma = A^\top \left(\Sigma_\mu^{-1} - \Sigma_\mu^{-1} B (\Sigma_{Z_{\mathcal{V}}}^{-1} + B^\top \Sigma_\mu^{-1} B)^{-1} B^\top \Sigma_\mu^{-1}\right) A \tag{8.153}$$

$$- 2\mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1} (\Sigma_{Z_{\mathcal{V}}}^{-1} + B^\top \Sigma_\mu^{-1} B)^{-1} B^\top \Sigma_\mu^{-1} A \tag{8.154}$$

$$+ \left(Z_{\mathcal{V}} - M_1^{-1} y_1\right)^\top M_1 \left(Z_{\mathcal{V}} - M_1^{-1} y_1\right) \tag{8.155}$$

$$- \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1} (\Sigma_{Z_{\mathcal{V}}}^{-1} + B^\top \Sigma_\mu^{-1} B)^{-1} \Sigma_{Z_{\mathcal{V}}}^{-1} \mu_{Z_{\mathcal{V}}} \tag{8.156}$$

$$+ \mu_{Z_{\mathcal{V}}}^\top \Sigma_{Z_{\mathcal{V}}}^{-1} \mu_{Z_{\mathcal{V}}}. \tag{8.157}$$

By Lemma 8.5, we further simplify to

$$\Gamma = A^\top \left(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}} B^\top\right)^{-1} A \tag{8.158}$$

$$- 2\mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}(\Sigma_{Z_\mathcal{V}} - \Sigma_{Z_\mathcal{V}} B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}} B^\top)^{-1} B\Sigma_{Z_\mathcal{V}})B^\top \Sigma_\mu^{-1} A \tag{8.159}$$

$$+ \left(Z_\mathcal{V} - M_1^{-1} y_1\right)^\top M_1 \left(Z_\mathcal{V} - M_1^{-1} y_1\right) \tag{8.160}$$

$$- \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}(\Sigma_{Z_\mathcal{V}}^{-1} + B^\top \Sigma_\mu^{-1} B)^{-1} \Sigma_{Z_\mathcal{V}}^{-1} \mu_{Z_\mathcal{V}} \tag{8.161}$$

$$+ \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1} \mu_{Z_\mathcal{V}}. \tag{8.162}$$

Using Lemma 8.4 with

$$x_2 = A \tag{8.163}$$

$$M_2 = \left(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}} B^\top\right)^{-1} \tag{8.164}$$

$$y_2^\top = \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}(\Sigma_{Z_\mathcal{V}} - \Sigma_{Z_\mathcal{V}} B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}} B^\top)^{-1} B\Sigma_{Z_\mathcal{V}})B^\top \Sigma_\mu^{-1} \tag{8.165}$$

$$y_2 = \Sigma_\mu^{-1} B(\Sigma_{Z_\mathcal{V}} - \Sigma_{Z_\mathcal{V}} B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}} B^\top)^{-1} B\Sigma_{Z_\mathcal{V}})\Sigma_{Z_\mathcal{V}}^{-1} \mu_{Z_\mathcal{V}} \tag{8.166}$$

yields

$$\Gamma = (A - M_2^{-1} y_2)^\top M_2 (A - M_2^{-1} y_2) - y_2^\top M_2^{-1} y_2 \tag{8.167}$$

$$+ \left(Z_\mathcal{V} - M_1^{-1} y_1\right)^\top M_1 \left(Z_\mathcal{V} - M_1^{-1} y_1\right) \tag{8.168}$$

$$- \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}(\Sigma_{Z_\mathcal{V}}^{-1} + B^\top \Sigma_\mu^{-1} B)^{-1} \Sigma_{Z_\mathcal{V}}^{-1} \mu_{Z_\mathcal{V}} \tag{8.169}$$

$$+ \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1} \mu_{Z_\mathcal{V}}. \tag{8.170}$$

Note that $M_2^{-1}y_2$ may be reduced to

$$M_2^{-1}y_2 = (\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)\Sigma_\mu^{-1}B(\Sigma_{Z_\mathcal{V}} - \Sigma_{Z_\mathcal{V}}B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)^{-1}B\Sigma_{Z_\mathcal{V}})\Sigma_{Z_\mathcal{V}}^{-1}\mu_{Z_\mathcal{V}} \tag{8.171}$$

$$= (\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)\Sigma_\mu^{-1}B(I - \Sigma_{Z_\mathcal{V}}B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)^{-1}B)\mu_{Z_\mathcal{V}} \tag{8.172}$$

$$= (\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)\Sigma_\mu^{-1}B\mu_{Z_\mathcal{V}} - (\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)\Sigma_\mu^{-1}B\Sigma_{Z_\mathcal{V}}B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)^{-1}B\mu_{Z_\mathcal{V}}. \tag{8.173}$$

Because $p$ is a single point $B\Sigma_{Z_\mathcal{V}}B^\top\Sigma_\mu^{-1}$ evaluates to a scalar quantity. Notably for any scalar $\alpha$ and matrices $\Lambda_1$ and $\Lambda_2$ we have the properties that

$$\alpha = \alpha^\top \tag{8.174}$$

$$\Lambda_1\alpha\Lambda_2 = \alpha\Lambda_1\Lambda_2. \tag{8.175}$$

Therefore

$$M_2^{-1}y_2 = B\mu_{Z_\mathcal{V}} + B\Sigma_{Z_\mathcal{V}}B^\top\Sigma_\mu^{-1}B\mu_{Z_\mathcal{V}} \tag{8.176}$$

$$- B\Sigma_{Z_\mathcal{V}}B^\top\Sigma_\mu^{-1}(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)^{-1}B\mu_{Z_\mathcal{V}}$$

$$= B\mu_{Z_\mathcal{V}} + B\Sigma_{Z_\mathcal{V}}B^\top\Sigma_\mu^{-1}B\mu_{Z_\mathcal{V}} - B\Sigma_{Z_\mathcal{V}}B^\top\Sigma_\mu^{-1}B\mu_{Z_\mathcal{V}} \tag{8.177}$$

$$= B\mu_{Z_\mathcal{V}}. \tag{8.178}$$

Replacing $M_2^{-1}y_2$ with $B\mu_{Z_\mathcal{V}}$, we are left with

$$\Gamma = (A - B\mu_{Z_\mathcal{V}})^\top \left(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top\right)^{-1}(A - B\mu_{Z_\mathcal{V}}) \tag{8.179}$$

$$- (\mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}(\Sigma_{Z_\mathcal{V}} - \Sigma_{Z_\mathcal{V}}B^\top(\Sigma_\mu + B\Sigma_{Z_\mathcal{V}}B^\top)^{-1}B\Sigma_{Z_\mathcal{V}})B^\top\Sigma_\mu^{-1})B\mu_{Z_\mathcal{V}} \tag{8.180}$$

$$+ \left(Z_\mathcal{V} - M_1^{-1}y_1\right)^\top M_1 \left(Z_\mathcal{V} - M_1^{-1}y_1\right) \tag{8.181}$$

$$- \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}(\Sigma_{Z_\mathcal{V}}^{-1} + B^\top\Sigma_\mu^{-1}B)^{-1}\Sigma_{Z_\mathcal{V}}^{-1}\mu_{Z_\mathcal{V}} \tag{8.182}$$

$$+ \mu_{Z_\mathcal{V}}^\top \Sigma_{Z_\mathcal{V}}^{-1}\mu_{Z_\mathcal{V}}. \tag{8.183}$$

We recognise that the terms on lines 8.179 and 8.181 have the form of Gaussian distributions over $A$ and $Z_\mathcal{V}$ respectively. We also note that all other terms contain neither $\mu$ nor $Z_\mathcal{V}$ and may therefore be ignored as they simply represent constant scaling factors for the final product distribution. Therefore, recalling that $A = \mu - m_p - k_{p,\mathcal{Q}}\Phi_\mathcal{S}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{Q}}\Phi_\mathcal{V}m_\mathcal{V}$ and $B = k_{p,\mathcal{Q}}\Phi_\mathcal{V}$, we define

$$\tilde{\mu}_\mu = m_p + k_{p,\mathcal{Q}}\Phi_\mathcal{S}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{Q}}\Phi_\mathcal{V}(\mu_{Z_\mathcal{V}} - m_\mathcal{V}) \tag{8.184}$$

$$\tilde{\Sigma}_\mu = \Sigma_\mu + k_{p,\mathcal{Q}}\Phi_\mathcal{V}\Sigma_{Z_\mathcal{V}}\Phi_\mathcal{V}^\top k_{\mathcal{Q},p} \tag{8.185}$$

and

$$\tilde{\mu}_{Z_\mathcal{V}} = (\Sigma_{Z_\mathcal{V}}^{-1} + \Phi_\mathcal{V}^\top k_{\mathcal{Q},p}\Sigma_\mu^{-1}k_{p,\mathcal{Q}}\Phi_\mathcal{V})^{-1}(\Phi_\mathcal{V}^\top k_{\mathcal{Q},p}\Sigma_\mu^{-1}A + \Sigma_{Z_\mathcal{V}}^{-1}\mu_{Z_\mathcal{V}}) \tag{8.186}$$

$$\tilde{\Sigma}_{Z_\mathcal{V}} = \Sigma_{Z_\mathcal{V}}^{-1} + \Phi_\mathcal{V}^\top k_{\mathcal{Q},p}\Sigma_\mu^{-1}k_{p,\mathcal{Q}}\Phi_\mathcal{V} \tag{8.187}$$

which gives

$$\pi(\mu|Z_{\mathcal{S}}, Z_{\mathcal{V}})\pi(Z_{\mathcal{V}}|Z_{\mathcal{S}}) \tag{8.188}$$
$$\propto \exp(-\frac{1}{2}(\mu - \tilde{\mu}_\mu)^\top \tilde{\Sigma}_\mu^{-1}(\mu - \tilde{\mu}_\mu)) \exp(-\frac{1}{2}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}})^\top \tilde{\Sigma}_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}})).$$

Then we have

$$\int \pi(\mu|Z_{\mathcal{S}}, Z_{\mathcal{V}})\pi(Z_{\mathcal{V}}|Z_{\mathcal{S}})dZ_{\mathcal{V}} \tag{8.189}$$
$$\propto \int [\exp(-\frac{1}{2}(\mu - \tilde{\mu}_\mu)^\top \tilde{\Sigma}_\mu^{-1}(\mu - \tilde{\mu}_\mu)) \exp(-\frac{1}{2}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}})^\top \tilde{\Sigma}_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}}))]dZ_{\mathcal{V}}$$
$$= \exp(-\frac{1}{2}(\mu - \tilde{\mu}_\mu)^\top \tilde{\Sigma}_\mu^{-1}(\mu - \tilde{\mu}_\mu)) \int \left[\exp(-\frac{1}{2}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}})^\top \tilde{\Sigma}_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}}))\right] dZ_{\mathcal{V}} \tag{8.190}$$

where it is well known that

$$\int \left[\exp(-\frac{1}{2}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}})^\top \tilde{\Sigma}_{Z_{\mathcal{V}}}^{-1}(Z_{\mathcal{V}} - \tilde{\mu}_{Z_{\mathcal{V}}}))]dZ_{\mathcal{V}}\right] dZ_{\mathcal{V}} = \det(2\pi\tilde{\Sigma}_{Z_{\mathcal{V}}})^{\frac{1}{2}}. \tag{8.191}$$

It follows that

$$\pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) \propto \exp(-\frac{1}{2}(\mu - \tilde{\mu}_\mu)^\top \tilde{\Sigma}_\mu^{-1}(\mu - \tilde{\mu}_\mu)). \tag{8.192}$$

Thus, we conclude that

$$\pi(\mu|Z_{\mathcal{S}}, \mathcal{P}_{\mathcal{Q}}) = \mathcal{N}(\tilde{\mu}_\mu, \tilde{\Sigma}_\mu). \tag{8.193}$$

Defining $\Sigma_{\mathcal{S}} = (k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)$, the parameters $\tilde{\mu}_\mu$ and $\tilde{\Sigma}_\mu$ may be simplified as follows:

1. **Mean:**

$$\tilde{\mu}_\mu = m_p + k_{p,\mathcal{Q}}\Phi_\mathcal{S}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{Q}}\Phi_\mathcal{V}(\mu_{Z_\mathcal{V}} - m_\mathcal{V}) \tag{8.194}$$

$$= m_p + k_{p,\mathcal{S}}\phi_{\mathcal{S},\mathcal{S}}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) + (k_{p,\mathcal{S}}\phi_{\mathcal{V},\mathcal{S}} \tag{8.195}$$

$$+ k_{p,\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}})(\mu_{Z_\mathcal{V}} - m_\mathcal{V})$$

$$= m_p + k_{p,\mathcal{S}}(\Sigma_\mathcal{S}^{-1} - \Sigma_\mathcal{S}^{-1}k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}})(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.196}$$

$$+ (k_{p,\mathcal{S}}\phi_{\mathcal{V},\mathcal{S}} + k_{p,\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}})k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.197}$$

$$= m_p + k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) - k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.198}$$

$$+ (k_{p,\mathcal{S}}\phi_{\mathcal{V},\mathcal{S}} + k_{p,\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}})k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.199}$$

$$= m_p + k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) - k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.200}$$

$$+ k_{p,\mathcal{S}}\phi_{\mathcal{V},\mathcal{S}}k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}}k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.201}$$

Note that $\Sigma_\mathcal{S}^{-1}k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}} = \phi_{\mathcal{V},\mathcal{S}}k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1}$ such that

$$\tilde{\mu}_\mu = m_p + k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}}k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}). \tag{8.202}$$

Also, $\phi_{\mathcal{V},\mathcal{V}}k_{\mathcal{V},\mathcal{S}}\Sigma_\mathcal{S}^{-1} = -\phi_{\mathcal{S},\mathcal{V}}$ yielding

$$\tilde{\mu}_\mu = m_p + k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) + k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) - k_{p,\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.203}$$

$$= m_p + k_{p,\mathcal{S}}\Sigma_\mathcal{S}^{-1}(Z_\mathcal{S} - m_\mathcal{S}) \tag{8.204}$$

$$= \mu_{p|\mathcal{S}}. \tag{8.205}$$

2. **Variance:** From (8.185),

$$\tilde{\Sigma}_\mu = \Sigma_\mu + k_{p,\mathcal{Q}} \Phi_\mathcal{V} \Sigma_{Z_\mathcal{V}} \Phi_\mathcal{V}^\top k_{\mathcal{Q},p}. \tag{8.206}$$

Adding $\sigma^2_{p|\mathcal{Q}}$ to each side gives

$$\tilde{\Sigma}_\mu + \sigma^2_{p|\mathcal{Q}} = \Sigma_\mu + k_{p,p} + k_{p,\mathcal{Q}} \begin{bmatrix} \phi_{\mathcal{S},\mathcal{S}} & \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{S},\mathcal{V}} & \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} k_{\mathcal{Q},p} + k_{p,\mathcal{Q}} \Phi_\mathcal{V} \Sigma_{Z_\mathcal{V}} \Phi_\mathcal{V}^\top k_{\mathcal{Q},p}. \tag{8.207}$$

Note that $\phi_{\mathcal{S},\mathcal{V}} = \phi_{\mathcal{V},\mathcal{S}}^\top$. To convince yourself of this, just look at the block matrix inverse identities and note that $C = B^\top$. Therefore,

$$\begin{aligned}
\Phi_\mathcal{V} \Sigma_{Z_\mathcal{V}} \Phi_\mathcal{V}^\top &= \Phi_\mathcal{V}(k_{\mathcal{V},\mathcal{V}} - k_{\mathcal{V},\mathcal{S}}(k_{\mathcal{S},\mathcal{S}} + \sigma_n^2 I)^{-1} k_{\mathcal{S},\mathcal{V}} + \sigma_n^2 I)\Phi_\mathcal{V}^\top && (8.208)\\
&= \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} \phi_{\mathcal{V},\mathcal{V}}^{-1} \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}}^\top & \phi_{\mathcal{V},\mathcal{V}}^\top \end{bmatrix} && (8.209)\\
&= \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}} \phi_{\mathcal{V},\mathcal{V}}^{-1} \phi_{\mathcal{V},\mathcal{S}}^\top & \phi_{\mathcal{V},\mathcal{S}} \phi_{\mathcal{V},\mathcal{V}}^{-1} \phi_{\mathcal{V},\mathcal{V}} \\ \phi_{\mathcal{V},\mathcal{V}} \phi_{\mathcal{V},\mathcal{V}}^{-1} \phi_{\mathcal{V},\mathcal{S}}^\top & \phi_{\mathcal{V},\mathcal{V}} \phi_{\mathcal{V},\mathcal{V}}^{-1} \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} && (8.210)\\
&= \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}} \phi_{\mathcal{V},\mathcal{V}}^{-1} \phi_{\mathcal{V},\mathcal{S}}^\top & \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{V},\mathcal{S}}^\top & \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} && (8.211)\\
&= \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}} \phi_{\mathcal{V},\mathcal{V}}^{-1} \phi_{\mathcal{S},\mathcal{V}} & \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{S},\mathcal{V}} & \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix}. && (8.212)
\end{aligned}$$

It follows that

$$\tilde{\Sigma}_\mu + \sigma^2_{p|\mathcal{Q}} = \Sigma_\mu + k_{p,p} - k_{p,\mathcal{Q}} \left( \begin{bmatrix} \phi_{\mathcal{S},\mathcal{S}} & \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{S},\mathcal{V}} & \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} - \begin{bmatrix} \phi_{\mathcal{V},\mathcal{S}}\phi^{-1}_{\mathcal{V},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}} & \phi_{\mathcal{V},\mathcal{S}} \\ \phi_{\mathcal{S},\mathcal{V}} & \phi_{\mathcal{V},\mathcal{V}} \end{bmatrix} \right) k_{\mathcal{Q},p} \tag{8.213}$$

$$= \Sigma_\mu + k_{p,p} - k_{p,\mathcal{Q}} \left( \begin{bmatrix} \phi_{\mathcal{S},\mathcal{S}} k_{\mathcal{S},p} - \phi_{\mathcal{V},\mathcal{S}}\phi^{-1}_{\mathcal{V},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}} \\ 0 \end{bmatrix} \right) \tag{8.214}$$

$$= \Sigma_\mu + k_{p,p} - k_{p,\mathcal{S}}(\phi_{\mathcal{S},\mathcal{S}} k_{\mathcal{S},p} - \phi_{\mathcal{V},\mathcal{S}}\phi^{-1}_{\mathcal{V},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}} k_{\mathcal{S},p}) \tag{8.215}$$

$$= \Sigma_\mu + k_{p,p} - k_{p,\mathcal{S}}(\phi_{\mathcal{S},\mathcal{S}} k_{\mathcal{S},p} + \Sigma^{-1}_{\mathcal{S}} k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{S},\mathcal{V}} k_{\mathcal{S},p}) \tag{8.216}$$

$$= \Sigma_\mu + k_{p,p} \tag{8.217}$$

$$- k_{p,\mathcal{S}}(\Sigma^{-1}_{\mathcal{S}} + \Sigma^{-1}_{\mathcal{S}} k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}} k_{\mathcal{V},\mathcal{S}}\Sigma^{-1}_{\mathcal{S}} - \Sigma^{-1}_{\mathcal{S}} k_{\mathcal{S},\mathcal{V}}\phi_{\mathcal{V},\mathcal{V}} k_{\mathcal{V},\mathcal{S}}\Sigma^{-1}_{\mathcal{S}}) k_{\mathcal{S},p}$$

$$= \Sigma_\mu + k_{p,p} - k_{p,\mathcal{S}}\Sigma^{-1}_{\mathcal{S}} k_{\mathcal{S},p} \tag{8.218}$$

$$= \Sigma_\mu + \sigma^2_{p|\mathcal{S}}. \tag{8.219}$$

Subtracting $\sigma^2_{p|\mathcal{Q}}$ from both sides and replacing $\Sigma_\mu$ with $\epsilon$ gives the final result

$$\tilde{\Sigma}_\mu = \sigma^2_{p|\mathcal{S}} - \sigma^2_{p|\mathcal{Q}} + \epsilon. \tag{8.220}$$

In the limit as $\epsilon$ approaches zero,

$$\lim_{\epsilon \to 0} \tilde{\Sigma}_\mu = \sigma^2_{p|\mathcal{S}} - \sigma^2_{p|\mathcal{Q}}. \tag{8.221}$$

# Bibliography


[1] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J Pappas. Information acquisition with sensing robots: Algorithms and error bounds. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6447–6454. IEEE, 2014.

[2] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J Pappas. Decentralized active information acquisition: Theory and application to multi-robot slam. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4775–4782. IEEE, 2015.

[3] Wenruo Bai and Jeff Bilmes. Greed is still good: maximizing monotone submodular+ supermodular (bp) functions. In *International Conference on Machine Learning*, pages 304–313. PMLR, 2018.

[4] J. Bellingham, A. Richards, and J.P. How. Receding horizon control of autonomous aerial vehicles. In *Proc. American Control Conference*, pages 3741–3746, 2002. ISBN 978-0-7803-7298-6. doi: 10.1109/ACC.2002.1024509.

[5] Graeme Best, Oliver M Cliff, Timothy Patten, Ramgopal R Mettu, and Robert Fitch. Dec-mcts: Decentralized planning for multi-robot active perception. *The International Journal of Robotics Research*, 38(2-3):316–337, 2019.

[6] Andrew An Bian, Joachim M Buhmann, Andreas Krause, and Sebastian Tschiatschek. Guarantees for greedy maximization of non-submodular functions with applications. In *International conference on machine learning*, pages 498–507. PMLR, 2017.

[7] Benjamin Biggs, Daniel J. Stilwell, Harun Yetkin, and James McMahon. Performance

guarantees for receding horizon search with terminal cost. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (iROS)*. IEEE, 2019.

[8] Ilija Bogunovic, Junyao Zhao, and Volkan Cevher. Robust maximization of non-submodular objectives. In *International Conference on Artificial Intelligence and Statistics*, pages 890–899. PMLR, 2018.

[9] Martin Brossard, Axel Barrau, and Silvère Bonnabel. A code for unscented kalman filtering on manifolds (UKF-M). In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5701–5708, 2020. doi: 10.1109/ICRA40945.2020.9197489.

[10] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[11] Brent Bryan, Robert C Nichol, Christopher R Genovese, Jeff Schneider, Christopher J Miller, and Larry Wasserman. Active learning for identifying function threshold boundaries. *Advances in neural information processing systems*, 18, 2005.

[12] Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[13] Richard Camilli, Brian Bingham, Michael Jakuba, Hanumant Singh, and Jean Whelan. Integrating in-situ chemical sampling with auv control systems. In *Oceans' 04 MTS/IEEE Techno-Ocean'04 (IEEE Cat. No. 04CH37600)*, volume 1, pages 101–109. IEEE, 2004.

[14] Lin Chen, Moran Feldman, and Amin Karbasi. Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy? In *International Conference on Machine Learning*, pages 804–813. PMLR, 2018.

[15] Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete applied mathematics*, 7(3):251–274, 1984.

[16] Micah Corah and Nathan Michael. Efficient online multi-robot exploration via distributed sequential greedy assignment. In *Robotics: Science and Systems*, volume 13, 2017.

[17] Micah Corah and Nathan Michael. Distributed submodular maximization on partition matroids for planning on large sensor networks. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6792–6799. IEEE, 2018.

[18] Micah Corah and Nathan Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice. *Autonomous Robots*, 43(2): 485–501, 2019.

[19] G. de Nicolao, L. Magni, and R. Scattolini. On the robustness of receding-horizon control with terminal constraints. *IEEE Transactions on Automatic Control*, 41(3): 451–453, 1996. doi: 10.1109/9.486649.

[20] Xuchu Ding, Mircea Lazar, and Calin Belta. LTL receding horizon control for finite deterministic systems. *Automatica*, 50(2):399–408, 2014. ISSN 00051098. doi: 10.1016/j.automatica.2013.11.030.

[21] Matthew Dunbabin and Alistair Grinham. Quantifying spatiotemporal greenhouse gas

emissions using autonomous surface vehicles. *Journal of field robotics*, 34(1):151–169, 2017.

[22] Jack Edmonds. Matroids and the greedy algorithm. *Mathematical programming*, 1(1): 127–136, 1971.

[23] Marshall L Fisher, George L Nemhauser, and Laurence A Wolsey. An analysis of approximations for maximizing submodular set functions—ii. In *Polyhedral combinatorics*, pages 73–87. Springer, 1978.

[24] Tobias Friedrich, Andreas Göbel, Frank Neumann, Francesco Quinzan, and Ralf Rothenberger. Greedy maximization of functions with bounded curvature under partition matroid constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2272–2279, 2019.

[25] Sergey Frolov, Bartolame Garau, and James Bellingham. Can we do better than the grid survey: Optimal synoptic surveys in presence of variable uncertainty and decorrelation scales. *Journal of Geophysical Research: Oceans*, 119(8):5071–5090, 2014.

[26] Sergey Frolov, Bartolame Garau, and James Bellingham. Can we do better than the grid survey: Optimal synoptic surveys in presence of variable uncertainty and decorrelation scales. *Journal of Geophysical Research: Oceans*, 119(8):5071–5090, 2014. ISSN 21699275. doi: 10.1002/2013JC009521.

[27] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013.

[28] Khashayar Gatmiry and Manuel Gomez-Rodriguez. The network visibility problem. *ACM Transactions on Information Systems (TOIS)*, 40(2):1–42, 2021.

[29] Bahman Gharesifard and Stephen L Smith. Distributed submodular maximization with limited information. *IEEE transactions on control of network systems*, 5(4):1635–1645, 2017.

[30] Alkis Gotovos. Active learning for level set estimation. Master's thesis, Eidgenössische Technische Hochschule Zürich, Department of Computer Science,, 2013.

[31] David Grimsman, Mohd Shabbir Ali, Joao P Hespanha, and Jason R Marden. The impact of information in distributed submodular maximization. *IEEE Transactions on Control of Network Systems*, 6(4):1334–1343, 2018.

[32] M. P. Hayes and P. T. Gough. Synthetic aperture sonar: A review of current status. *IEEE Journal of Oceanic Engineering*, 34(3):207–224, July 2009. ISSN 0364-9059. doi: 10.1109/JOE.2009.2020853.

[33] Gregory Hitz, Alkis Gotovos, Marie-Éve Garneau, Cédric Pradalier, Andreas Krause, Roland Y Siegwart, et al. Fully autonomous focused exploration for robotic environmental monitoring. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2658–2664. IEEE, 2014.

[34] Geoffrey A Hollinger and Gaurav S Sukhatme. Sampling-based robotic information gathering algorithms. *The International Journal of Robotics Research*, 33(9):1271–1287, 2014.

[35] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 2013. ISBN 9780521839402.

[36] Phillip Hyatt, Zachary Brock, and Marc D Killpack. A versatile multi-robot monte carlo tree search planner for on-line coverage path planning. *arXiv preprint arXiv:2002.04517*, 2020.

[37] Ali Jadbabaie and John Hauser. On the stability of receding horizon control with a general terminal cost. *IEEE Transactions on Automatic Control*, 50(5):674–678, 2005.

[38] Yiannis Kantaros, Brent Schlotfeldt, Nikolay Atanasov, and George J Pappas. Asymptotically optimal planning for non-myopic multi-robot information gathering. *Proceedings of Robotics: Science and Systems (RSS), Freiburg, Germany*, pages 22–26, 2019.

[39] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

[40] Levente Kocsis, Csaba Szepesvári, and Jan Willemson. Improved monte-carlo search. *Univ. Tartu, Estonia, Tech. Rep*, 1, 2006.

[41] Nikolai E Korotkov and Alexander N Korotkov. *Integrals related to the error function*. Chapman and Hall/CRC, 2020.

[42] Andreas Krause. *Optimizing sensing: Theory and applications*. Ph.D dissertation, Carnegie Mellon University, 2008.

[43] Andreas Krause and Carlos E Guestrin. Near-optimal nonmyopic value of information in graphical models. *arXiv preprint arXiv:1207.1394*, 2012.

[44] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008.

[45] Stephen T Krauss and Daniel J Stilwell. Unscented kalman filtering on manifolds for auv navigation-experimental results. In *OCEANS 2022, Hampton Roads*, pages 1–6. IEEE, 2022.

[46] Yoshiaki Kuwata and Jonathan How. Three dimensional receding horizon control for UAVs. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 5144, 2004.

[47] Jay H Lee. Model predictive control: Review of the three decades of development. *International Journal of Control, Automation and Systems*, 9(3):415, 2011.

[48] Su Liu and Jinfeng Liu. Economic model predictive control with extended horizon. *Automatica*, 73:180–192, 2016. ISSN 00051098. doi: 10.1016/j.automatica.2016.06.027.

[49] Cheng Lu, Wenguo Yang, Ruiqi Yang, and Suixiang Gao. Maximizing a non-decreasing non-submodular function subject to various types of constraints. *Journal of Global Optimization*, pages 1–25, 2022.

[50] David Q Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

[51] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6): 789–814, 2000.

[52] Seth McCammon, Gilberto Marcon dos Santos, Matthew Frantz, TP Welch, Graeme Best, R Kipp Shearman, Jonathan D Nash, John A Barth, Julie A Adams, and Geoffrey A Hollinger. Ocean front detection and tracking using a team of heterogeneous marine vehicles. *Journal of Field Robotics*, 38(6):854–881, 2021.

[53] James McMahon, Harun Yetkin, Artur Wolek, Zachary J Waters, and Daniel J Stilwell. Towards real-time search planning in subsea environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 87–94. IEEE, 2017.

[54] Bernard Mettler and Olivier Toupet. Receding horizon trajectory planning with an environment-based cost-to-go function. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 4071–4076. IEEE, 2005.

[55] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

[56] Edward W Ng and Murray Geller. A table of integrals of the error functions. *Journal of Research of the National Bureau of Standards B*, 73(1):1–20, 1969.

[57] Qingqin Nong, Tao Sun, Suning Gong, Qizhi Fang, Dingzhu Du, and Xiaoyu Shao. Maximize a monotone function with a generic submodularity ratio. In *International Conference on Algorithmic Applications in Management*, pages 249–260. Springer, 2019.

[58] James A Primbs and Vesna Nevistić. Feasibility and stability of constrained finite receding horizon control. *Automatica*, 36(7):965 – 971, 2000. doi: 10.1016/S0005-1098(00)00004-2.

[59] Graham Pulford. A 4-term exponential-quadratic approximation for gaussian q or error functions accurate to $1.65 \times 10^{-4}$. *REV Journal on Electronics and Communications*, 2022.

[60] Mohammed Rahimi, Mark Hansen, William J Kaiser, Gaurav S Sukhatme, and Deborah Estrin. Adaptive sampling for environmental field estimation using robotic sensors. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3692–3698. IEEE, 2005.

[61] Richard Santiago and Yuichi Yoshida. Weakly submodular function maximization using local submodularity ratio. *arXiv preprint arXiv:2004.14650*, 2020.

[62] Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J Pappas. Anytime planning for decentralized multirobot active information gathering. *IEEE Robotics and Automation Letters*, 3(2):1025–1032, 2018.

[63] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

[64] Amarjeet Singh, Andreas Krause, Carlos Guestrin, and William J Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

[65] Amarjeet Singh, Andreas Krause, and William J Kaiser. Nonmyopic adaptive informative path planning for multiple robots. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 2009.

[66] Andrés Solé, Vicent Caselles, Guillermo Sapiro, and Francesc Aràndiga. Morse description and geometric encoding of digital elevation maps. *IEEE Transactions on Image Processing*, 13(9):1245–1262, 2004.

[67] Fouad Sukkar, Graeme Best, Chanyeol Yoo, and Robert Fitch. Multi-robot region-of-interest reconstruction with dec-mcts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9101–9107. IEEE, 2019.

[68] Robert Szewczyk, Eric Osterweil, Joseph Polastre, Michael Hamilton, Alan Mainwaring, and Deborah Estrin. Habitat monitoring with sensor networks. *Communications of the ACM*, 47(6):34–40, 2004.

[69] Michael P Vitus, Wei Zhang, Alessandro Abate, Jianghai Hu, and Claire J Tomlin. On efficient sensor scheduling for linear dynamical systems. *Automatica*, 48(10):2482–2493, 2012.

[70] Jennifer Wakulicz, Ki Myung Brian Lee, Chanyeol Yoo, Teresa Vidal-Calleja, and Robert Fitch. Informative planning for worst-case error minimisation in sparse gaussian process regression. *arXiv preprint arXiv:2203.03828*, 2022.

[71] R. M. Willett and R. D. Nowak. Minimax optimal level-set estimation. *IEEE Transactions on Image Processing*, 16(12):2965–2979, 2007. doi: 10.1109/TIP.2007.910175.

[72] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[73] Bin Xu, Andrew Kurdila, and Daniel J. Stilwell. A hybrid receding horizon control method for path planning in uncertain environments. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 4887–4892, 2009. ISBN 978-1-4244-3803-7. doi: 10.1109/IROS.2009.5354235.

[74] Yee Hwa Yang, Michael J Buckley, Sandrine Dudoit, and Terence P Speed. Comparison of methods for image analysis on cdna microarray data. *Journal of computational and graphical statistics*, 11(1):108–136, 2002.

[75] Harun Yetkin, Collin Lutz, and Daniel Stilwell. Acquiring environmental information yields better anticipated search performance. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–6. IEEE, 2016.

[76] Harun Yetkiny, James McMahon, Nicholay Topin, Artur Wolek, Zachary Waters, and Daniel J Stilwell. Online planning for autonomous underwater vehicles performing information gathering tasks in large subsea environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6354–6361. IEEE, 2019.

[77] Soo-Hyun Yoo, Andrew Stuntz, Yawei Zhang, Robert Rothschild, Geoffrey A. Hollinger, and Ryan N. Smith. Experimental analysis of receding horizon planning algorithms

for marine monitoring. In *Proc. Tenth International Conference on Field and Service Robotics (FSR)*, volume 113, pages 31 – 44, 2016.

[78] Yanwu Zhang, James G Bellingham, John P Ryan, Brian Kieft, and M Jordan Stanway. Autonomous four-dimensional mapping and tracking of a coastal upwelling front by an autonomous underwater vehicle. *Journal of Field Robotics*, 33(1):67–81, 2016.