NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ACTIVE TERRAIN-AIDED NAVIGATION

by

Darren Kurt

March 2020

Thesis Advisor: Douglas P. Horner
Co-Advisor: Monique P. Fargues

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2020	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE ACTIVE TERRAIN-AIDED NAVIGATION	**5. FUNDING NUMBERS**
6. AUTHOR(S) Darren Kurt	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research (ONR), Monterey, CA 93940	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Navigation without external beacon systems has been a long-standing goal within the autonomous systems community. Terrain-aided navigation (TAN) presents an alternative to external localization but requires a prior map. In many cases, the unmanned vehicle (UV) is surveying areas where no map exists and therefore TAN cannot be employed. Active terrain-aided navigation (ATAN) incorporates reinforcement learning (RL) into TAN to reduce the dependence on a prior map. The spatial and temporal measurement uncertainties create the classical problem of exploration versus exploitation: the desire to explore all map areas while exploiting known areas to minimize position error. A dual stochastic optimal estimation problem models the exploration-exploitation dilemma through an information theoretic framework (ITF) that maximizes information gain in real time: a challenge with high computational complexity. The computational cost is reduced using an RL technique called a partially observable Monte Carlo planning process (POMCP). ATAN employs the ITF and POMCP to provide a better, more flexible coverage plan for TAN. The results of this thesis demonstrate the ability of a UV to navigate autonomously without a prior map while minimizing position error, ensuring total coverage, and creating an accurate map within time/energy constraints. Through ATAN, greater levels of autonomy are exhibited, improving upon the TAN framework but also providing a larger offering to the field of robotics.

14. SUBJECT TERMS machine learning, artificial intelligence, reinforcement learning, autonomous systems, robotics, POMCP, information theoretic framework, terrain-aided navigation, TAN, active terrain-aided navigation, ATAN, UTAN, unmanned vehicles, exploration, exploitation	**15. NUMBER OF PAGES** 175
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ACTIVE TERRAIN-AIDED NAVIGATION

Darren Kurt
Lieutenant, United States Navy
BS, University of California, Berkeley, 2012

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 2020

Approved by: Douglas P. Horner
Advisor

Monique P. Fargues
Co-Advisor

Douglas J. Fouts
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Navigation without external beacon systems has been a long-standing goal within the autonomous systems community. Terrain-aided navigation (TAN) presents an alternative to external localization but requires a prior map. In many cases, the unmanned vehicle (UV) is surveying areas where no map exists and therefore TAN cannot be employed. Active terrain-aided navigation (ATAN) incorporates reinforcement learning (RL) into TAN to reduce the dependence on a prior map. The spatial and temporal measurement uncertainties create the classical problem of exploration versus exploitation: the desire to explore all map areas while exploiting known areas to minimize position error. A dual stochastic optimal estimation problem models the exploration-exploitation dilemma through an information theoretic framework (ITF) that maximizes information gain in real time: a challenge with high computational complexity. The computational cost is reduced using an RL technique called a partially observable Monte Carlo planning process (POMCP). ATAN employs the ITF and POMCP to provide a better, more flexible coverage plan for TAN. The results of this thesis demonstrate the ability of a UV to navigate autonomously without a prior map while minimizing position error, ensuring total coverage, and creating an accurate map within time/energy constraints. Through ATAN, greater levels of autonomy are exhibited, improving upon the TAN framework but also providing a larger offering to the field of robotics.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

ADCP Acoustic Doppler Current Profiler

ATAN Active Terrain-Aided Navigation

AUV Autonomous Underwater Vehicle

BI Box In Mapping Methodology

CAVR Center for Autonomous Vehicle Research

CDF Cumulative Distribution Function

CI Confidence Interval

CSUMB California State University, Monterey Bay

ESS Effective Sampling Size

GP Gaussian Process

GPM Gaussian Process Model

GPS Global Positioning System

IID Independent, Identically Distributed

INS Inertial Navigation System

ITF Information theoretic framework

KF Kalman Filter

KLD Kullback Leibler Divergence

MAP Maximum *A Posteriori*

MCL Monte Carlo Localization

MCMC Markov Chain Monte Carlo

MCTS Monte Carlo Tree Search

MDP Markov Decision Process

MLE Maximum Likelihood Estimate

MtL Mow-The-Lawn Mapping Methodology

OE Optimal Estimator

OOA Out-of-Area

PDF Probability Density Function

PF Particle Filter

POMCP Partially Observable Monte Carlo Planning Process

POMDP Partially Observable Markov Decision Process

POSYDON Positioning System for Deep Ocean Navigation

REMUS Remote Environmental Monitoring UnitS

RL Reinforcement Learning

RP Random Process

SLAM Simultaneous Localization And Mapping

SMC Sequential Monte Carlo

SO Spiral Out Mapping Methodology

TAN Terrain-Aided (or Relative) Navigation

UAV Unmanned Aerial Vehicle

UUV Unmanned Underwater Vehicle

UV Unmanned Vehicle

Executive Summary

Navigation, without the dependence on external beacon systems, has been a long-standing goal for the U.S. Navy. Never has it been more important. Current external beacon systems such as global positioning system (GPS), acoustic transponders, and the Positioning System for Deep Ocean Navigation (POSYDON) have significant limitations: GPS may be jammed or spoofed, and GPS signals do not penetrate water; undersea acoustic transponders have range and accuracy limitations and require manual set-up; and POSYDON has a combination of the GPS and acoustic transponder limitations and is dependent on the deployment and operation of a large array of resources covering the ocean vastness.

One solution, known as terrain-aided (or relative) navigation (TAN), uses a terrain map coupled with onboard sensing systems to determine position through a correlation process. However, TAN requires a prior map. For autonomous systems, such as unmanned vehicles (UVs), frequently there is no accurate prior map since the mission purpose is to survey the area for the first time.

This research seeks to improve TAN by de-emphasizing the requirement for a prior map. Instead the approach actively builds a map as the vehicle conducts a terrain survey. We call this new approach active terrain-aided navigation (ATAN). Key to the approach is the balancing of *exploration* and *exploitation* objectives, where *exploration* is the goal of covering the entire area with the UV sensor system (e.g., sonar or radar) and *exploitation* is the re-localization of the UV position through exploiting the best terrain features.

The TAN approach is designed as an information theoretic framework where all components of the estimation and optimization software are designed to maximize information gain. This approach brings with it a challenge of high computational complexity due to the near real time optimization routine. However, the computational complexity can be reduced by leveraging one of the most recent techniques in Artificial Intelligence Reinforcement Learning called a partially observable Monte Carlo planning process (POMCP), which can be used to solve the dual *exploration-exploitation* optimization problem.

In conclusion, ATAN gives greater levels of autonomy that allow the UV to adapt to its environment and permit more flexibility in a variety of mission areas.

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I would like to thank my thesis advisor, Dr. Douglas Horner, whose door was always open and whose insight, direction, and assistance throughout ensured that I was progressing in the correct direction to providing a significant contribution to the field and a thesis that I am proud of.

I would also like to thank my thesis co-advisor, Dr. Monique Fargues, whose feedback and assistance were invaluable in navigating through the electrical engineering pipeline.

Also, thank you to my friends and family who allowed me to put this thesis first when required.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction to Active Terrain-Aided Navigation

The ability to navigate accurately is a well-established problem that has been made considerably easier by the invention and implementation of external localization methods such as Global Positioning System (GPS) and acoustic beacon systems. These systems have become increasingly integrated into the global infrastructure of today, even to a point of dependence. This dependence has led to a growing concern about the system vulnerabilities [1] and a desire for navigation solutions independent from external localization systems. Over the last twenty-five years, there has been significant progress in removing unmanned system dependence on external localization within the domain of terrain-aided (or relative) navigation (TAN). However, the best current techniques in TAN require an accurate prior map in order to estimate position. In many instances, a prior map is not available or does not exist, making TAN ineffective without external position localization.

1.1 Motivation

The goal of this thesis is to improve upon existing TAN methods using advances in machine learning, specifically artificial intelligence reinforcement learning, to de-emphasize the requirements of external localization and an *a priori* map. The solution demonstrates the ability for an unmanned system to navigate autonomously while also fulfilling the following objectives: minimize total positional error; completely cover a defined, bounded region with a designated sensor; create an accurate map to within a certain accuracy; and complete the task within time and energy constraints. By accomplishing this goal, the unmanned system will have the ability to quantify and make decisions about its environment, allowing the unmanned system to be more autonomous.

1.2 Statement

An external localization source (e.g., GPS) may not always be guaranteed and, when not available, results in positional uncertainty for the unmanned vehicle. For a robust system, the vehicle must instead conduct position estimation using terrestrial references. Mathe-

matically this can be represented using the following process and measurement models:

$$\begin{aligned} \mathbf{x_{k+1}} &= f(\mathbf{x_k}, \mathbf{a_k}) + \omega_\mathbf{k} \\ \mathbf{z_k} &= h(\mathbf{x_k}) + \nu_\mathbf{k}, \end{aligned} \tag{1.1}$$

in which the updated state estimate $\mathbf{x}_{k+1}$ and sensor measurement vector $\mathbf{z}_k$ utilize an vehicle state vector $\mathbf{x}_k$, a control input $\mathbf{a}_k$, and a terrain map $\mathbf{h}(\cdot) : \mathbb{R}^2 \rightarrow \mathbb{R}$ while also including process noise $\omega \in \mathbb{R}^\omega$ and measurement noise $\nu \in \mathbb{R}^\nu$.

The position estimate and map estimate can be initialized through an external localization source or an arbitrary zero-zero point: the initial position estimate $\mathbf{x}_0$ builds an initial map estimate $\mathbf{m}_0$ at $\mathbf{x}_0$.

Previous TAN approaches provide an underlying assumption of a *prior* high-confidence, low-error overall map estimate. This research seeks to degrade the emphasis of an accurate prior map by investigating a new approach to path planning through simultaneously optimizing the position and map non-linear functions (equation 1.1). Through careful trajectory planning, a balance between *exploration*, the desire to discover new areas, and *exploitation*, the requirement to localize position using previously explored areas, can be achieved. The required map accuracy, vehicle dynamics, and time (energy) add further constraints on the *exploration-exploitation* relationship to achieve the mission goal: building an accurate survey map.

1.3 Methodology

Traditional TAN serves as an alternative to external localization methods by using the correlation between sensor data and terrain data to minimize the positional uncertainty of the vehicle over a variety of terrain types and domains. For example, in the 1970s missile guidance systems used radar to conduct position localization through terrain contour matching (TERCOM) [2], [3] and, in this thesis, the autonomous underwater vehicle (AUV) uses sonar sensors as inputs to position estimation through active terrain-aided navigation (ATAN). The breadth of traditional TAN includes a large range of applications which allows for this research to be applied to all domains, including aerial, surface, and underwater vehicles.

The information process flow within ATAN is depicted in Figure 1.1. Initially, the vehicle follows a default navigational path to the target area while also conducting position estimator initialization. The cross-track error (CTE) controller implements the intended path by giving control orders to the unmanned vehicle (UV). As the UV continues through the trajectory, set and drift occur, taking the UV off course. An initial position estimate is made by the UV based on its inertial navigation system (INS) and dead-reckoning systems. Next, the UV periodically classifies the surrounding terrain and outputs a point cloud to the navigation system which conducts position estimation and updates the map estimate.

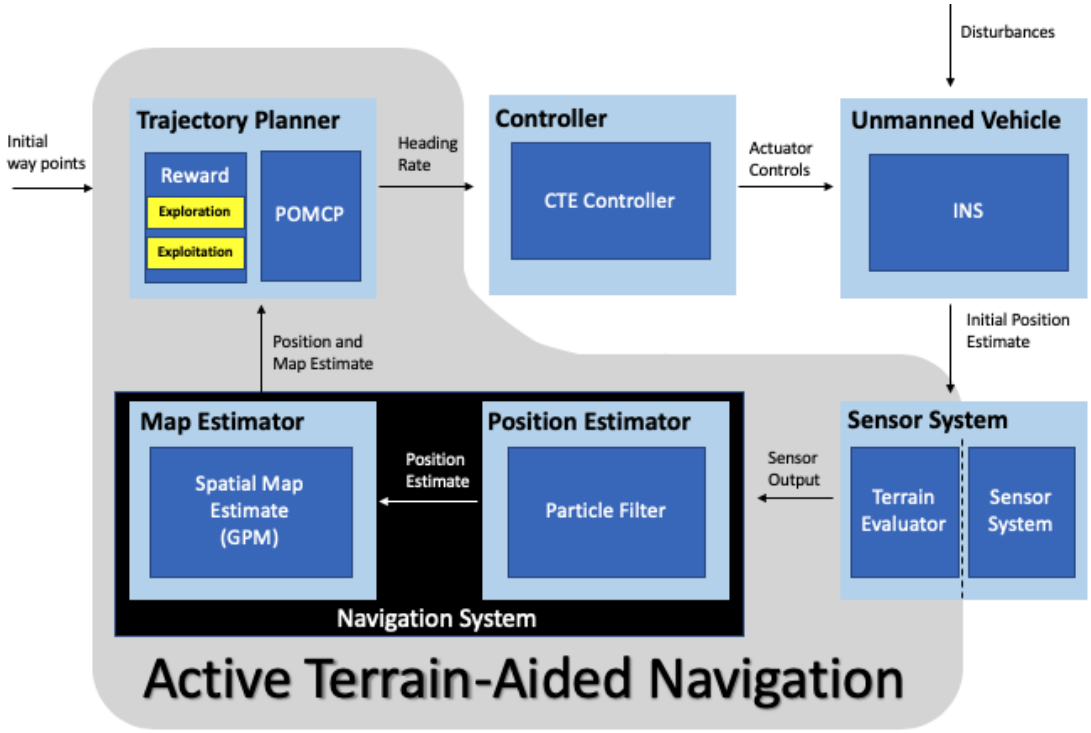


Figure 1.1. Information Process Flow for Unmanned Vehicles Using ATAN.

Finally, the trajectory planner employs the position estimate and map estimate to generate and evaluate possible trajectories, selecting the best trajectory and outputting the desired heading rate.

In many coverage problems, the best trajectory is determined by leveraging *exploration* and *exploitation*: the desire to explore all areas of the map is challenged by the obligation to reduce the positional uncertainty of the autonomous vehicle by exploiting known areas.

The autonomous vehicle must balance *exploration* and *exploitation* such that when the positional uncertainty is too great, the vehicle has a higher incentive to look for terrain advantageous to localization and when position uncertainty is low, the vehicle conducts exploration. This tension between *exploration* and *exploitation* creates the stochastic dual optimization problem that can be managed through adaptive gains of a reward function.

The partially observable Monte Carlo planning process (POMCP), is a combination of a partially observable Markov Decision Process (POMDP) and Monte Carlo tree search (MCTS), and was theoretically proposed by Lauri and Ritala to solve the *exploration-exploitation* dilemma [4], described more completely in Chapter 7. The solution of a POMDP balances *exploration* and *exploitation* through at least one optimal policy, π^*, which gives a solution over a finite horizon, with the computational complexity increasing as the number of finite horizons increase [4]. To counteract the increase of computational complexity, the POMDP can be simplified with recent machine learning techniques used in AlphaGo [5] which uses a MCTS over all POMDP trajectories and leads to a computationally manageable POMCP [6], [7]. Selective searching the POMCP through biasing [6], [7] allows for the determination of the "best" solution within a limited amount of time.

Through ATAN, the dual optimization problem between *exploration* and *exploitation* efficiently, and effectively, determines the best trajectory within the allowable computational complexity, permitting real-time application and greater levels of autonomy.

1.4 Active Terrain-Aided Navigation

An information-theoretic framework—ATAN—was created by combining TAN and the POMCP in order to provide a solution to the dual optimization problem between *exploration* and *exploitation* within the TAN environment. The key enhancement of ATAN stresses autonomy, interpreting sensor information and quantifying the environment into a map. By quantifying *exploration* and *exploitation*, the vehicle can decide its own trajectory, making the unmanned system autonomous. ATAN is comprised of four elements:

- Position Estimator (particle filter (PF)) *Sec. 3.2*
- Map Estimator (Gaussian Process Model (GPM)) *Sec. 4.1*
- Terrain Evaluator (Maximum *a posteriori* (MAP)) *Sec. 5.4*
- Trajectory Planner (POMCP) *Sec. 7.4*.

Using artificial intelligence reinforcement learning, ATAN gives the possibility of improving TAN by quantifying the environment with a spatial estimator in a Gaussian Process Model (GPM) for *exploration* which can be leveraged against a temporal Maximum *A Posteriori* (MAP) for *exploitation*. Using the GPM and MAP as inputs to a reward function, a trajectory planner (POMCP) calculates current and future states to determine the optimal trajectory, effectively and efficiently mapping an area. Together, the four components of ATAN have the potential to create an accurate map with no *a priori* information and enables the AUV to dynamically plan and execute a trajectory while balancing *exploration* and *exploitation* aspects.

Put simply, ATAN is TAN with the addition of artificial intelligence reinforcement learning: a real-time coverage planner that eliminates the requirement of an accurate *a priori* map.

1.5 Applications of Active Terrain-Aided Navigation

Autonomous systems can improve numerous operations, both militarily and non-militarily, by terminating the dependence of external beacon systems on localization. Possible missions in the military include but are not restricted to the following:

- map an area prior to an amphibious operation to determine, and subsequently avoid, obstacles
- conduct searches over a large areas
- perform mine or explosive ordinance countermeasure operations
- survey new or inaccessible geographic terrain
- carry out Intelligence, Surveillance, and Reconnaissance (ISR).

In non-military operations, the unmanned systems can also be used for a variety of tasks:

- monitor habitats, count, or observe aquatic or land animals [8], [9], [10]
- assess coral reefs [11] or plant height [12]
- conduct Emergency Response [13]
- perform security operations [14]
- expedite search and rescue operations [15].

Regardless of the operation, ATAN is a versatile navigation approach that can improve

current methods, provide better results, and improve unmanned system autonomy.

1.6 Autonomous Underwater Vehicle System Description

This thesis demonstrates ATAN in an underwater environment though a MATLAB simulation based on a modified Remote Environmental Monitoring UnitS (REMUS) 100. The modified REMUS 100 (Figure 1.2) operated by Center for Autonomous Vehicle Research (CAVR) at the Naval Postgraduate School (NPS) can support a wide variety of mission sets, including hydro-graphic surveys, environmental monitoring, and mapping.

Figure 1.2. REMUS 100 Vehicle.

The system description and implementation, shown in Table 1.1, was modeled with parameters shown in italics. The REMUS also has a GPS antenna and the ability to use acoustic transponders of the Long BaseLine (LBL) and Ultra Short BaseLine (USBL) systems.

Table 1.1. REMUS Configuration and Model Parameters.

Part/Module	**Used Setup/Utilized Implementation**
Robot	REMUS 100. Length 2.28m. Diameter 0.19m. Weight 40kg. Speed 0.25-2.57 m/s. Depth Range 3m-100m. Battery Duration 22 hours. [16], [17] *Simulated Speed 1.8 m/s*.
Propulsion	Direct drive DC brush-less motor directly connected to open three bladed propeller [16], [17].
Locomotion Control	2 coupled yaw and pitch fins. Cross Tunnel Thrusters [17]. *Simulated Heading Angular Rates* $[-2,-\frac{4}{3},-\frac{2}{3},0,\frac{2}{3},\frac{4}{3},2]$ *°/s*.
Inertial Navigation System (INS)	SeaDeViL INS navigational uncertainty equal to 0.5 percent distance traveled [16], [17]. *Simulated uncertainty equal to 0.7 percent distance traveled.*
Acoustic Doppler Current Profiler (ADCP)	900 kHz, four downward-looking, four upward-looking transponders measure forward and side velocity of vehicle [16], [17].
Sonar	BlueView MB2250-N downward-looking, ultra-high resolution sonar that operates 256 beams 0.18m apart providing 0.6cm resolution over a 40° field of view [18]. *Simulated Sonar as 10m circular patch.*
Computational Resources	(2) intel Core i9 processors for the secondary controller and sensor processing. (2) Hydroid CPUs and hard drives for the main controller and the side scan sonar.
Map	3D grid map with grid size 1m×1m.

1.7 Structure

The remainder of this thesis is organized as follows. Chapter 2 discusses the most recent research and methodologies for underwater mapping. Chapter 3 examines different types of position estimation techniques, specifically Kalman filter (KF) and particle filter (PF), and shows the ATAN implementation of the PF. Chapter 4 and 5 define and derive *exploration* and *exploitation* as a Gaussian Process Model (GPM) and Maximum *a posteriori* (MAP) estimate, respectively. *Exploration* and *exploitation* are then used to define the reward function, using adaptive gains, in Chapter 6, presenting the stochastic dual optimization equation. In Chapter 7, the methodology of Markov Decision Processes (MDPs), partially observable Markov Decision Processes (POMDPs), and Monte Carlo tree search (MCTS) are reviewed and then combined to form the partially observable Monte Carlo planning process (POMCP). Simulation results from MATLAB are presented and analyzed in Chapter 8. The thesis concludes with Chapter 9 which discusses future work.

CHAPTER 2:
Background

This chapter provides background and a literature review for Active Terrain Aided Navigation (ATAN). The relevant literature includes several related fields but emphasizes robotic autonomy. Subcategories include position estimation, geophysical navigation techniques, coverage planning algorithms, information theory, and artificial intelligence reinforcement learning techniques for trajectory planning.

2.1 Position Estimation

The accuracy of the map is entirely dependent on the position estimate of the unmanned vehicle (UV) carrying the sensor system. In most instances, positional errors are nonlinear and the UV requires the capability of position estimation through either external localization sources or through an internal correlation of sensor information and the terrain. However, many of the methods utilize external localization sources which have operational vulnerabilities for both civilian and military missions.

2.1.1 GPS Vulnerability

GPS utilizes a constellation of satellites which broadcast a microwave signal used by a GPS receiver to localize the position of the UV. While radically changing the world with respect to all aspects of economic, military, and recreational tasks, GPS also possesses vulnerabilities and limitations.

Jamming and spoofing are two inherent vulnerabilities within GPS. Jamming can be used defensively or offensively by transmitting an additional noise signal across one or more GPS frequencies, raising the overall noise level at the GPS receiver which can generate more error and cause an overload at the GPS receiver or a loss of GPS lock. In Russia, GPS jammers are placed in key areas, in accordance with its defense initiative *Pole-21*, to protect its domestic infrastructure in the event of a war [19]. In an offensive sense, GPS jamming was reportedly used by China to jam GPS on U.S. drones while the drones conducted reconnaissance during territorial disputes in the Spratly Islands [19]. While GPS jamming

will affect the ability of the UV to conduct localization, spoofing may be more dangerous. Spoofing transmits a false GPS signal indistinguishable from the real, correct GPS signal to either provide a false position or in a more extreme situation, take control of the UV. In 2016, two U.S. Navy ships were suspected of having their GPS systems tampered with, causing them to violate international waters. In 2011, Iran used spoofing to capture a U.S. drone by interfering with its GPS and having the drone land in Iran when the drone GPS read a position that was in Afghanistan [20]. Both jamming and spoofing exploit significant vulnerabilities to GPS which can lead to the failure of a mission, damage or destruction of vital equipment, or loss of human life.

For AUVs which predominately operate underwater, GPS has another glaring vulnerability: GPS signals cannot penetrate water. Once the AUV is underwater, the GPS signal can no longer be used for localization, making the AUV reliant on onboard sensors. For operations in deep water, or for long duration operations, the AUV must re-surface to obtain a fix which consumes valuable time, energy, and resources while also risking a possible counter detection event or collision in shallow water high contact density environments.

The vulnerabilities of GPS are well-known, leading to a desire of reducing the dependence of position localization on GPS [21].

2.1.2 Acoustic Undersea Beacon Systems

In the undersea domain, acoustic beacon systems are a viable method for position estimation that can compliment GPS. With an acoustic beacon system, the AUV triangulates position through the deliberate, pre-planned placement of fixed undersea acoustic beacons. Common systems used today are Long Baseline (LBL), Ultra Short Baseline (USBL), and Short Baseline (SBL), or some combination of the three [22]. Essentially, the different types of beacon systems correspond to the different range and geometry between the beacon system and the AUV. Regardless of the system, a beacon can operate as one of the following:

- Transponder: beacon receives predetermined interrogation pulse and transmits an acoustic ping
- Responder: beacon transmits a predetermined pulse after its electrically triggered
- Pinger: beacon transmits acoustic signals continually on a specific frequency and pulse length.

However, beacons have limitations. In both a monetary and temporal sense, beacons have significant costs while providing limited accuracy. The ocean is a corrosive environment, and time needs to be devoted to ensure that beacons are properly maintained, calibrated, and operated. Further, beacons offer limited exactness within a finite range—±10m over 2000 meters for LBL and ±1m over 1500 meters for USBL [16]—and must be placed prior to use, which may be inconvenient or impracticable in some places like conducting under-ice operations. Additionally, some military operations may require a covert posture or execution within a limited time. Using active acoustic beacons can lead to the AUV being counter detected or exceeding time constraints through placing and retrieving beacons, minimizing operational flexibility.

2.1.3 Positioning System for Deep Ocean Navigation (POSYDON)

Defense Advanced Research Projects Agency (DARPA) is making a hybrid system between GPS localization and beacon localization called Positioning System for Deep Ocean Navigation (POSYDON) [23]. A surface buoy, using GPS to localize itself, can act as a beacon system to transmit underwater acoustic signals. This coordinated buoy system attempts to imitate actual GPS using multiple small long-range acoustic sources (satellites) to allow accurate positioning via acoustic signals (microwaves) without the AUV surfacing for a GPS fix. While Phase I of a three-phase project is underway, there are still multiple sources of error and vulnerabilities from using GPS and the acoustic beacons, as discussed in Sections 2.1.1-2.1.2.

2.2 Geophysical Navigation

Autonomous vehicles require a methodology to estimate position. Two geophysical navigation techniques—TAN and simultaneous localization and mapping (SLAM)—minimize the emphasis on external beacon systems by using terrain or physical features to produce a position estimate. Both TAN and SLAM over the last couple of years have made significant progress in navigation independent from external beacon systems and provide notable contributions to ATAN.

2.2.1 Terrain Aided (Relative) Navigation (TAN)

TAN is a substantial building block with multiple attributes contributing to ATAN. TAN reduces positional uncertainty of an unmanned vehicle by correlating sensor data and a known terrain map, as shown in Figure 2.1.

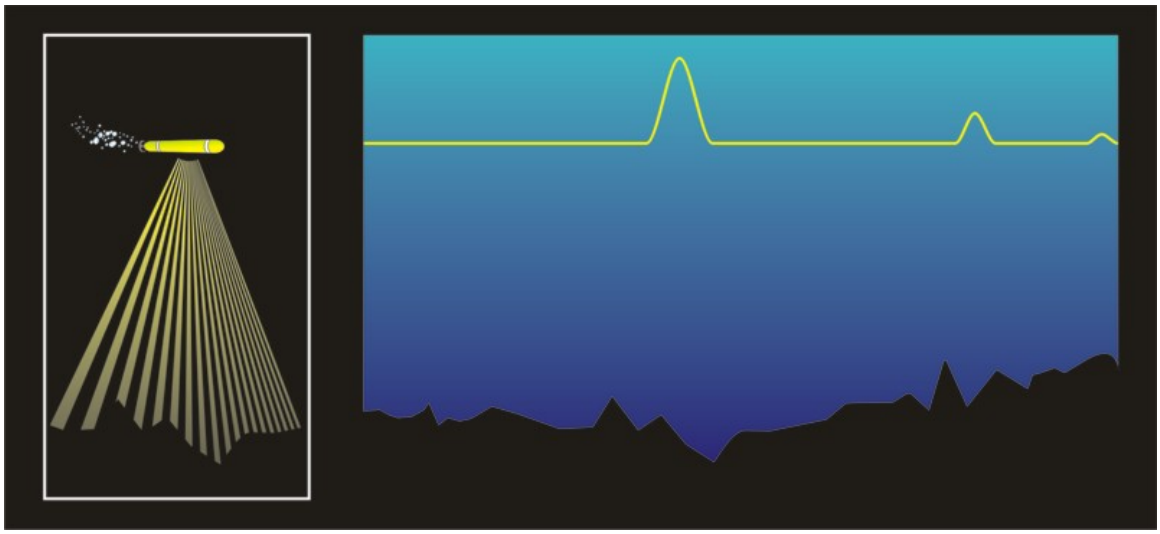


Figure 2.1. Correlation of Observation (left) and Terrain (right) in One Dimension. Source: [24].

The highest correlation between the sensor data and the known map, shown in Figure 2.1 (right), is the position estimate. In the underwater domain, TAN has become a popular choice of position estimation and navigation, as GPS is unable to penetrate water at depth.

In 2004, Nygren and Jansson introduced the Correlator (Correlation) Method—a recursive position estimation by correlating multi-beam sensors and terrain information—to use on unmanned aerial vehicles (UAVs), AUVs, and submarines [25]. The following year, Nygren utilized a submersible vehicle that showcased significant improvement of position estimation over a 65 km test track, yielding root mean squared (RMS) error less than one meter for simulation and a few meters for the sea trial [26]. Two main outcomes of Nygren's dissertation were well known sensor beam placements converged to a Gaussian distribution when the number of sensor beams approached infinity, and that the accuracy of the position estimation can be judged using the lower bound on the variance, Cramér-Rao Lower Bound (CRLB) [26].

At Stanford, Rock and Kimball discussed and implemented Terrain Relative Navigation (TRN) to combat issues listed previously in this chapter. Specifically, Rock and Kimball presented TRN to navigate [27] and localize [28] the AUV position underneath icebergs. Since icebergs are constantly moving (either linearly or rotationally), the AUV can conduct relative localization using a particle filter and the correlation between a *a priori* map and its sensor measurements [28].

Gao, Peng, Zhou, Wang, and Xu proposed an improved matching algorithm (Terrain Matching Localization with Gradient Fitting) to minimize computational complexity and raise positional accuracy of the position estimator [29]. By partitioning the area into smaller sections, the gradient of each section can be fitted to a normal distribution and matched to the vehicle observations, reducing the number of comparisons required and reducing interference of repetitive terrain [29].

One of the main, well documented difficulties within TAN has been localizing over featureless terrain [26], [30], [31]. Nygren proposed increasing the sensor beam count which gives the ability to calculate smaller terrain gradients through interpolation [26], [32]. Later, Meduna, Ewan, and Rock showed that interpolation leads to higher computational complexity and possible over-fitting for minimal performance gains [32], [33] and a more robust solution was presented by Dektor and Rock by exponentially weighting PF outputs (p^{α} where $0 < \alpha < 1$) based on map error, sensor error, and terrain information [30]. In subsequent work, Dektor and Rock built on the exponentially weighting PF outputs by examining reasons and proposing a methodology for detecting false convergence of the PF, which successfully allowed an AUV to localize while driving over flat terrains [31].

2.2.2 Simultaneous Localization and Mapping (SLAM)

SLAM has been largely viewed as the precursor for true autonomy [34]—concurrently conducting AUV position localization with progressive map building [35]—and consists of two components, front end and back end, shown in Figure 2.2.

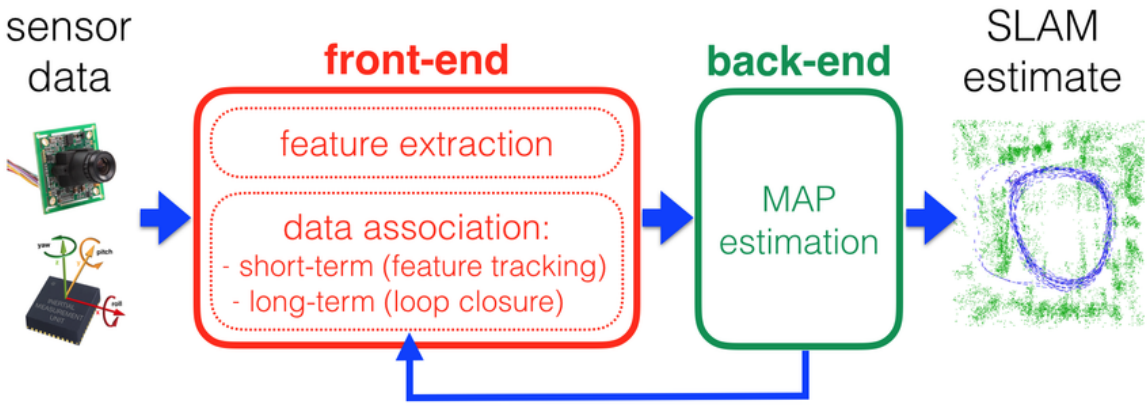


Figure 2.2. SLAM Front End and Back End Diagram. Source: [36].

The front end extracts topological features from sensor data and assigns measurements to each feature while the back end calculates a map estimate [36]. One of the more important aspects of SLAM is data association, or more specifically loop closure. As the AUV drives over a previously explored area, the current loop closes, anchoring the map estimate and reducing the uncertainties in robot and feature estimates [34]. Without loop closure, the SLAM problem exhibits quadratic growth [34], for each new feature encountered could lead to failure of the problem if the loop is not closed in a reasonable time. The map estimate registers all landmarks and obstacles discovered by the vehicle, but further performs an optimization to minimize positional uncertainty of the vehicle and the features [34].

From a theoretical and conceptual perspective, the SLAM problem is considered solved, but from an implementation perspective SLAM still has difficulties in relation to computational complexity (proportional to the square of the number of landmarks) [35] and the convergence of the algorithm [37]. Further, SLAM is a relativity model using the relative joint state model of the robot pose and its relative locations to the observed landmarks [37]. The process model affects the vehicle pose states, and the observation model only affects the vehicle-landmark pair [37]. While the optimal algorithms aim to reduce the computational complexity of the problem, the solutions still result in estimates and covariances that are equal to those without the reduction [37].

Active SLAM

Active SLAM research utilizes the SLAM framework, but adds an *exploration* versus *exploitation* decision making dynamic [36]. Two popular models for decision making, information theoretic and POMDP [36], are discussed further in Sections 2.5 and 2.6. The optimal decision, within time and computational budget constraints, maximizes the area explored while also minimizing the positional uncertainty [38].

2.2.3 Geophysical Navigation in ATAN

ATAN draws techniques from both the TAN and the SLAM domains. For the purposes of the thesis, TAN provides the underlying structure of ATAN since TAN involves a complete estimate of the terrain associated with the UV coverage mission and permits a correlation estimate at any position. Conversely, SLAM techniques involve a data association task for determining the correlation by a sensor measurement and a finite number of landmarks. Overall the differences between TAN and SLAM are minor and future work (not covered in this thesis) will include the development of an Active SLAM component incorporating the techniques used in this thesis.

Within TAN, the correlation method was adopted for position estimation (Chapter 3) and the ability to model terrain using a Gaussian distribution was used for the *exploration* component (Chapter 4). The position estimation can be further improved using methods in [31], discussed in future work not covered by this thesis.

The underlying fundamentals of SLAM relate to the approach of this thesis: feature extraction, data association, and map generation. However, instead of feature classification, each 1m $\times$ 1m cell is quantified through the unevenness of the terrain. The position estimator compares the current AUV observation with projected position observations to determine the best position. The map is generated through the uncertainty of the AUV position and AUV sensors using a model discussed in Section 2.4. While SLAM attempts to do all calculations concurrently, this thesis intentionally separates the process, preferring a dual optimization problem with *exploration* and *exploitation*.

Applicable methods to ATAN used within Active SLAM include path planning, information theoretic, and reinforcement learning. These concepts are universal between Active SLAM and ATAN, as both SLAM and TAN share the goal of maximizing *exploration*

while minimizing vehicle error. The difference between ATAN and SLAM is that ATAN intentionally breaks the optimization apart and instead performs a dual optimization between *exploration* and *exploitation* before re-combining while SLAM performs a single optimization. As a result, *exploration* and *exploitation* can be modeled independently and then brought together with a reward function (Chapter 6): a mathematical framework that prioritizes either *exploration* (Chapter 4) or *exploitation* (Chapter 5) depending on the state of the UV.

2.3 Path Planning

Path planning has been around since at least 1991 with the goal of collision-free trajectories that have the robot travel from an initial state to the goal state in the fastest time [39]. Decision-theoretic planning was introduced around 2006 [40], to select the best path for the goal state. Recent techniques in path planning utilize either Gaussian Processes (GPs) or maximum Likelihood estimates (MLEs) which was used prior to implementing and designing ATAN.

2.3.1 Gaussian Process (GP)

GPs can help classify value functions within the path planning environment. Prágr, Čížek, Bayer, and Faigl use a GP to quantify terrain traversal cost (TTC) [41] for a multi-legged walking robot. The robot explores by using the GP predictive variance to determine whether to discover new areas (*exploration*) or decrease uncertainty within the TTC (*exploitation*) [41]. The key takeaway from Prágr et al. is the combination of Gaussian Processes using a Robust Bayesian Committee Machine (RBCM) which reduces the computational complexity from O(N^3) to approximately O(N) [41]. In another instance of using a GP, Nardi and Stachniss use a GP mixture model coupled with a top-down aerial picture to reach a desired end state resulting in improved navigation [42]. The GP mixture model inputs experiences from the vehicle to determine the proper action between exploring new areas and exploiting the environment by circumventing obstacles or difficult terrain [42]. Lastly, Rui and Chitre utilize a path planning algorithm to reach a desired end state by using a prior bathymetry map combined with a GPM to minimize the AUV uncertainty (*exploitation*) over the path length [43].

2.3.2 Maximum Likelihood Estimate (MLE)

The MLE determines the most probable outcome using a probabilistic model of observed data, which can be useful to exploit partially observable environments for path planning. Zambom, Seguin, and Zhao use a MLE to determine future positions of moving obstacles to determine the shortest path [44]. Ornik and Topcu propose a MLE path planning algorithm within a previously unknown time varying environment [45]. Using only previously observed information and known system dynamics, the MLE can either improve the accuracy of the MLE by exploring new areas or use current estimates of the environment (*exploitation*) to progress closer to the objective [45]. Zhang, Hu, Chadha, and Singh propose a Maximum Likelihood Path Planning algorithm that determines the maximum likelihood to reach the goal (or end of trajectory) and avoid obstacles, exploiting the environment within sensor range and probabilistically determining the environment beyond the sensor range [46]. By using a limited trajectory, the computational complexity is lowered to a manageable level, allowing the robot to quickly make decisions regarding path planning [46].

2.3.3 Path Planning in ATAN

Path planning is an important part of ATAN, as the correct choice between *exploration* and *exploitation* must be considered and evaluated during the path planning portion.

The *exploration* component (Chapter 4) must be able to measure the amount of *exploration* over all possible trajectories, including unexplored areas, areas with large uncertainty, and areas perfectly known. The quantification of *exploration* lends itself to a GPM as a MLE cannot accurately estimate unknown areas.

Conversely, the *exploitation* component (Chapter 5) uses a MLE, as areas previously explored can be measured through terrain classification and positional uncertainty of the vehicle when the location was measured. This thesis utilizes a similar approach to the Zhang et al. Maximum Likelihood Path Planning algorithm with slight modifications.

2.4 Coverage Planning

Coverage has many similarities with path planning, as both require an understanding of *exploration* and *exploitation*. However, the key difference is that while path planning gets from a starting location to an ending location, coverage planning must pass over all points

of interest within the map and is evaluated by the quality of the finished map. There are two types of coverage planning: inspection planning and full coverage planning.

2.4.1 Inspection Planning

Inspection planning, a subset of coverage planning, uses robots to examine certain features of an overall map with the goal of minimizing trajectory length and maximizing the number of features visited [47]. Essentially, the *exploration* component attempts to maximize the number of features visited and the *exploitation* component attempts to maximize information collected at each feature, which is made difficult since the computational complexity of inspection planning grows exponentially with the number of features [47]. Fu, Kuntz, Salzman, and Alterovitz propose an inspection planning method called Incremental Random Inspection-roadmap Search (IRIS) [47], which provides a structure that efficiently computes asymptotically-optimal inspection plans [47]. Another inspection planning algorithm, ergodic coverage directs and operates robots to certain areas for specified times based on the probability of locating a feature within that search space [48]. Furthermore, additions to ergodic coverage given by Ayvali, Salman, and Choset can be applied to non-linear problems, incorporating sensor areas and obstacle avoidance [48].

2.4.2 Full Coverage Planning

Full coverage planning covers every area of the map with a sensor. The oldest approach is the Boustrophedon method, synonymous with the back and forth motion of mowing the lawn [40]. The issue with the Boustrophedon method is that while *exploration* is maximized, *exploitation* does not exist as areas are not covered multiple times, which translates to a growing positional uncertainty.

One of the earlier applications of full coverage planning was frontier-based exploration which uses an occupancy grid to track previously explored areas [4], [49]. The intersection between the known and unknown areas, called frontiers, were visited until there were no frontiers remaining, an equivalent way to express complete coverage [49].

2.4.3 Coverage Planning in ATAN

ATAN utilizes *exploration* tactics from [4], as further discussed in Section 2.5. However, ATAN can be improved further with inspection planning to specifically target unexplored areas when certain thresholds of coverage are met, which is left for future work.

2.5 Information Theory

Claude Shannon started the Information Theory field in 1948 to "measure information, choice, and uncertainty" [50]. The field of Information Theory has evolved significantly since [51], [52], encompassing an expansive field that includes probability theory, entropy, GPs, and mutual information while maintaining a common theme: the quantification of information.

In most robotic applications, correctly measuring the change of uncertainty directly contributes to the success of the the robot making correct decisions, particularly within the *exploration* and *exploitation* domains.

One popular method for quantifying the *exploration-exploitation* dilemma is through the uncertainty measurement of Rao-Blackwellized particle filters. Stachniss, Grisetti, and Burgard use the change of entropy within a Rao-Blackwellized particle filter to measure the information gain of an action [53]. By combining multiple actions together, the *exploration-exploitation* dynamic can be resolved by the greatest reduction of uncertainty within the posterior [53]. Carlone, Du, Ng, Bona, and Indri also investigate the uncertainty within Rao-Blackwellized particle filters, but instead use the Kullback Leibler Divergence (KLD) to determine the best expected information gain of a policy which is a function of information gain (*exploration*) and loop closure (*exploitation*) [54]. The advantage of quantifying *exploration* further allows for the re-visitation of previously explored areas to lower the uncertainty of the higher uncertainty areas [54].

Another method to quantify exploration was proposed by Lauri and Ritala to model observations, $z = f(s, a)$, probabilistically as a function of state s and action a, which can then be used to calculate the belief state b through Bayesian filtering [4]. The information gain of a trajectory (measure of *exploration*) is a function of the belief state [4].

Ayvali, Salman, and Choset use ergodicity to determine exploration trajectories which result

in weighting certain areas of a map based on the probability of success [48]. Taking the Fourier Transform of the trajectory and the desired coverage distributions, higher weighting values can be given to lower frequency components which correspond to higher similarities between trajectory and desired coverage, leading to efficient *exploration* [48]. The norm of each distribution can then be compared using KLD, where the best trajectory will lead to the highest mutual information [48].

Rui and Chitre utilize a path planning algorithm to reach a desired end state by using a prior bathymetry map combined with a GPM and reinforcement learning (RL) to minimize localization uncertainty [43]. Using information entropy of the terrain, the GPM determines the estimated value function of localization [43].

2.5.1 Information Theory in ATAN

ATAN conducts a dual optimization through an information-theoretic framework that properly prioritizes *exploration* and *exploitation*. The *exploration* (Chapter 4) component is calculated by a spatial GPM and mutual information gain of the KLD. The position estimation uses a PF to represent the multi-modal, non-Gaussian, non-linear error associated with the vehicle. The variance of the PF and GPM are then used as inputs to the gain values of the reward function to prioritize *exploration* and *exploitation* (Chapter 6).

2.6 Reinforcement Learning

RL is subsection of machine learning where machine agents make decisions to maximize the total rewards over a set of actions. Common forms of RL include the POMDP, MCTS, or the combination of the two: POMCP.

2.6.1 Partially Observable Markov Decision Process (POMDP)

A POMDP is a decision process for partially observable environments that has been used to make decisions in real-time scenarios. Cai, Luo, Saxena, Hsu, and Lee at the National University of Singapore utilized the combination of a search algorithm to bias a POMDP to minimize the computational complexity and provide real time solutions for autonomous driving through a crowded environment, called LeTS-Drive [55]. Lauri and Ritala proposed a framework to find the best control policy to maximize *Exploration* utilizing an open loop

POMDP to efficiently and effectively explore an area using a Turtlebot [4] equipped with laser range finders (LRF). The Turtlebot operates within a Robot Operating System (ROS) environment that starts with an empty goal, empty result, and uses feedback to generate *exploration* tasks [4]. The *exploration* task is then generated using a planner software of costmaps to record open areas, obstacles, and unknown space using the probability of occupying the cell [4]. Based on the costmaps, the Turtlebot calculates the best trajectory from randomized trajectories (ten trajectories of three control actions) using a POMDP and executes the best trajectory using move_base [4]. Once the trajectory is complete, the Turtlebot pauses and repeats the process [4].

2.6.2 Monte Carlo Tree Search (MCTS)

MCTS is an effective technique for searching over a rich configuration space. It uses random sampling to search a decision space, normally structured as a tree, with the purpose of finding optimal decisions within the given domain [7]. MCTS is a popular decision-theoretic framework within Game Theory, mapping considerable branching factors of large fully observable decision spaces like Chess, Checkers, Chinese Checkers, and AlphaGo. Liu, Zhou, Cao, Qu, Yeung, and Chung used a biased MCTS coupled with deep reinforcement learning to model Chinese Checkers [56]. The key difference between Chinese Checkers compared to most board games is that all pieces stay on the board, and all pieces can move in any direction which leads to a non-narrowing, consistent branching factor [56].

AlphaGo

AlphaGo has an extremely large search space with a substantial branching factor correlating to an average of 250 moves per play [7]. Evaluation of multiple moves can be represented mathematically through the number of moves available, $b \approx 250$, and number of moves per game, $d \approx 150$, yielding a search space of b^d [5]. AlphaGo uses two networks: policy and network [5]. The policy network computes the conditional probability, $p(a|s)$, of all legal moves a given the board representation s [5]. The value network computes the expected outcome of the move using the future board s'. With b^d possible moves [5], the computational complexity to rival human experts is difficult.

However, by using and effectively biasing the MCTS, AlphaGo has achieved success, noted by a 99.8% win rate against programs and winning five games to zero against the

European Champion [5]. Within the MCTS, each node stores a prior probability $p(s,a)$, a visit count $N(s,a)$, and a reward $Q(s,a)$ to bias the MCTS during each iteration [5]. The bias allows the MCTS to continue choosing branches with higher values (*exploitation*) but also ensuring other possibly optimal policies are also searched (*exploration*), allowing the MCTS to quickly search an immense decision space and converge close to the optimal value function [5]. The MCTS is further improved using deep convolutional neural networks to reduce the search space (depth and breadth of the tree), with supervised and reinforcement learning occurring to better tailor the search [5].

The ability of AlphaGo to quickly and advantageously search a large decision space set the foundation for this thesis: incorporating RL into TAN. RL improves TAN by utilizing the MCTS coupled with biasing to perform real-time evaluations of future trajectories for *exploration* and *exploitation*, yielding efficient coverage while also minimizing UV position error.

2.6.3 Partially Observable Monte Carlo Planning (POMCP)

A POMCP is a combination of the POMDP and the MCTS, extending the MCTS to a partially observable environment [6] and can be used to solve finite POMDPs [4]. There are multiple ways to bias the POMCP, with Silver and Veness providing a biasing technique that allows for *exploration* of all branches within a node, and then selecting the branch with the highest reward for further searches [6]. In another approach to biasing the POMCP, Castellini, Chalkiadakis, and Farinelli constrain the search of the POMCP through probabilistic Markov random fields [57]. Bai, Wu, and Chen propose two POMCP algorithms that utilize cumulative reward from an MCTS search, represented as an unknown distribution random variable, to weight *exploration* and *exploitation* inside the MCTS [58].

2.6.4 Reinforcement Learning within ATAN

ATAN utilizes key aspects of RL, specifically from AlphaGo, to aid in the determination of the next trajectory by implementing a POMCP as a trajectory evaluator. The POMCP equates to the combination of a POMDP and a MCTS. Within ATAN, the POMDP models the *exploration* and *exploitation* components through the first horizon consisting of multiple trajectories of a certain distance and the MCTS searching beyond the first horizon to a further secondary horizon. Each horizon exponentially increases the computational complexity of

the trajectory evaluator, but by using the MCTS only selected trajectories are required to be calculated. Further, the MCTS can be improved through biasing, using the same biasing techniques shown in [6]. Through the use of techniques in RL, ATAN determines the best trajectory within a finite time to correctly prioritize *exploration* and *exploitation*.

2.7 Contributions of This Thesis

This thesis responds to the Department of Defense Artificial Intelligence initiative [59] by increasing the decision capability of autonomous robots in both military and civilian applications.

ATAN quantifies the terrain through a Boltzmann entropy representation [60], [61], but more specifically, our work presents the following as major contributions:

- A real-time information-theoretic trajectory planner that adaptively leverages *exploration* and *exploitation* to solve for the best trajectory within a computational limit
- A real-time *exploration* component that maximizes the information gain of the selected trajectory through a GPM
- A real-time mapping algorithm that does not require a prior map
- A comparative framework that allows for analysis of different approaches—POMDP, POMCP, and biased POMCP—within ATAN.

Through these contributions, ATAN stresses greater levels of autonomy for UVs and improves upon the foundation and application of TAN.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Position Estimator

Position estimation is essential for the success of ATAN. Without the ability to accurately estimate position, the positional uncertainty of the UV increases over distance traveled, yielding a less accurate map. Further, an inaccurate position estimator eliminates the ability to evaluate and employ the best trajectory for *exploration*, *exploitation*, or a combination of the two. Therefore, ATAN requires a robust position estimator to ensure positional accuracy. This chapter introduces and designs the optimal position estimator prior to evaluating the performance of the optimal position estimator.

3.1 Introduction to Optimal Position Estimation

Two popular types of optimal position estimators—the Kalman filter (KF) and the particle filter (PF)—use stochastic estimation to minimize positional uncertainty.

The KF is a linear, recursive position estimator that uses a set of prediction and update equations to calculate a position estimate for a UV. The prediction step starts with an *a priori* state estimate; an error covariance; a control action; and a process noise covariance to calculate the state estimate and the respective error covariance. The update step then uses the state estimate and error covariance as well as observation noise covariance to determine the *a posteriori* state estimate and error covariance. In order for the KF to correctly model the UV, the KF necessitates a linear system model in which system errors, vehicle position, and observations follow a Gaussian distribution. For non-linear models, the KF can be adjusted using the Extended Kalman Filter (EKF), which approximates the non-linearity by using Taylor series expansion to create a linear system. While the KF is useful for position estimation, the KF has the following limitations: the KF computational complexity, $\mathbb{O}(N^2)$, poses limitations on larger data sets, where N is the number of data points [62]; the KF cannot use negative information and does not have the ability to see a feature that isn't expected [62]; and the KF builds on previous data, which can lead to failure of the model if the previous data is inaccurate [62].

The other common position estimator, the PF, uses a probabilistic approach to conduct

position estimation. The PF generates possible hypotheses around a position estimate $\hat{x_t}$, called particles, that each contain individual state estimates. A transition model moves each particle in accordance with the vehicle motion model, and a measurement model determines the observation of each particle, z_t, using an onboard map. The measurement model further calculates the particle weight, $w_t = p(x_t|z_t)$, through the correlation of the particle observation and the actual observation of the vehicle, in which the highest weights give better position estimates. The PF keeps the higher weighted particles, and the process is repeated.

The main advantage of the PF is the ability to track multi-modal, non-linear, non-Gaussian probability densities [63], [64], [65], [66]. Thrun, Burgard, and Fox also provide additional PF advantages: the PF computational complexity, $\mathbb{O}(N)$, is linear with respect to the number of particles, N [67]; the PF works for any high dimensional system, observation, or motion model [67]; the PF allows for multiple different types of combinations of probability distributions [67]; the PF maintains the ability to correct itself after converging towards an erroneous state (kidnapping); and the PF algorithm is relatively easy to implement [67].

Thrun, Burgard, and Fox also give the following PF limitations: the PF can suffer from a loss of particle diversification when samples converge [67]; the PF may fail with minimal noise present [67]; the PF performance is difficult to measure [67]; and the PF is is hard to predict [67].

Geophysical navigation introduces multiple error sources [33] that combine into a non-linear, non-Gaussian problem [68], [69], [70]. As the limitations of the PF can be alleviated, as discussed in Section 3.2, the advantages of the PF allow for a better position estimation and allow ATAN to better estimate a non-linear problem that does not conform to Gaussian distributions.

3.2 Particle Filter

The PF is a position estimator that can be used for complicated, nonlinear applications such as ATAN. Starting with multiple state hypothesises, or particles, the PF converges to the best hypothesis (Figure 3.1) through Bayes filtering: estimating the current state based on sensor measurements and independent, identically distributed (IID) data [71].

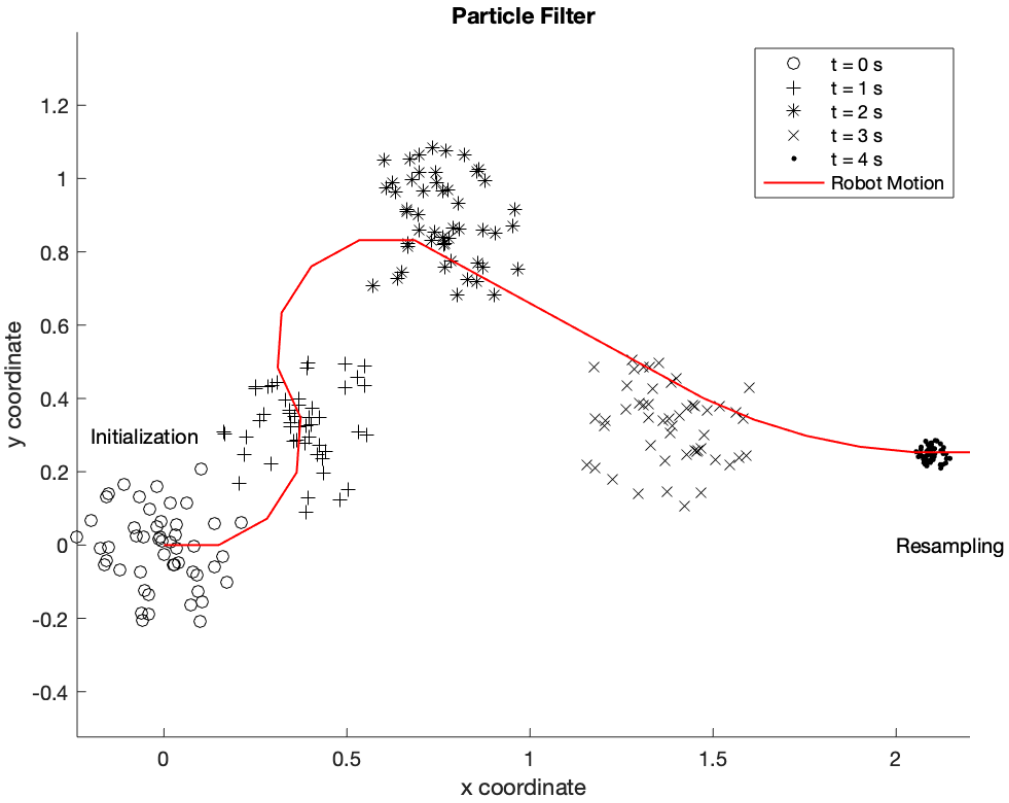


Figure 3.1. Two-Dimensional PF Visual Representation with State Hypothesess During Different Time Updates.

As seen in Figure 3.1, the particles initially start covering a large distribution, and as the robot moves through the environment, converge on the location of the robot.

3.2.1 Monte Carlo Localization (MCL)

In this thesis, a specific type of PF, the monte carlo localization (MCL), shown in Algorithm 1, is used to localize the UV. The MCL consists of an Initialization, Motion Model (Line 4), Sensor Model (Line 5), and a Resampling process (Lines 8-11) where B_t is the full belief state , a_t is the action, z_t is the observation, n is the particle, and N is the total number of particles.

Algorithm 1: Monte Carlo Localization [67]

1 **MCL**(B_{t-1}, a_t, z_t, n);
2 $\overline{B}_t = B_t = 0$;
3 **for** $n = 1$ *to* N **do**
4 $\quad b_t^{[n]} =$ **motion model**$(a_t, b_{t-1}^{[n]})$;
5 $\quad w_t^{[n]} =$ **sensor model**$(z_t, b_t^{[n]}, n)$;
6 $\quad \overline{B}_t = \overline{B}_t + < b_t^{[n]}, w_t^{[n]} >$;
7 **end**
8 **for** $n = 1$ *to* N **do**
9 $\quad$ draw i with probability $\propto w_t^i$;
10 $\quad$ add b_t^i to B_t ;
11 **end**
12 **return** B_t

The initialization consists of the particle generation and the initial belief state, b_{t_0}. Using the initial belief state b_{t_0}, sensor measurement z_{t_0}, and action sequence a_{t_0}, the initial position estimate $\hat{x}_{t_0}$ can be determined through the maximum conditional probability,

$$\hat{x} = \underset{z \in \mathbb{Z}}{arg\,max}\; p(b_{t_0}|z_{t_0}). \tag{3.1}$$

The particle generation utilizes a bootstrap method, random sampling of a data-set with replacement, to produce a Gaussian distribution $\sim\mathcal{N}(\hat{x}, \sigma^2)$ around the position estimate. The output of the initialization gives N particles representing the current belief state $(b_{t_1}, ...b_{t_N}) \in B_t$ and the posterior probability density of the current belief state.

The motion model, line 4 in algorithm 1, uses the inputs of a control action a_t and prior belief state $b_{t-1}^{[n]}$ to output the current belief state $b_t^{[n]}$ by updating all particles based on the control action of the vehicle. For example, if the UV chooses an action of maintaining course and speed then all particles maintain course and speed, or if the UV turns right at a heading rate of two degrees per minute then all particles turn right using a heading rate of two degrees per minute. With each control action, the particles diverge, an expected result as the positional uncertainty increases as the UV moves through the trajectory. A key aspect of the motion model is the incorporation of the control actuator error which can lead to the

PF converging on the wrong solution if the control actuator error is modeled incorrectly or not given a large enough magnitude. Therefore, error within the motion model must incorporate the error of the UV movement model to ensure PF effectiveness.

The sensor model, line 5 in algorithm 1, uses the output belief state of the motion model, $b_t^{[n]}$, and computes the weights, or level of importance, of each particle using the observation of each particle. This can be more easily described using Figure 3.2.

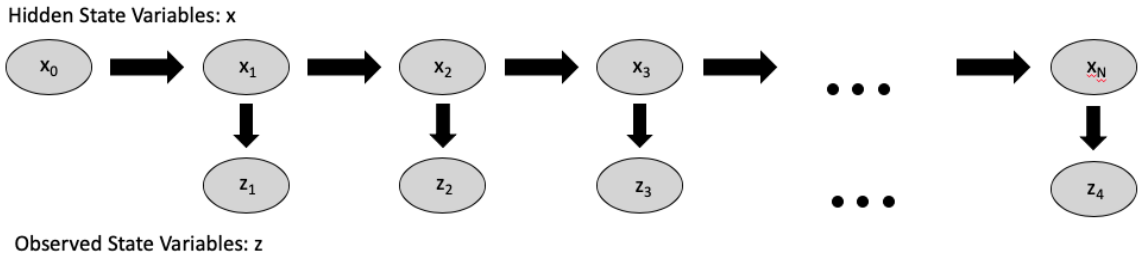


Figure 3.2. Particle Filter Sensor Model Visual Description.

As the UV moves, the true state of the UV is unknown (hidden states) while the observations are known. For each particle, the UV takes the individual particle observation based on the pose of the particle, and compares the particle observation to the current UV observation. By moving the particles with the same control actions of the UV, the particles that correlate better to the observed terrain are closer to the true UV position. The correlation between the particle observation and the UV observation is the particle weight, or likelihood of the position of the UV at the particle. With the new weights, a particle with a larger correlation has the higher probability of being more accurate.

Resampling, implemented in Algorithm 1 lines 8 through 11, draws with replacement, to make a new N-particle set with a new distribution based on the previous weighting values from the sensor model [67]. This changes the distribution of particles from the initial distribution, $\overline{b}(x_t)$, to the posterior distribution, $b(x_t) = \eta p(z_t|x_t^{[n]})\overline{b}(x_t)$, with particles with lower weighting values less likely to be included in the new distribution [67].

3.2.2 Particle Kidnapping

Particle Kidnapping, or particles converging on the wrong location, is a significant problem with the PF. However, this issue can easily be avoided by adding a small percentage

of particles to each particle set which can be justified by the low probability of the PF converging incorrectly [67]. The number of random particles added can be directly related to the average measurement probability [67]:

$$\frac{1}{N}\sum_{n=1}^{N} w_t^{[n]} \approx p(z_t|z_{1:t-1}, u_{1:t}, n). \tag{3.2}$$

Thrun, Burgard, and Fox show an augmented MCL, shown in Algorithm 2, that uses two parameters to model exponential decay rates: slow, α_{slow}, and fast, α_{fast}, in which $0 \leq \alpha_{slow} \ll \alpha_{fast}$ [67]. During the re-sampling process, the probability of generating a random sample can be mathematically defined through $max(0.0, 1 - w_{fast}/w_{slow} > 0)$.

Algorithm 2: Augmented MCL [67]

MCL(B_{t-1}, a_t, z_t, n)
static w_{slow}, w_{fast} $\overline{B}_t = B_t = 0$;
for $n = 1$ *to* N **do**
    $b_t^{[n]}$ = **motion model**$(a_t, b_{t-1}^{[n]})$;
    $w_t^{[n]}$ = **sensor model**$(z_t, b_t^{[n]}, n)$;
    $\overline{B}_t = \overline{B}_t + < b_t^{[n]}, w_t^{[n]} >$;
    $w_{avg} = w_{avg} + \frac{1}{N} w_t^{[n]}$
end
$w_{slow} = w_{slow} + \alpha_{slow}(w_{avg} - w_{slow})$;
$w_{fast} = w_{fast} + \alpha_{fast}(w_{avg} - w_{fast})$;
for $n = 1$ *to* N **do**
    with probability $max(0.0, 1 - w_{fast}/w_{slow})$ **do**
        add random pose to B_t
    else draw i $\in 1, ..., N$with probability $\propto w_t^i$;
        add b_t^i to B_t ;
    end
end
return B_t

3.2.3 Effective Sample Size (ESS) and Resampling

Resampling—changing the distribution of the particles—during every iteration can lead to errors within the PF by introducing a loss of diversity [67]. An important metric of particle diversity, known as effective sampling size (ESS), is shown in equation 3.3 [72], [73], [66]:

$$ESS = \frac{1}{\sum_{i=1}^{N}(\tilde{w}_k^i)^2} \tag{3.3}$$

where $\tilde{w}_k$ is the normalized weight of each particle. Instead of changing the particle distribution during every iteration, the distribution change only needs to occur when $ESS < \beta N$, where $\beta \in (0, 1)$ (typically 0.5) [72] and N is the total number of particles. By choosing only specified times to resample, the particles maintain more diversity and therefore are more likely to converge on the correct position of the UV.

There are many ways to conduct resampling [74]. Commonly used techniques within resampling include multinominal, residual, stratified, systematic, and a parallel resampling method called Metropolis [73], [74]. However, each of these re-sampling techniques leads to higher computational complexity [73], [74]. Within ATAN, there is a finite amount of time and computational cost allowed such that real-time decisions can be made. Therefore, by increasing the cost of the PF, the cost of another ATAN process (shown in Figure 1.1) must be reduced. By resampling only when $ESS < \beta N$ using the resampling methodology shown in Algorithm 1, the PF requires fewer computations, but also maintains a positional uncertainty measurement of less than one meter, as discussed in Section 3.3, which is acceptable for this thesis.

3.2.4 Multimodal Distribution

The PF in Algorithm 2 is solely focused around one Gaussian distribution, but can be improved with minimal computational cost by using multimodal distributions in which multiple modes are used to increase PF robustness. Multi-modal distributions also give greater particle diversity which minimizes the chance of resampling [75], and further, the loss of multimodality [66]. Recent PF techniques utilize multimodal distributions through cluster-based distributions [66], [76], but multimodal distributions come with two issues that need to be solved in order to produce an effective filter: (1) choosing and implementing

each mode and (2) combining modes, if practicable [77].

In this thesis, the multimodal model takes a simple approach by taking the top ten weighted particles,

$$n_{\vartheta} = \underset{n \in \mathbb{N}}{arg\,max}\; w_{n,\,t-1} \tag{3.4}$$

in which $\vartheta \in \mathbf{Z}$ ($\vartheta = 1, 2, 3 \ldots 10$ for this thesis) to be used as the modes for the next distribution. The distribution of particles around particle n_{ϑ} is determined through the weighted mean distances from particle n_{ϑ} to the other top particles,

$$\sigma^2_{\vartheta_n} = \frac{\sum w_{t-1} d_n}{\sum w_{t-1}}. \tag{3.5}$$

Lastly, the probability of each particle occurring within a distribution is proportional to the weight of the particle,

$$p(\vartheta_n | w_{t-1}) = \frac{w_{t-1}}{\sum w_{n,\,t-1}}. \tag{3.6}$$

Using equations 3.4-3.6, the multimodal distribution can be employed.

3.2.5 Final Model

Combining mitigations for particle kidnapping and ESS, and adding in the multi-modal distribution culminate in the final MCL algorithm used for ATAN, as shown in Algorithm 3. While Lines 1-11 are the same as in Algorithm 2 [67], particle distribution changes (Lines 12-22) only occurs when the ESS is less than βN.

Algorithm 3: Final MCL

Final MCL(B_{t-1}, a_t, z_t, n)
static w_{slow}, w_{fast}
$\overline{B}_t = B_t = 0$
for $n = 1$ *to* N **do**
    $b_t^{[n]}$ = **motion model**$(a_t, b_{t-1}^{[n]})$
    $w_t^{[n]}$ = **sensor model**$(z_t, b_t^{[n]}, n)$
    $\overline{B}_t = \overline{B}_t + < b_t^{[n]}, w_t^{[n]} >$
    $w_{avg} = w_{avg} + \frac{1}{N} w_t^{[n]}$
end
$w_{slow} = w_{slow} + \alpha_{slow}(w_{avg} - w_{slow})$
$w_{fast} = w_{fast} + \alpha_{fast}(w_{avg} - w_{fast})$
if $ESS < \beta N$ **then**
    for $n = 1$ *to* N **do**
        $\varphi \in \mathcal{U}[0, 1]$;
        if $max(0.0, 1 - w_{fast}/w_{slow}) < \varphi$ **then**
            add **random pose** to B_t
        else
            draw i from **multi-modal sample** $\mathcal{N}(n_\vartheta, \sigma^2_{\vartheta_n})$
            add b_t^i to B_t
        end
    end
else
    draw i from **multi-modal sample** $\mathcal{N}(n_\vartheta, \sigma^2_{\vartheta_n})$
    add b_t^i to B_t
end
return B_t

3.3 Particle Filter Implementation and Analysis

The Algorithm 3 PF was implemented and analyzed using a full, accurate *a priori* map through a map model, motion model, sensor model, and establishing PF parameters with a maximum AUV uncertainty equal to one meter.

3.3.1 Map

A 255m $\times$ 255m bathymetry map (Figure 3.3) provided by California State University, Monterey Bay (CSUMB) [78] was validated using the REMUS prior to the simulation.

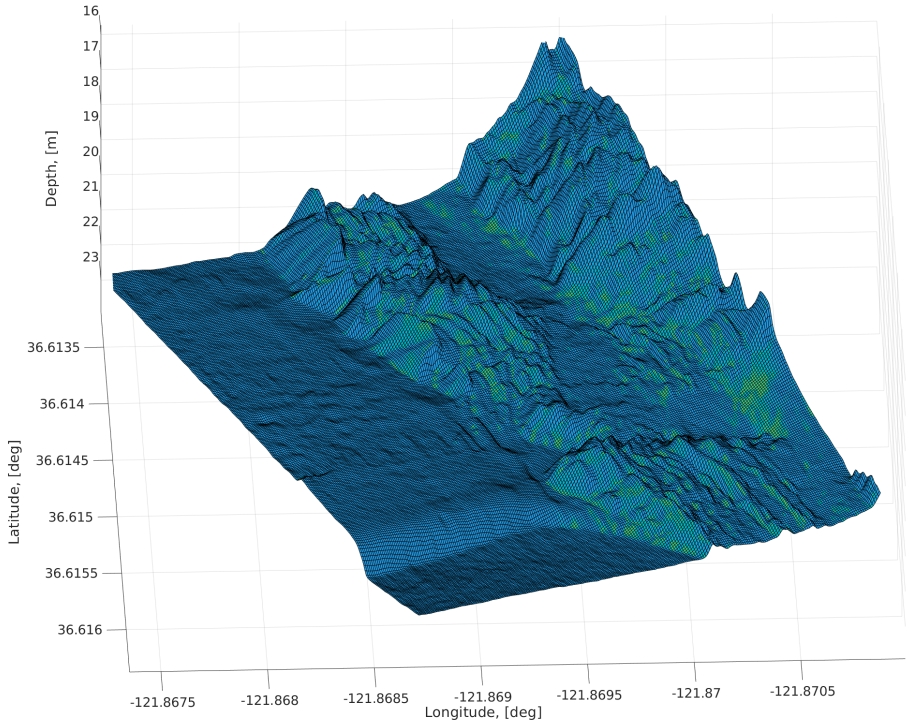


Figure 3.3. Sea Floor Bathymetry Provided by CSUMB, Verified by REMUS. Adapted from [78].

Shown in Figure 3.3, the one meter resolution map area was chosen due to its combination of flat, rough, and changing terrain that also was in the required depth range of the REMUS.

3.3.2 Motion Model

The simulated motion model utilized the same model for vehicle position updates, but also incorporated a noise aspect. The actual vehicle motion model uses a Kalman Filter,

$$\mathbf{x} = A\mathbf{v} + \mathbf{x}_0 \tag{3.7}$$

in which A is a transition matrix, $\mathbf{v}$ is the AUV velocity vector, and $\mathbf{x}_0$ is the original position vector. The PF motion model added two noise aspects to equation 3.7 to account for position error and velocity error, with noise power equal to 0.2 and 0.3, respectively.

3.3.3 Sensor System Modeling

The REMUS vehicle has two predominant sensor systems: BlueView MB2250 downward looking sonar and ARC Scout side-scan sonar. The BlueView MB2250 sonar sensor gives an image space, linear twenty meter swath, directly under and perpendicular to the vehicle. The ARC Scout side-scan sonar also outputs in image space, with a range of thirty feet [17]. BlueView MB2250 sonar sensor and ARC Scout side-scan sonar image spaces are shown in Figure 3.4.

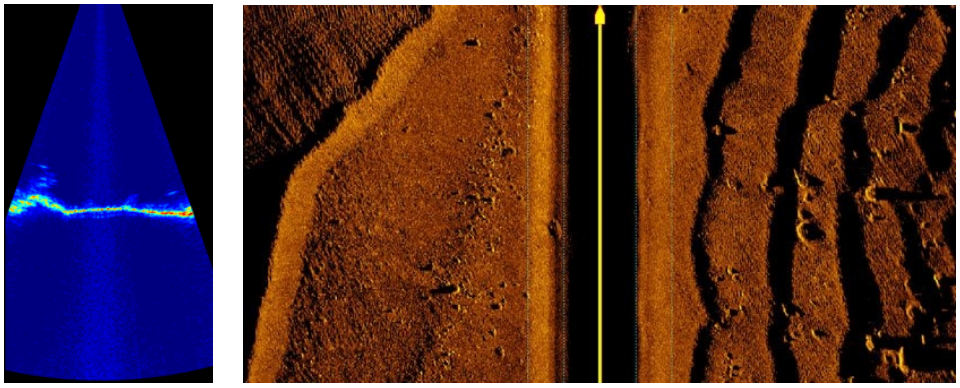
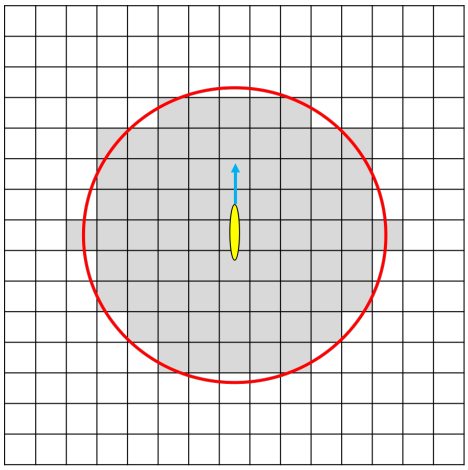
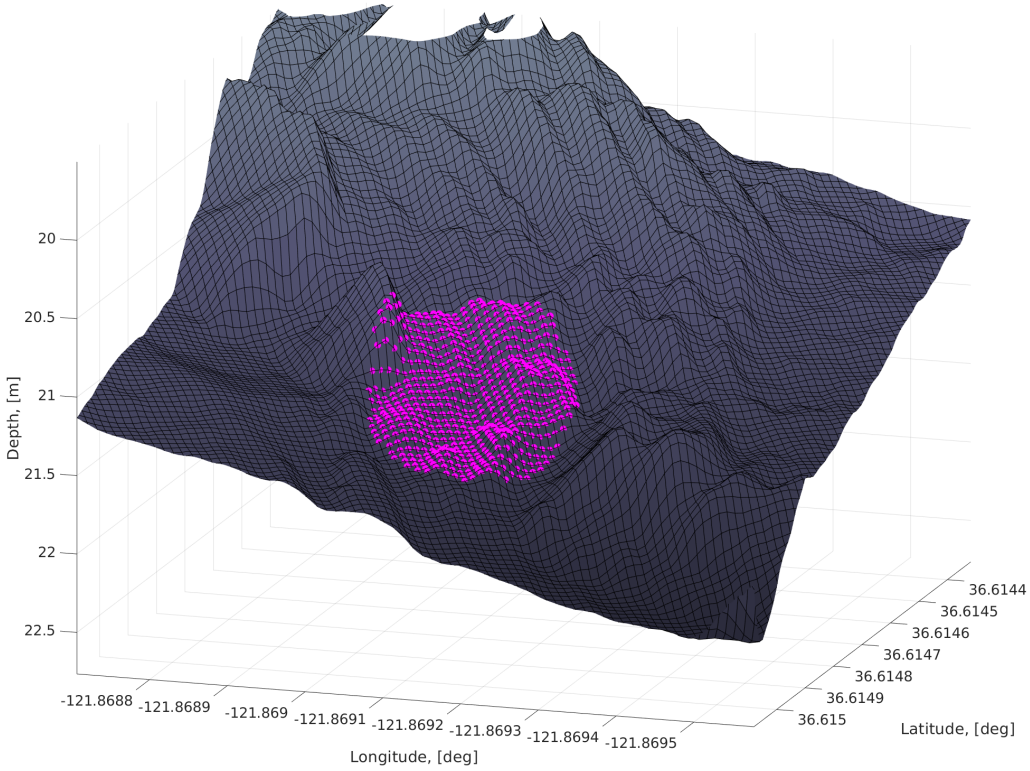


Figure 3.4. Sensor System Model. (left) Image Space BlueView MB2250. (right) Image Space ARC Scout Side-Scan Sonar. Source: [13].

Through computer vision, both the BlueView MB2250 image and side-scan sonar image can be transformed into a spatial grid, discussed in future work not completed with this thesis. In this thesis we utilize a simplification of the sensor system that emulates the combination of both sensors as a ten-meter circular swath around the vehicle, as shown in Figure 3.5.



(a)



(b)

Figure 3.5. Sensor Model. (a) Top-down view of REMUS and sensor of 10m radius. (b) View of 10m sensor model over terrain.

The weighting of each particle can be determined through the correlation method [26]. From the full, accurate map, the sensor model uses the correlation between the particle observation, with noise power equal to 0.2, and the AUV sensor, shown in equation 3.8:

$$w_t^{[n]} = \frac{\mathbb{E}[(s_p - m_p)(s_a - m_a)]}{\sigma_{s_p}\sigma_{s_a}} \tag{3.8}$$

in which the particle sensor matrix s_p,particle sensor mean m_p, AUV sensor matrix s_a, AUV sensor mean m_a, particle sensor standard deviation, σ_{s_p} and AUV sensor standard deviation σ_{s_a} are known or can be calculated.

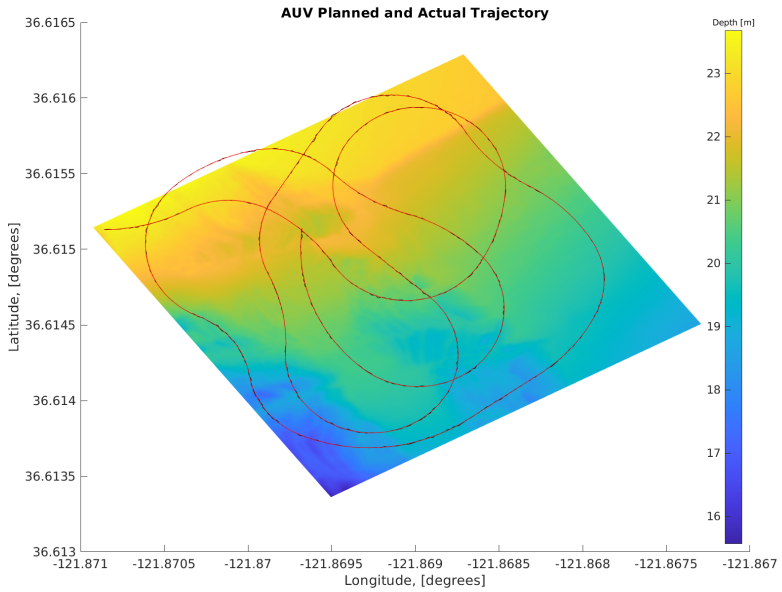
3.3.4 Particle Filter Parameters

The PF in Algorithm 3 was implemented in MATLAB for position estimation of the AUV used in this thesis over a 2000 meter and 8000 meter trajectory to gauge accuracy and also evaluate recovery from kidnapping using the constants shown in Table 3.1.

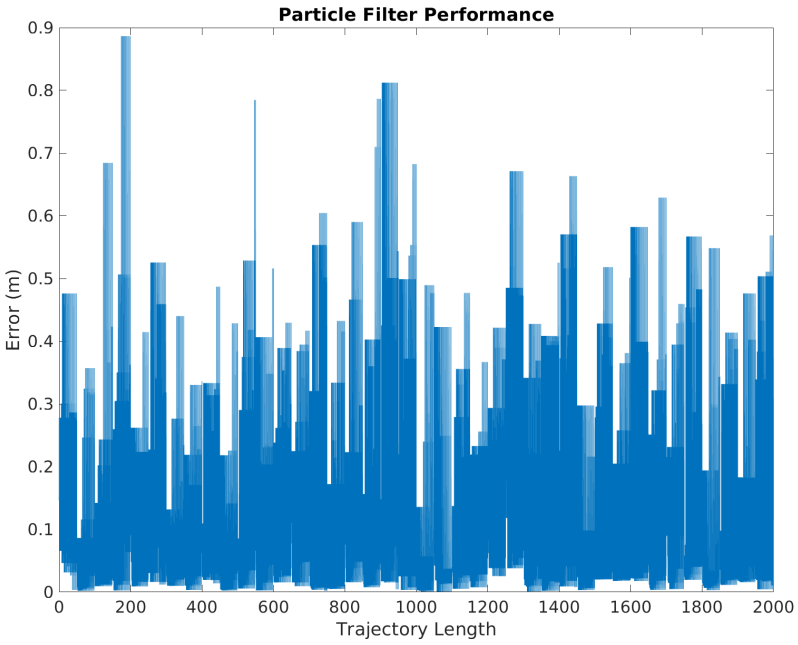
Table 3.1. Particle Filter Constants used in Implementation.

PF Constants			
N	1000	β	0.5
w_{slow}	0.5	α_{slow}	0.1
w_{fast}	0.5	α_{fast}	0.9

Over a short term trajectory length of 2000m (Figure 3.6) and long term trajectory length of 8000m (Figure 3.7), the PF maintains an error of less than one meter. Based on the resolution of the map, the simulation the PF is able to localize the AUV position to within a single $1m \times 1$m cell.

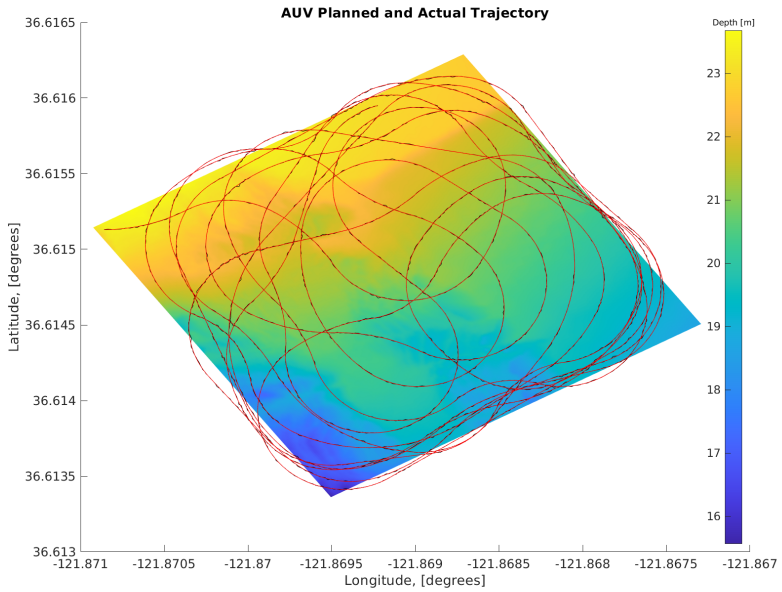


(a)

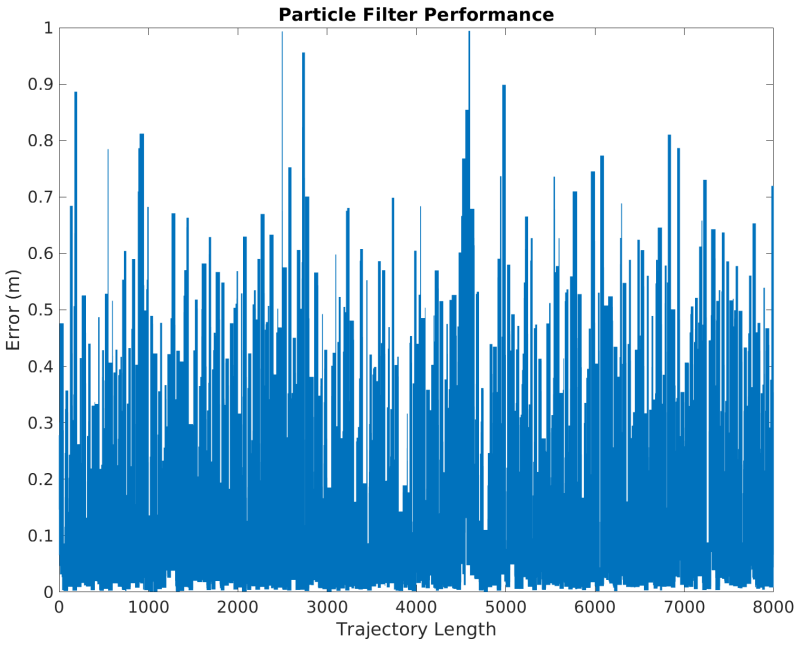


(b)

Figure 3.6. Particle Filter Performance Over 2000m Trajectory Length Using Algorithm 3. (a) AUV trajectory in red, PF trajectory in black (completely overlapped by red trajectory) shows small degree of error throughout the AUV motion. (b) PF short term error in meters along trajectory.



(a)



(b)

Figure 3.7. Particle Filter Performance Over 8000m Trajectory Length Using Algorithm 3. (a) AUV trajectory in red, PF trajectory in black (completely overlapped by red trajectory) shows small degree of error throughout the AUV motion. (b) PF long term error in meters along trajectory.

The PF was also tested in regards to particle kidnapping. The estimated position of the AUV was moved to a random location once the AUV traveled 2000 meters, as shown in Figure 3.8, to simulate the response of the PF converging on an incorrect position and to determine the recovery of the PF. The PF error before, during, and after kidnapping is shown in Figures 3.9-3.11.

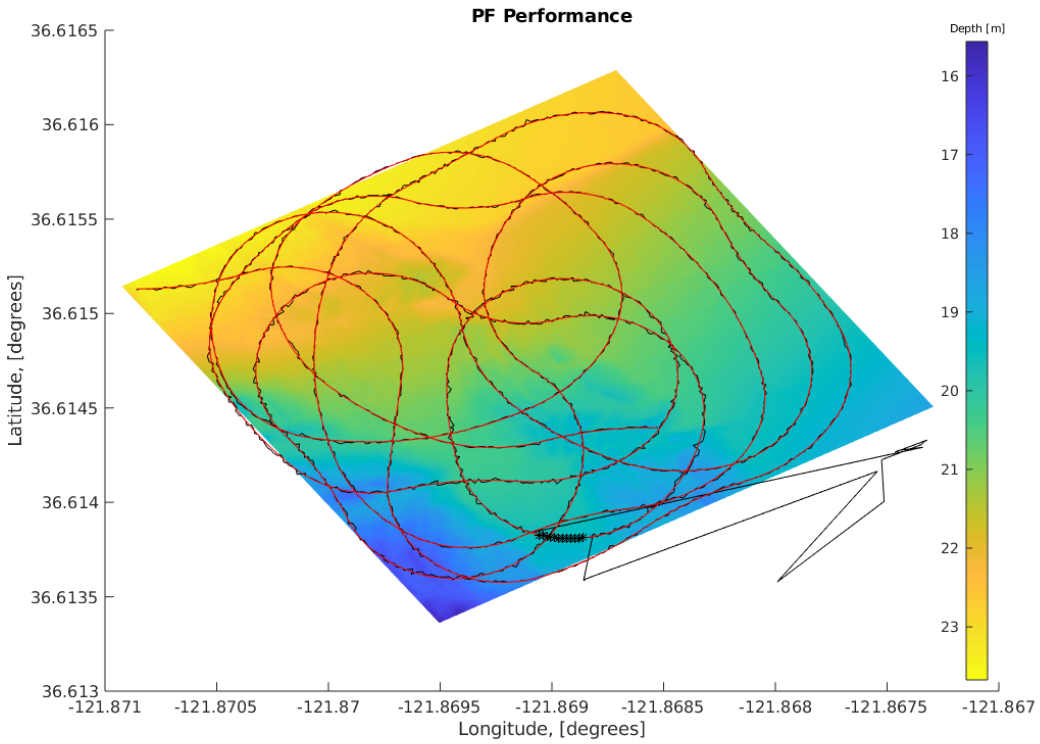


Figure 3.8. Particle Filter Response to Particle Kidnapping: (red) AUV Trajectory (black) PF Position Estimate.

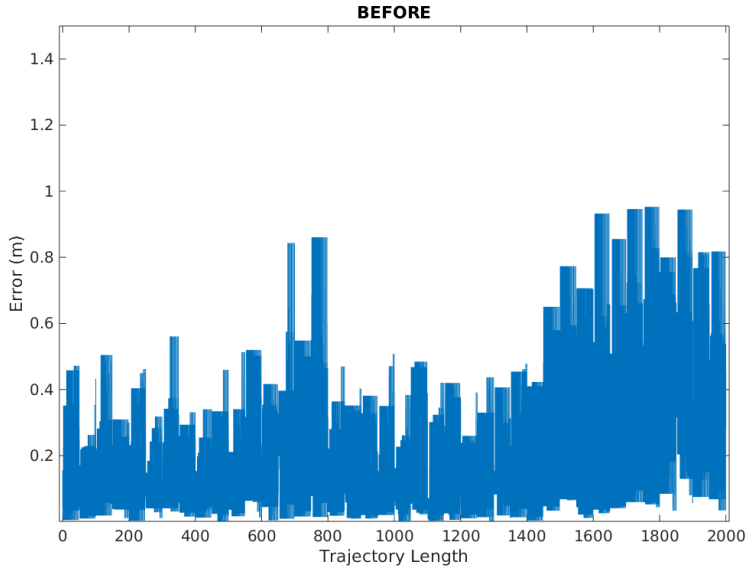


Figure 3.9. Particle Filter Error Prior to Particle Kidnapping.

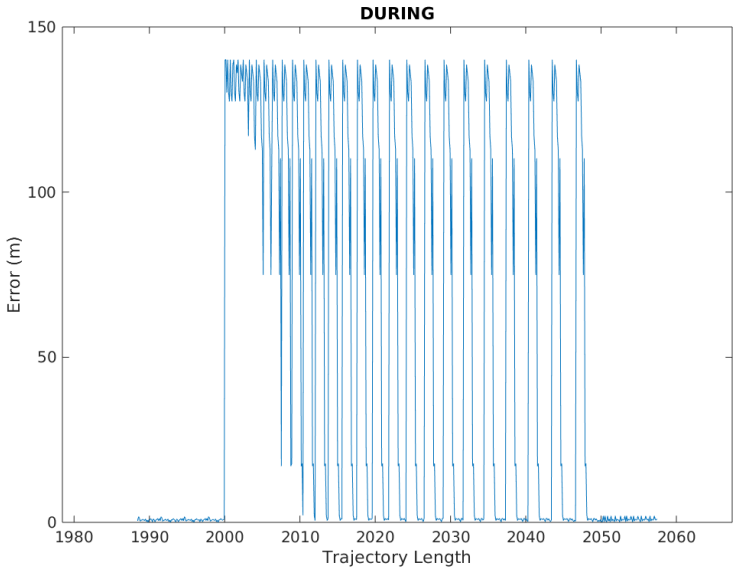


Figure 3.10. Particle Filter Error Response and Recovery to Particle Kidnapping.

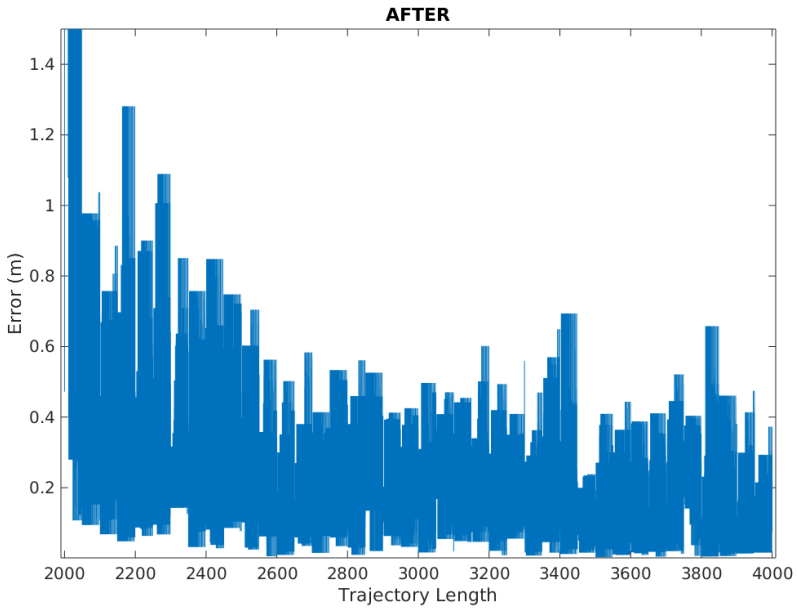


Figure 3.11. Particle Filter Error After Recovery from Particle Kidnapping.

As seen in Figures 3.9-3.11, the PF error increases drastically to ≈ 140 meters (Figure 3.10) when the AUV is moved at a path length of 2000m. However, the PF recovers to within 1.4 meters of positional error after the AUV travels 50 meters (Figure 3.11), and within 1 meter of positional error after around 230 meters of distance traveled, successfully converging on the correct AUV position.

3.4 Particle Filter Conclusion

The AUV position estimator, the PF, is able to localize the position of the AUV to within ±1m over an 8000m trajectory using an *a priori* map. For comparison, the INS discussed in Table 1.1, would generate an error of 40m with the same trajectory. The high success of the PF can be attributed to the simplifications made to the sensor system through the employment of a circular sensor. However, PF has been proven to maintain one meter of error in real time within TAN, shown in [33], giving the assumption results credibility.

The PF increases the ability to estimate position by maintaining 1m or less of positional uncertainty over an 8000m trajectory which achieves the goal of minimizing positional error listed in Section 1.1. But the PF can still be improved. Unfortunately, the PF is only as

good as the map that the AUV creates. The map created and used has a resolution of 1m which makes it difficult for the AUV to localize, and maintain localized, to an error of less than 1 meter. However, the PF is easily scaleable, and therefore the sensor accuracy will be the limiting factor as opposed to the map accuracy when making a map (not in simulation).

Lastly, the outstanding PF performance is due to two factors. First, the amount of information within the map (rough, uneven bathymetry) contributes to the PF easily determining the AUV position (with noise sensor noise having a minimal affect). Second, the PF success is due to the *a priori* map, which during ATAN the PF will no longer have. Instead, the input to the PF will only be the AUV generated map, which comes full circle back to the *exploration-exploitation* dilemma.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Exploration

Many machine learning techniques use the following template: sample from a distribution, derive multiple IID random variables, solve an optimization problem to build a model, and make predictions based on the model. This chapter builds, maintains, and utilizes an optimal spatial estimate model of a random field, mathematically depicted as a Gaussian Process Model (GPM), to conduct *exploration* planning. The GPM provides a mean estimate and covariance estimate, the later of which represents the uncertainty measure quantifying *exploration*. Through the GPM, *exploration* can be defined as the reduction of uncertainty in the GPM by either exploring new, unmapped areas or by decreasing the uncertainty in high uncertainty areas through re-visitation. This chapter discusses the GPM and describes how the GPM can be used for path planning in a real-time process.

4.1 Gaussian Process Model

A GP is defined by Rasmussen and Williams as multiple Gaussian random variables [79]. For modeling the terrain, a bounded rectangular $N \times M$ region is subdivided into $n \times m$ cells where each cell $\mathbf{m}$ can be fully described as a Gaussian distribution with a mean μ and a covariance $k(\mathbf{m}, \mathbf{m}')$, $\sim N(\mu, k(\mathbf{m}, \mathbf{m}'))$. This distribution is based on the center of the cell and is known as the point of interest (POI) of the cell.

In order to employ a GPM as a spatial estimator of terrain, three assumptions and two simplifications need to be made.

Assumptions

The following assumptions are associated with using the GPM:

- There exists a function that describes the exact terrain. To minimize computational complexity, linear interpolation is used , $\mathbf{z} = \mathbf{w}\mathbf{x} + \epsilon$, in which $\mathbf{z}$ is the observed target value, $\mathbf{w}$ is a vector of weighted parameters, $\mathbf{x}$ is the location, and ϵ is IID additive noise with $\sim \mathrm{N}(0, \sigma_n^2)$) [79].

- All cells within the bounded region adhere to a second-order stationarity assumption: the mean and covariance of each individual cell do not vary significantly over time and space, and the covariance between each cell is only a function of the relative distance between two cells.
- The covariance relationship with distance can be described as isotropic—uniform in all directions—with the covariance inversely proportional to distance. As the distance increases, the similarities within the terrain tend to decrease, leading to a lower covariance.

Simplifications

The following simplifications are associated with using the GPM:

- The information gain is a function of the forward looking sensor only on the direct path of the vehicle when calculating a potential trajectory for exploration. This constraint reduces the computational complexity of the GPM by not determining the information gain over every cell touched by the sensor but rather only those which the UV drives directly over.
- The predictive distribution (mean prediction and covariance) of the GPM are calculated for one test point on the UV path, using a random sampling of nearby surrounding points. By only having one test point, the computational complexity of the GPM decreases, and the mean and covariance of the cells can still be adequately modeled.

Employing the GPM

Taking a function-space view approach, the Gaussian process can be defined as [79]

$$f(\mathbf{m}) \sim \mathcal{GP}(\mu(\mathbf{m}), k(\mathbf{m}, \mathbf{m}')), \tag{4.1}$$

in which the mean function $\mu(\mathbf{m})$ and covariance function $k(\mathbf{m}, \mathbf{m}')$ can be expressed using [79]

$$\begin{aligned} \mu(\mathbf{m}) &= \mathbb{E}[f(\mathbf{m})] \\ k(\mathbf{m}, \mathbf{m}') &= \mathbb{E}[(f(\mathbf{m}) - \mu(\mathbf{m}))(f(\mathbf{m}) - \mu(\mathbf{m}'))]. \end{aligned} \tag{4.2}$$

The GPM can be written through a number of input points M as [79],

$$\begin{bmatrix} \mathbf{z} \\ \mathbf{f}_* \end{bmatrix} = N\left(\begin{bmatrix} \mu \\ \mu_* \end{bmatrix}, \begin{bmatrix} K(M,M) + \sigma_n^2 I & K(M,M_*) \\ K(M_*,M) & K(M_*,M_*) \end{bmatrix}\right) \tag{4.3}$$

in which the n explored area observations $\mathbf{z}$ are represented through a mean estimate μ and a $n \times n$ covariance matrix $K(M,M)$ [79]. The n_* unexplored area observations $\mathbf{f}_*$ are represented with a mean estimate μ_* and $n_* \times n_*$ covariance matrix $K(M_*,M_*)$ [79]. The $n \times n_*$ covariance matrix $K(M,M_*)$ represents the n explored estimates and the n_* unexplored areas [79].

4.1.1 The Matérn Covariance Function

The covariance matrix can be calculated from commonly used models such as spherical, squared exponential, Matérn, or any other covariance function such that the covariance matrix is positive semi-definite [80]. In this thesis, the Matérn model is used. It is generally considered to be the best choice for representing spatial differences between two points [81],

$$K(M,M_*) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}|M-M_*|}{\lambda}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}|M-M_*|}{\lambda}\right) \tag{4.4}$$

in which ν and λ are non-negative parameters, K_v is the modified Bessel function of the second kind, and $\Gamma(\nu)$ is the Gamma function.

4.1.2 Variogram

The parameters, ν and λ, used for the Matérn covariance function (equation 4.4) can be determined through the use of a semi-variogram, $\gamma(\mathbf{m},h)$ or variogram, $2\gamma(\mathbf{m},h)$, [80]

$$2\gamma(\mathbf{m},h) = E[f(\mathbf{m}) - f(\mathbf{m}+h)]^2, \tag{4.5}$$

in which $\mathbf{m}$ is the position in the map and h is the distance from location $\mathbf{m}$. The variogram and covariance functions are conceptually related, with the variogram representing the dissimilarities of the terrain and the covariance depicting the similarities of the terrain. Further, both the covariance and variogram maintain an underlying hypothesis that the variogram and the covariance function only depend on the distance h between two points,

which allows for the covariance $C(h)$ and variogram $2\gamma(h)$ to be expressed as only a function of distance h. The covariance-variogram relationship is further described in Appendix A, with the resulting mathematical relationship between the two represented as [80]

$$C(h) = \gamma(\infty) - \gamma(h). \tag{4.6}$$

The variogram, shown in Figure 4.1, can be defined through the properties of sill, range, and nugget,

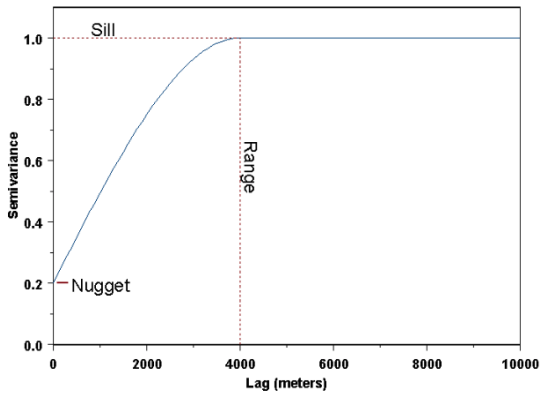


Figure 4.1. Variogram Terminology: Range, Sill, and Nugget. Source: [82].

in which the sill is the maximum value of the variogram plateau [82], the nugget describes the measure of dissimilarities at close distances (located at the y-intercept) [82], and the range is the distance between the location of the nugget and the location of the start of the sill [82].

The centralized hypothesis of the variogram—the variogram only depends on the separation between two points—allows for the creation of the variogram estimator [80],

$$2\gamma^*(h) = \frac{1}{N(h)} \sum_{i=1}^{N(h)} [f(\mathbf{m}_i) - f(\mathbf{m}_i + h)]^2, \tag{4.7}$$

in which $N(h)$ is the number of data points between $f(\mathbf{m}_i)$ and $f(\mathbf{m}_i + h)$.

Applying the variogram methodology to the bathymetry data used in this thesis, the initial spatial relationship between depth and distance is determined. Using equation 4.7 a variogram is created, as shown in Figure 4.2.

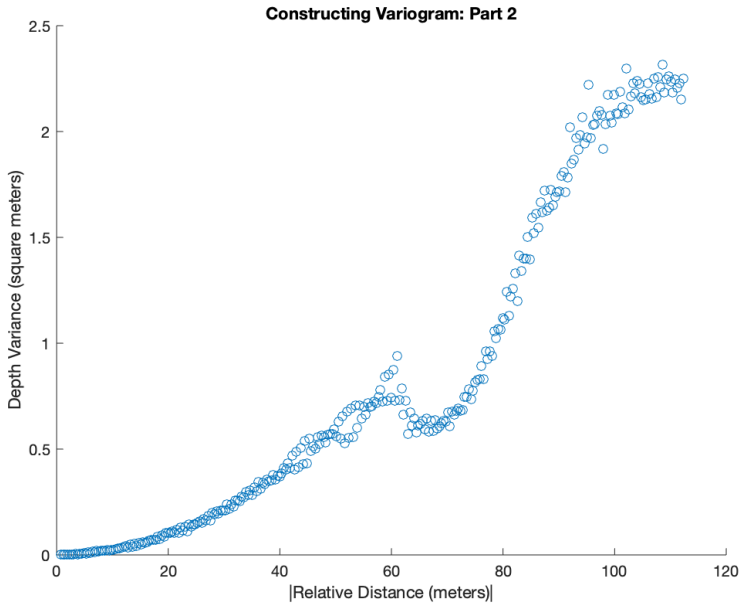


Figure 4.2. Variogram Estimated from Experimental Data using 100 Partitions.

4.1.3 Fitting Matérn Covariance Function to the Variogram

The Matérn covariance function, equation 4.4, has two non-negative parameters: ν and λ. Using the relationship between the variogram and covariance, equation 4.6, and the variogram, Figure 4.2 (right), the covariance as a function of distance can be calculated and fit using the the Matérn function through a least squares method, producing Figure 4.3.

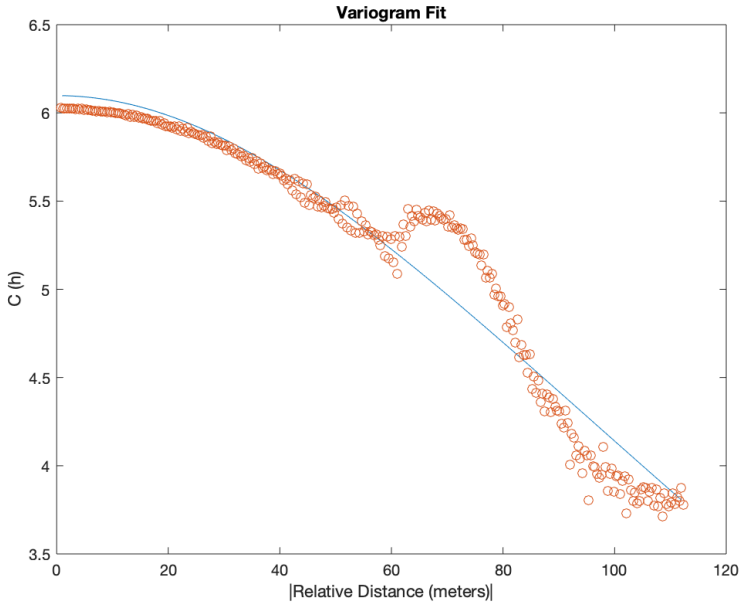


Figure 4.3. Fitting the Variogram using the Matérn Function. (red) Variogram, (blue) Matérn Function.

4.1.4 Lagrange Multiplier

With the Matérn covariance function, the predictive distribution has a mean and a covariance which can be written as [79]

$$\bar{f}_* = K(M_*, M)[K(M, M) + \sigma_n^2 I]^{-1}\mathbf{z} \tag{4.8}$$

$$\sigma_*^2 = K(M_*, M_*) - K(M_*, M)[K(M, M) + \sigma_n^2 I]^{-1}K(M, M_*). \tag{4.9}$$

To ensure a minimum bias, a Lagrange multiplier can be used [80] to subtract out the local mean from $\bar{f}_*$. To simplify equations 4.8 and 4.9, a short form will be used such that $K(M_*, M) = [K_*]$, $K(M_*, M_*) = [K_{**}]$, $K(M, M) = [K]$, and $[\lambda]^T = K(M_*, M)[K(M, M) + \sigma_n^2 I]^{-1}$. Equations 4.8 and 4.9 can be rewritten in the matrix form [80],

$$\bar{f}_* = [\lambda]^T[z] \tag{4.10}$$

$$\sigma_*^2 = [K_{**}] - [\lambda]^T[K_*], \tag{4.11}$$

in which $[z]$ is the sampled depths, and $[\lambda]^T$ is the transpose of the weighting matrix $[\lambda]$.

For this thesis, the GPM uses a simplification to have one test point calculated from surrounding points. The system of equations for spatial optimization with the Lagrange multiplier κ_L can be written as [80],

$$[K][\lambda] = [K_*], \tag{4.12}$$

in which

$$[K] = \begin{bmatrix} C(M_1, M_1) & \dots & C(M_1, M_n) & 1 \\ \vdots & \ddots & \vdots & 1 \\ C(M_n, M_1) & \dots & C(M_n, M_n) & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}, \tag{4.13}$$

$$[\lambda] = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ -\kappa_L \end{bmatrix}, \tag{4.14}$$

and

$$[K_*] = \begin{bmatrix} C(M_1, M_*) \\ \vdots \\ C(M_n, M_*) \\ 1 \end{bmatrix}, \tag{4.15}$$

in which $C(M_1, M_2)$ is the covariance determined from Figure 4.3 based on the distance between M_1 and M_2.

Solving the spatial optimization (equation 4.12) [80],

$$[\lambda] = [K]^{-1}[K_*] \tag{4.16}$$

can then be used to solve for the predictive distribution mean and covariance, reiterated below [79]:

$$\bar{f}_* = [\lambda]^T[z] \tag{4.17}$$

$$\sigma_*^2 = [K_{**}] - [\lambda]^T[K_*]. \tag{4.18}$$

4.2 Constructing A Real-Time Gaussian Process Model

Normally the covariance function is calculated through a batch process. In this thesis, the covariance determination occurs when the terrain is learned through a Bayesian update [83]: using the Bayesian prior and the current observations to calculate an updated mean and covariance.

4.2.1 Establishing the Prior Matérn Covariance Function

The prior Matérn covariance function uses the isotropic assumption to show the similarity of the terrain as a function of distance, regardless of direction. Through second-order stationarity, the covariance between two closer areas gives higher similarities, while further distances provide lower covariance values. The covariance relationship decreases, with negative concavity, as a function of distance [83]. Therefore, the initial Matérn covariance function can be described through

$$\mathbf{z} = C(0) - \tau h^2 \tag{4.19}$$

in which $C(0)$ is the initial covariance, h is the distance between two measurements, $\tau = C(0)/r^2$, and r is the range.

Additionally, the initial Matérn covariance function is expected to provide an inadequate fit, with the fit being continuously improved as *exploration* occurs.

4.2.2 Update Matérn Covariance Function

As exploration occurs, the Matérn covariance function is adjusted using the known regions of the GPM. Using a Markov Chain Monte Carlo (MCMC) approach, the posterior semi-

variogram, $\hat{\gamma}$, can be determined [84],

$$\hat{\gamma}(h) = \frac{1}{V - W} \sum_{v=W+1}^{V} \gamma_v(h), \tag{4.20}$$

in which $\gamma_v(h)$ are prior semi-variograms, V is the length of the Markov Chain, and W is the amount of initial iterations to be discarded [84].

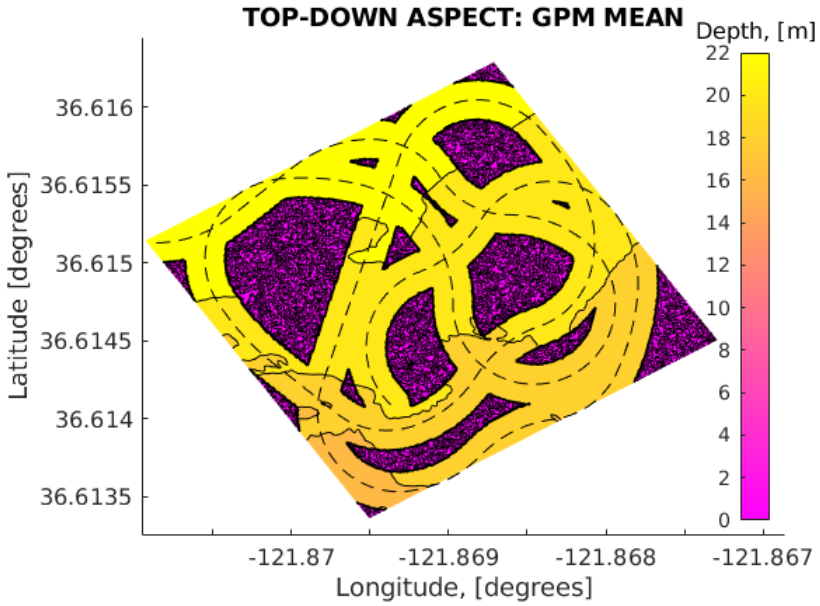
Using the variogram-covariance relationship, equation 4.6, converts equation 4.20 to

$$\hat{C}(h) = \hat{\gamma}(\infty) - \hat{\gamma}(h), \tag{4.21}$$

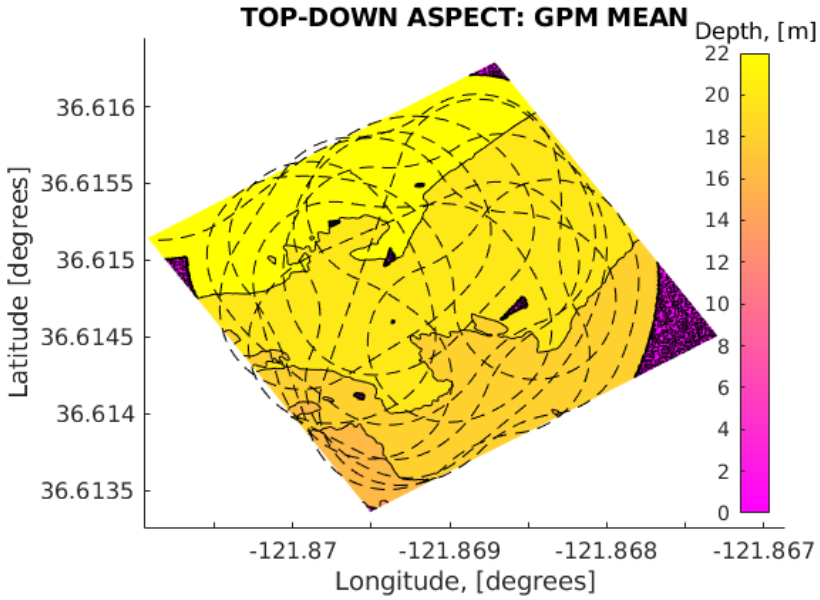
a posterior covariance function—computed only using explored areas—that can be modeled using the Matérn covariance function. As *exploration* occurs, the Matérn covariance function becomes more refined, allowing for better mean and covariance predictive distributions (equations 4.17 and 4.18) to be calculated in real time.

4.2.3 GPM Update

As *exploration* occurs, the GPM is updated through the predictive distributions (mean and covariance), which change as the AUV path length increases and *exploration* occurs. The initial zero mean, constant covariance model (in this thesis we use an initial constant covariance equal to four) is updated to the predicted mean and covariance, as illustrated in Figure 4.4 and Figure 4.5.



(a) Mean 2000m AUV path length.

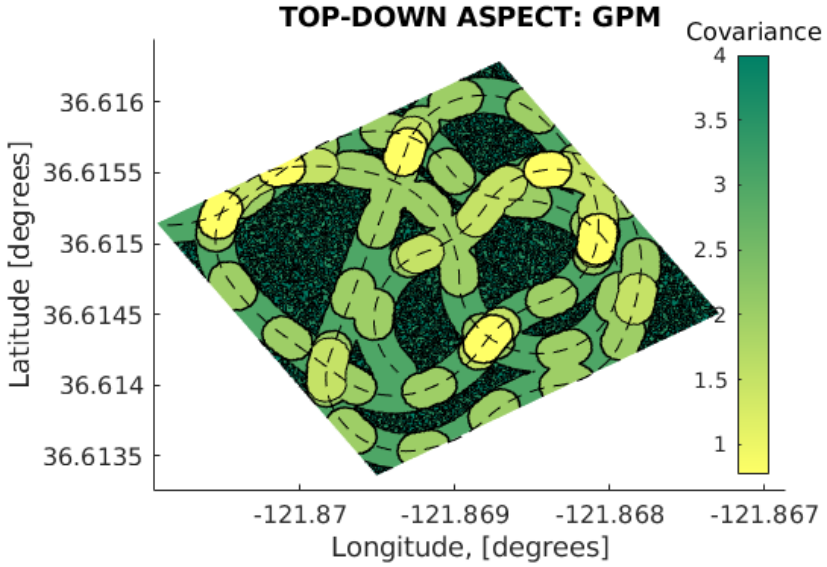


(b) Mean 6000m AUV path length.

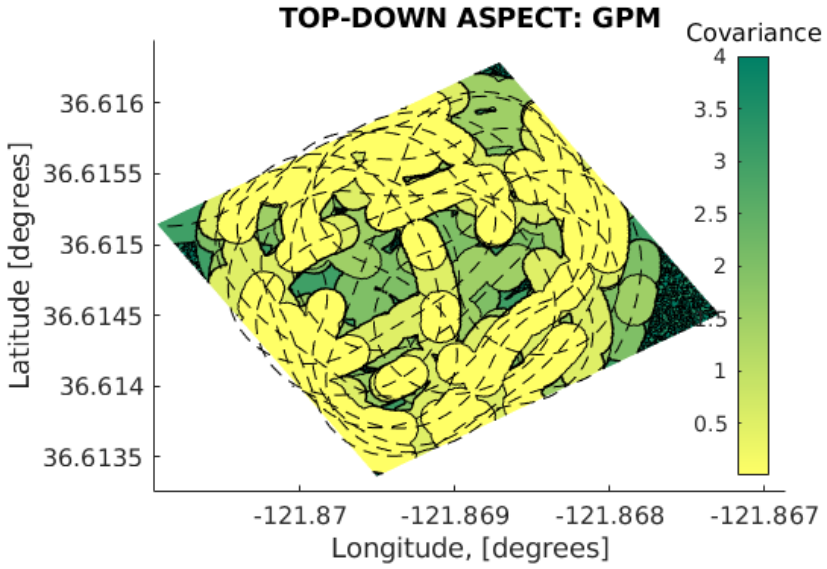
Figure 4.4. Gaussian Process Model Predictive Mean Updates.

The mean model (Figure 4.4) starts with all areas equal to zero. As the vehicle discovers new areas, each explored area is updated through the predictive mean (equation 4.17). The

vehicle track is shown as a dashed line, with unexplored areas shown in purple and explored areas shown in shades of yellow. Additionally, as the vehicle explores more area (Figure 4.4b), the mean map more resembles the true bathymetry of the area.



(a) Covariance 2000m AUV path length.



(b) Covariance 6000m AUV path length.

Figure 4.5. Gaussian Process Model Predictive Covariance Updates.

The covariance model (Figure 4.5) is initialized to a high uncertainty of the bathymetry,

$\sim N(4, \sigma_n^2)$. As the vehicle explores or re-visits areas, the uncertainty of areas visited can be reduced using the predictive covariance (equation 4.9), shown in Figure 4.5a. The vehicle track is shown in a black dashed line, and areas of vehicle path intersection results in lower uncertainty (covariance) of the bathymetry. As more of the map is explored (Figure 4.5b), the uncertainty about the map decreases.

As the AUV explores more area, the GPM will provide a more accurate representation of the terrain, further resulting in a better ability to conduct *exploration*.

4.3 Kullback Leibler Divergence

The Kullback Leibler Divergence (KLD) computes the difference of mutual information between two probability distributions [85]. In this thesis, the KLD is used to determine the *exploration* score by calculating the information gain for each possible trajectory. Using the GPM mean and covariance predictions for each cell the UV drives over, the KLD can then evaluate each trajectory to determine the best course. The KLD between two distributions, p($\mathbf{m}$) and q($\mathbf{m}$), is defined as can be computed using equation 4.22 [85]:

$$KL(p||q) = \int p(\mathbf{m}) \log\Big(\frac{p(\mathbf{m})}{q(\mathbf{m}}\Big) d\mathbf{m}. \tag{4.22}$$

The KLD can be simplified by expressing both $p(\mathbf{m})$ and $q(\mathbf{m})$ as Gaussian distributions: $p(\mathbf{m})$ with mean μ_1 and variance σ_1^2 and $q(\mathbf{m})$ with mean μ_2 and variance σ_2^2. Simplifying equation 4.22, as described in Appendix B, yields equation 4.23:

$$KL(p||q) = \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}. \tag{4.23}$$

The KLD is normalized to ensure an *exploration* score per time-step $\in [0, 1]$,

$$KL_{Norm}(p||q) = 1 - e^{-KL(p||q)} \tag{4.24}$$

which is then averaged over the entire trajectory to give the combined *exploration* score.

4.4 Exploration Analysis

The *exploration* analysis is broken into two parts: evaluating the GPM at various stages of the UV path and evaluating future trajectories to ensure *exploration* is maximized.

4.4.1 Evaluating the GPM

As the AUV conducts *exploration*, the GPM converges to the true terrain, as seen in Figure 4.4 and Figure 4.5. The AUV begins with a zero-mean, constant covariance model over the entire area and updates the sensor area with the prediction mean and covariance as *exploration* occurs. The reduction in spatial covariances can be seen in Figure 4.6, where the lowest covariances correspond to areas of intersection with the previous path of the AUV. As the AUV path length increases from Figure 4.6 to 4.7 to 4.8, the covariance of the map decreases.

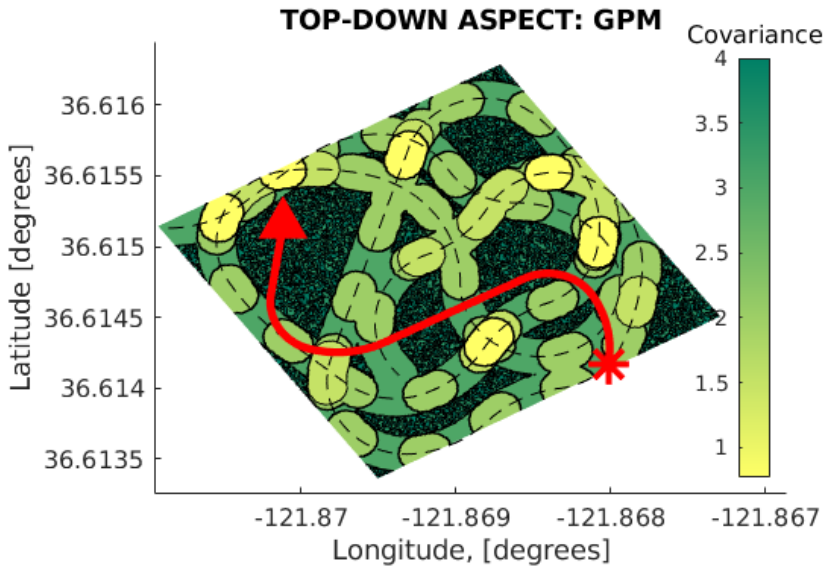


Figure 4.6. Top-Down Covariance GPM Prediction and AUV path (black dashed line) for AUV Path Length of 2000m.

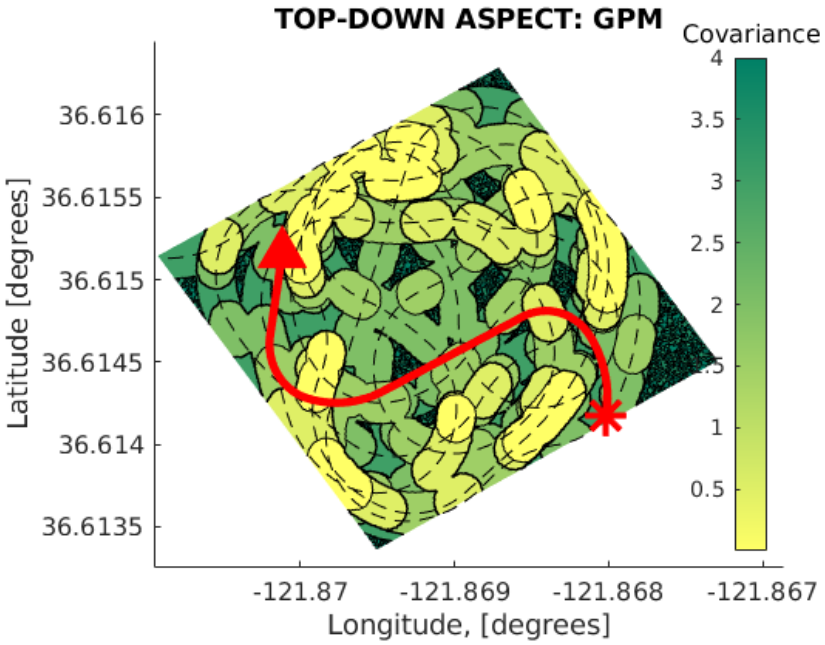


Figure 4.7. Top-Down Covariance GPM Prediction and AUV path (black dashed line) for AUV Path Length of 4000m.

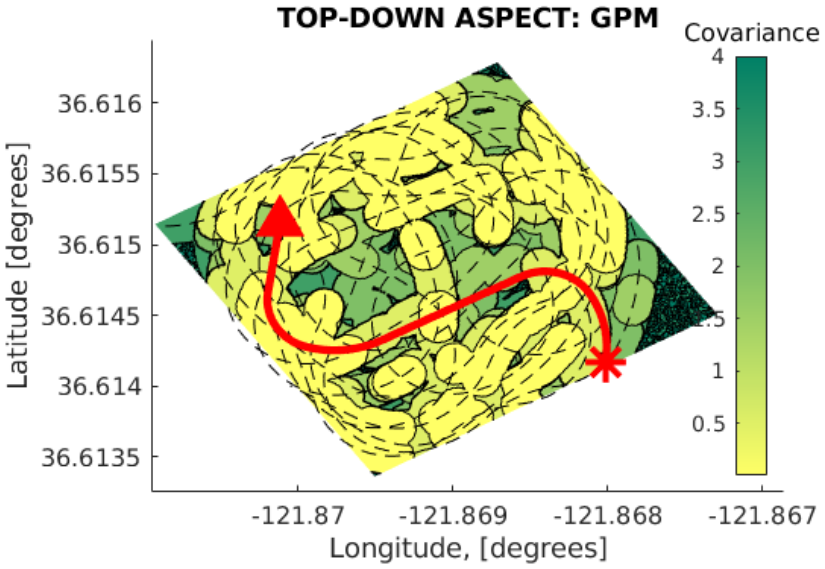


Figure 4.8. Top-Down Covariance GPM Prediction and AUV path (black dashed line) for AUV Path Length of 6000m.

As exploration occurs, the GPM changes to more resemble the true bathymetry of the map.

Looking horizontally at the GPM, depicted in Figure 4.9, shows the mean and covariance of the GPM along the red trajectory shown in Figures 4.6-4.8.

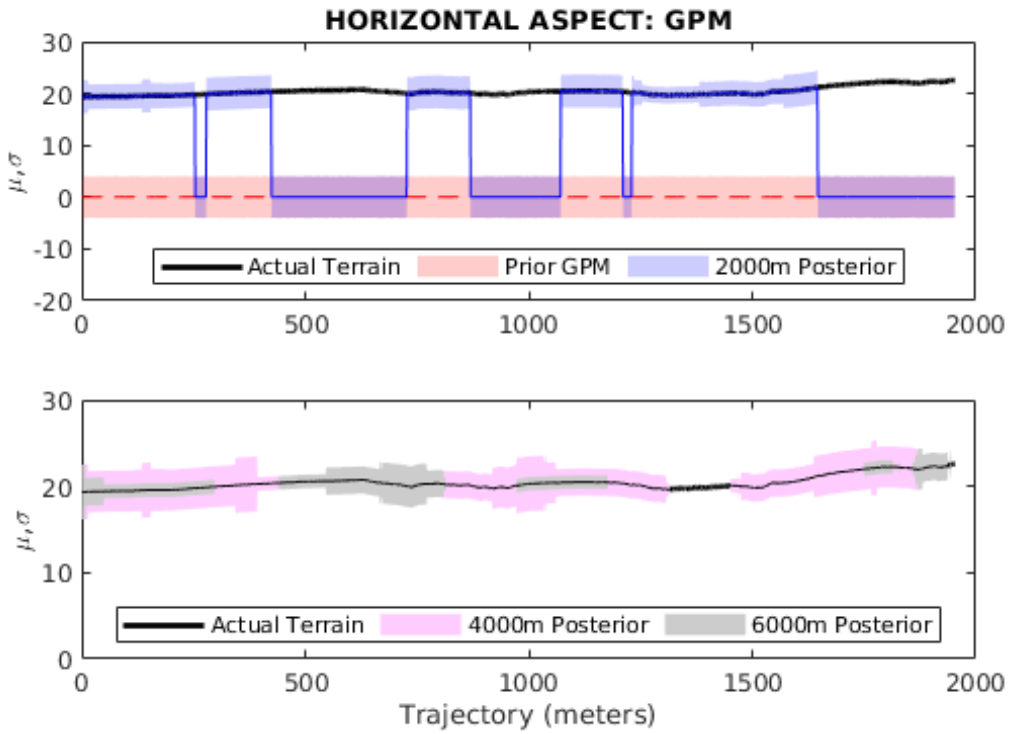


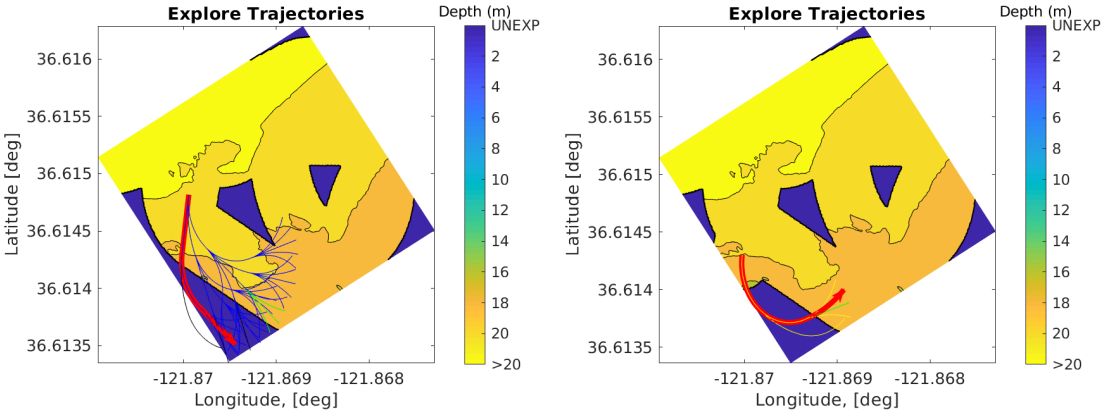
Figure 4.9. Horizontal GPM Aspect of Prior, 2000m, 4000m, and 6000m Trajectory over Reference Trajectory (Red) from $* \longrightarrow \triangle$ as shown in Figures 4.6, 4.7, and 4.8.

The initial AUV GPM, 2000m GPM, 4000m GPM, and 6000m GPM show an update to the true terrain mean and a decreasing covariance trend, corresponding to *exploration* occurring and map uncertainty decreasing.

4.4.2 Evaluating Future Trajectories

The GPM contributes to future trajectory evaluation by providing an *exploration* score. The highest *exploration* score parallels the highest information gain, whether that be exploring new areas or reducing the map uncertainty by re-covering previously explored areas. Looking at Figure 4.10, the purple constitutes uncharted regions, various yellow shades represent charted areas corresponding to depth, and the red trajectory shows the best trajectory for

exploration. The other trajectories not equivalent to the maximum *exploration* score are shown in green, yellow, and blue which correspond to *exploration* scores within the top third, middle third, and bottom third, respectively. All trajectories leaving the desired map region were not plotted for clarity.



(a) Trajectory 26.

(b) Trajectory 27.

(c) Trajectory 28.

(d) Trajectory 29.

Figure 4.10. Trajectory Evaluation for Trajectories 26-29 to Maximize Exploration. AUV only knows explored regions in yellow and unknown unexplored regions in purple. Red trajectory shows maximum exploration and green, yellow,and black show upper third, middle third, and bottom third exploration scores, respectively.

In trajectories 26-29 of Figure 4.10, the *exploration* component attempts to explore new areas and shows lower *exploration* scores for previously explored areas or areas with low

map uncertainty. In trajectories 27 and 28, the explored region has high map uncertainty, due to the AUV exploring other regions first, which leads to the AUV choosing to enter a previously explored region slightly earlier than other trajectories. Therefore, the maximum *exploration* trajectory can be validated as the trajectory with the highest information gain through either exploring new areas or reducing the map uncertainty.

4.5 Exploration Conclusions

The GPM allows for each individual cell of a larger map be quantified by a mean and a covariance. Using the GPM, *exploration* can be quantified as the information gain over a trajectory. Through *exploration*, the Matérn covariance function becomes more refined and the map uncertainty is lowered through either discovery of new areas or uncertainty reduction in high uncertainty areas. *Exploration* can be maximized through the employment of the KLD, which can compare each trajectory using the GPM mean and covariance. Lastly, with the constant incentive of *exploration*, the UV will provide complete coverage of a defined, bounded region which accomplishes the complete coverage criteria in Section 1.1.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Exploitation

Exploitation is the use of previously determined distinguishable terrain to localize the position of the UV, reducing the UV positional uncertainty to that of the terrain uncertainty. With lower positional uncertainty, the UV measurements reduces the original error and can more accurately map the terrain.

It is also important to distinguish the difference between position estimation discussed in Chapter 3, and *exploitation* discussed in this chapter. Position estimation, by means of a particle filter (PF), is used to minimize positional uncertainty of the UV based on errors introduced through sensors, modeling, and control actions, whereas *exploitation* uses a previously discovered terrain to localize position by loop closure, anchoring the positional uncertainty to a more recent UV position.

This section discusses, derives, and implements the *exploitation* component of the UV, which can be broken down into four parts: classifying the terrain using Boltzmann entropy, defining requirements for localization, introducing the Maximum *A Posteriori* (MAP) estimate, and applying the MAP estimate.

5.1 Boltzmann Entropy

The landscape ecology community recently classified terrain using the second law of thermodynamics [60], [61]. Configurational entropy, specifically Boltzmann entropy, represents the spatial unpredictability of the terrain [60], [61] and therefore quantifies the amount of information within the topology. The proportional variability of the ocean floor directly correlates to the amount of information that a sensor can detect, and further, can use to localize the position of the UV. Boltzmann entropy computes the number of permutations of a sample based on two assumptions: the sample is time independent and there is no prior knowledge of the sampled area. For most terrain mapping applications, both assumptions can be met. For time varying systems (e.g., sand ripples or dredging operations), the assumption can still apply for a small enough time interval in which the sampling occurs. The Boltzmann Entropy can be defined as $S = k_b \log(W)$ where k_b is the Boltzmann constant

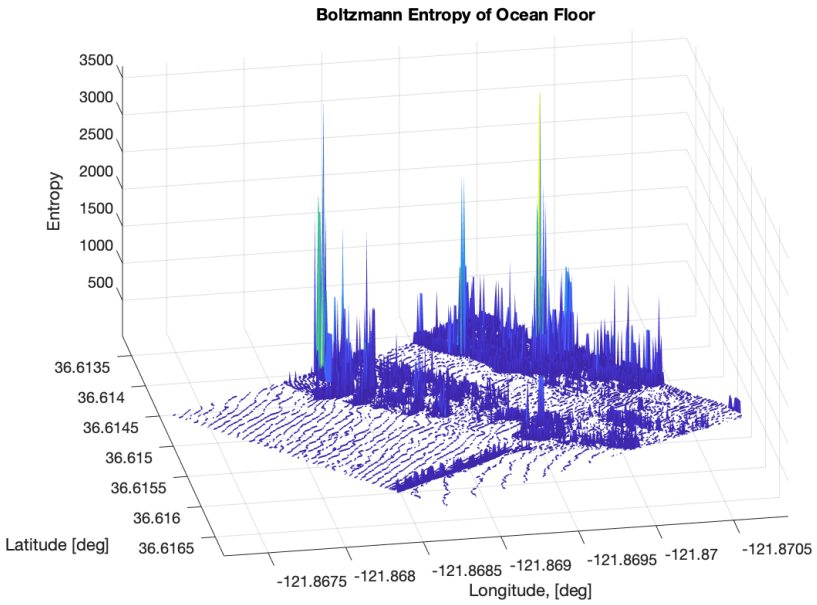
and W is the number of permutations within the sampled state. In this thesis, the Boltzmann Entropy is proportional to the number of permutations, and will be defined solely through its number of permutations

$$\xi(\mathbf{m}_n) \propto \log(W), \tag{5.1}$$

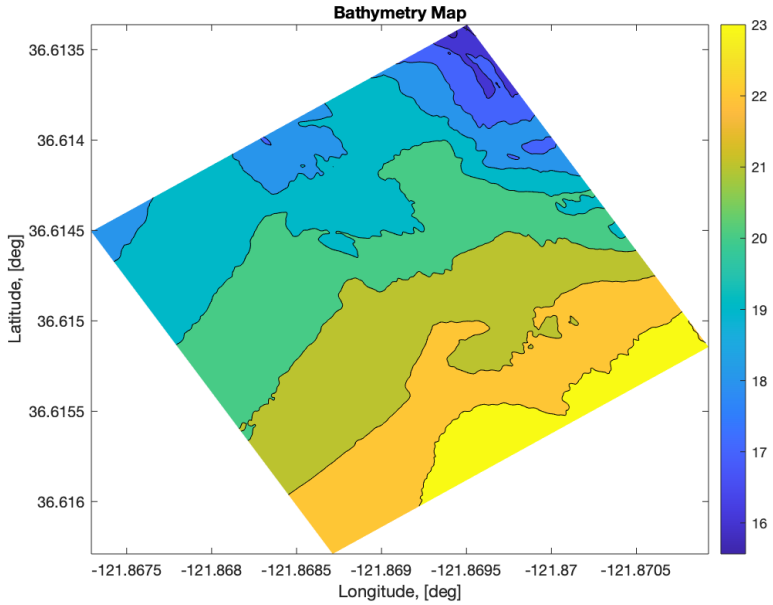
in which $\xi(\mathbf{m}_n)$ is the Boltzmann entropy at position $\mathbf{m}_n$ and W is the number of permutations.

5.2 Requirements for Localization

Through a Boltzmann entropy representation, terrain can be classified following the approach derived in [78] in which the Boltzmann entropy corresponds to the terrain uncertainty. A high Boltzmann entropy (high terrain uncertainty) translates to terrain features that have a strong potential to localize the position of the vehicle through the correlation method. Referring to Figure 5.1, the Boltzmann entropy quantifies the topology (5.1a) with large values corresponding to terrain features shown in the contour map (5.1b). With all sensors described in Section 1.6, the sensor accuracy is significantly better than that of the estimated map. Therefore, the localization is dependent, and limited by, the map resolution and is not reliant on any onboard systems.



(a)

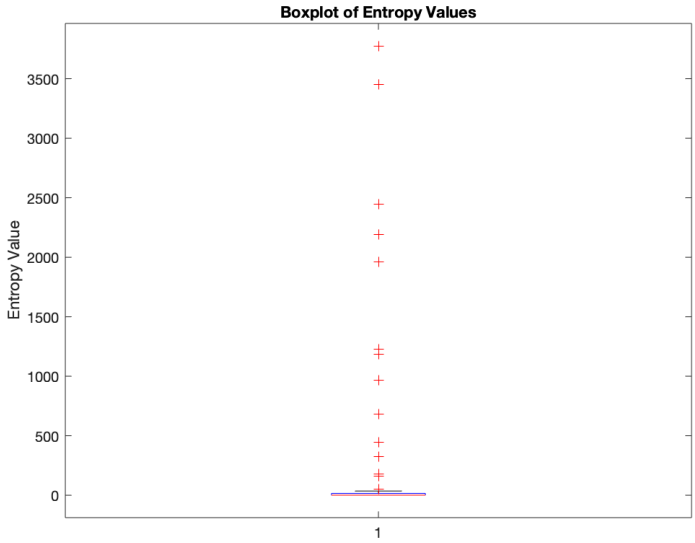


(b)

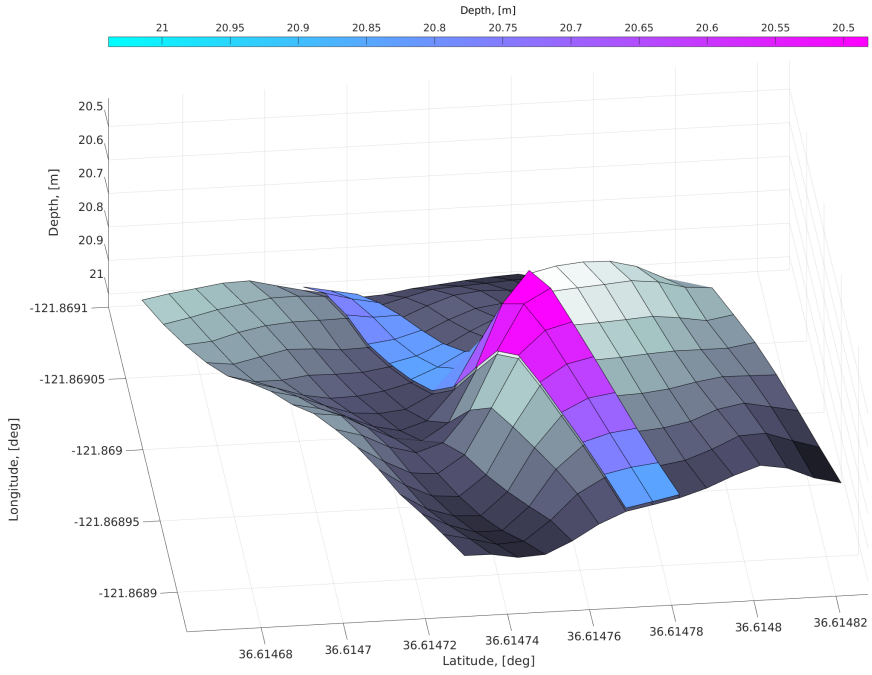
Figure 5.1. (a) Boltzmann Entropy Map (Ξ)and (b) Ocean Floor Bathymetry adapted from [78].

Due to the small amount of data available, two assumptions are needed in order to statistically evaluate the data and apply the model to any type of terrain, allowing the vehicle to operate anywhere as opposed to just one 255m $\times$ 255m area. The first assumption is that the 255m $\times$ 255m meter area is ergodic: the data contains all the properties of the larger area (specifically the ocean) and can therefore sufficiently model all areas of terrain. The second assumption is that the terrain can be modeled as a multi-variate Gaussian distribution—that a smaller sample of the map (in this case a 1m $\times$ 1m meter cell) can be represented fully through a entropy mean and variance within the cell.

The data from Figure 5.1 is tabulated, as represented by the box plot in Figure 5.2, which shows the high quantity of major outliers in the Boltzmann entropy map (Ξ). Furthermore, each major outlier is significantly different, which allows the UV to use these high Boltzmann Entropy values to localize its position.



(a)



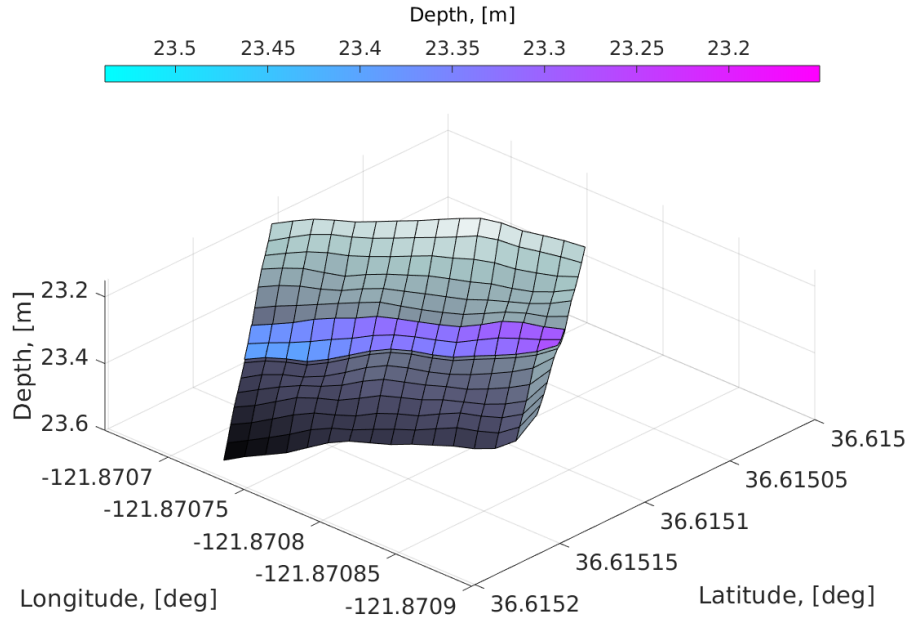
(b)

Figure 5.2. Statistical Modelling and Visual Representation of Boltzmann Entropy. (a) Box plot Boltzmann entropy values from Figure 5.1. (b) Visual representation of a portion of bathymetry map with highlighted terrain corresponding to a max entropy value of 300.

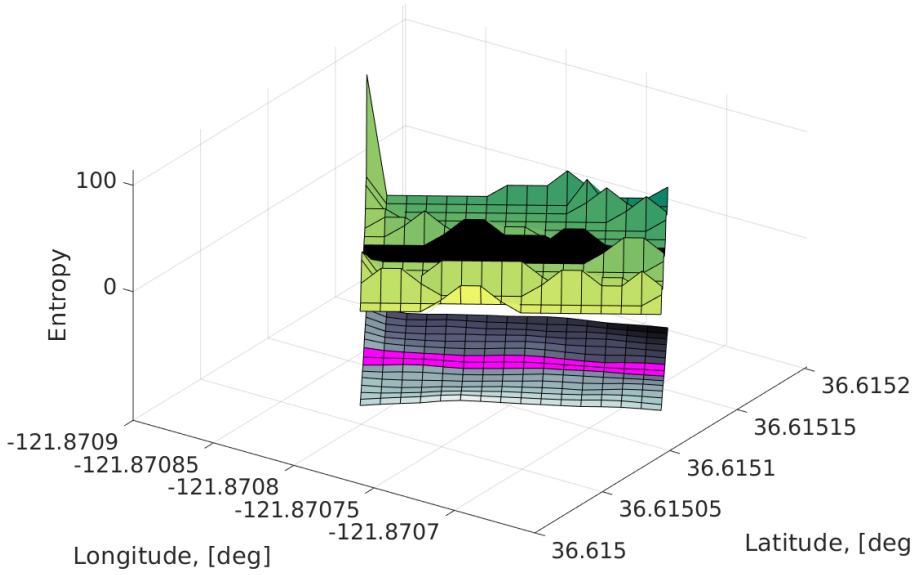
A Boltzmann entropy value of $\xi(\mathbf{m}) = 300$ can be seen visually in Figure 5.2b, showing asymmetrical terrain. No matter the orientation of the sensor used, the orientation of the UV can be determined based on the features position on the generated map.

To further demonstrate changes in Boltzmann entropy, Figure 5.3 and Figure 5.4 depict two drastically different areas with different maximum Boltzmann entropy ($\xi(\mathbf{m}) = 40$ for Figure 5.3 and $\xi(\mathbf{m}) = 2160$ for Figure 5.4).

Referring to Figure 5.3 and Figure 5.4, its very clear that a higher Boltzmann entropy corresponds to a better probability of localization. Furthermore, Figures 5.2, 5.3, and 5.4 show that at low entropy values there is a low probability of localization (flat terrain) but once Boltzmann entropy reaches a certain magnitude, the localization probability plateaus due to the asymmetrical terrain which lends the likelihood of localization, $P(X^*|\xi(\mathbf{m}_n))$, to be modeled as a cumulative distribution function (CDF).

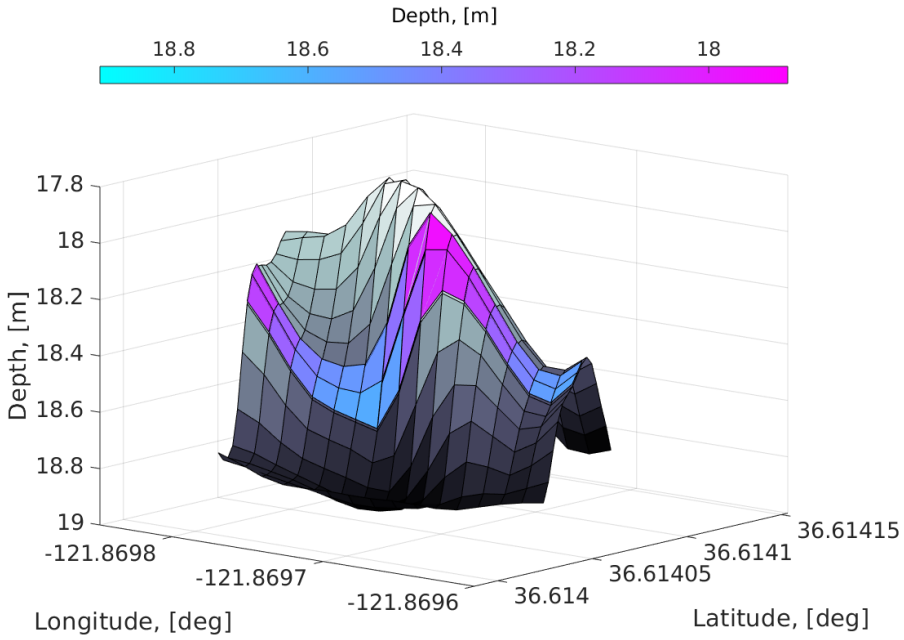


(a) max($\xi(\mathbf{m})$) = 40 Depth Profile.

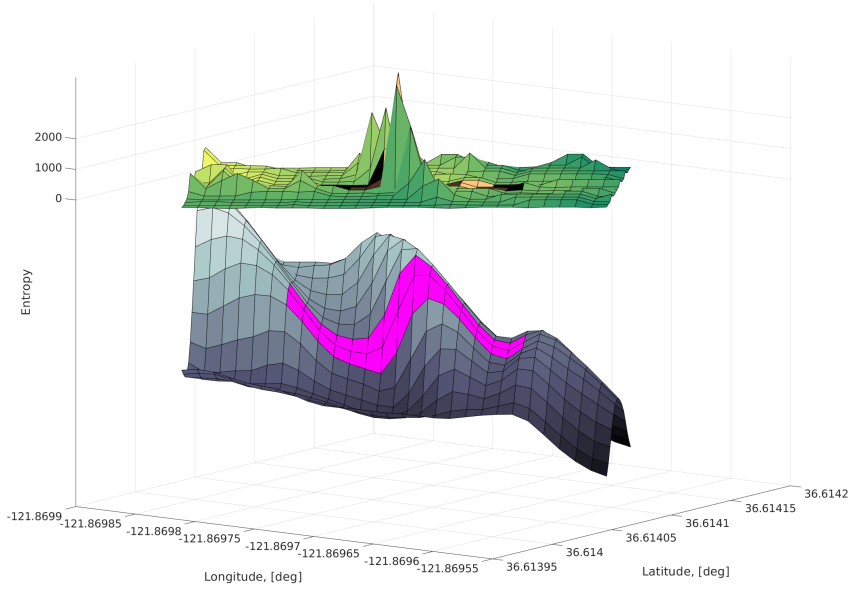


(b) max($\xi(\mathbf{m})$) = 40 Entropy Profile.

Figure 5.3. Boltzmann Entropy Visual Representation max($\xi(\mathbf{m})$) = 40. (a) Terrain Depth Profile. (b) Boltzmann Entropy Profile (above - green) and Terrain Depth Profile (below - grey). Sensor regions of Boltzmann entropy and terrain highlighted in black and pink, respectively.



(a) max($\xi(\mathbf{m})$) = 2160 Depth Profile.



(b) max($\xi(\mathbf{m})$) = 2160 Entropy Profile.

Figure 5.4. Boltzmann Entropy Visual Representation max($\xi(\mathbf{m})$) = 2160.(a) Terrain Depth Profile. (b) Boltzmann Entropy Profile (above - green) and Terrain Depth Profile (below - grey). Sensor regions of Boltzmann entropy and terrain highlighted in black and pink, respectively.

The data from Figure 5.2 is heavily skewed to the flat, smooth portions of the map (Boltzmann entropy equal to 1). Localization requires asymmetrical, uncertain terrain and therefore the flat areas were removed prior to modeling the probability of localization. Without the flat, asymmetrical areas, the Boltzmann entropy mean and standard deviation of the data from Figure 5.2 are $\mu_{\xi>1} = 83.0$ and $\sigma_{\xi>1} = 74.9$, respectively. With a specified mean and standard deviation of the Boltzmann entropy representation of the terrain omitting flat areas, the probability of localization, $P(X^*|\xi(\mathbf{m}_n)$, given the Boltzmann entropy at state $\mathbf{m}_n$ can be given through the Gaussian CDF,

$$P(X^*|\xi(\mathbf{m}_n) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{\xi(\mathbf{m}_n) - \mu_{\xi>1}}{\sigma_{\xi>1}\sqrt{2}}\right)\right], \tag{5.2}$$

with select inputs and outputs of the CDF function shown in Table 5.1.

Table 5.1. Boltzmann Entropy and Relative Localization Probability.

Boltzmann Entropy $\xi(\mathbf{m}_n)$	**Localization Probability** $P(X^*\|\xi(\mathbf{m}_n))$
1	13.7 %
50	32.9 %
100	59.0 %
150	81.4 %
200	94.0 %
300	99.8 %

Referring to Table 5.1 and Figures 5.2 - 5.4, a Boltzmann entropy value of $\xi(\mathbf{m}_n) = 1$ corresponds with a low localization probability of 13.7%. Flat terrain still gives a chance of localization, but the most important aspect of $P(X^*|\xi(\mathbf{m}_n))$ for $\xi(\mathbf{m}_n) > 300$ corresponds to about 100% localization, which matches with the asymmetrical terrain. Further examples of terrain and their respective Boltzmann entropy values are shown in Appendix C.

5.3 Maximum *A Posteriori* Estimate

Taking a deterministic view of the Boltzmann map, a MAP estimate utilizes the most expected hypothesis to make a prediction [86], [85]. Within ATAN, the most expected hypothesis corresponds to the vehicle position estimate using the best correlation between the vehicle and the terrain, as discussed in Chapter 3. Through the position estimate, multiple trajectories can be generated, as shown in Figure 5.5.

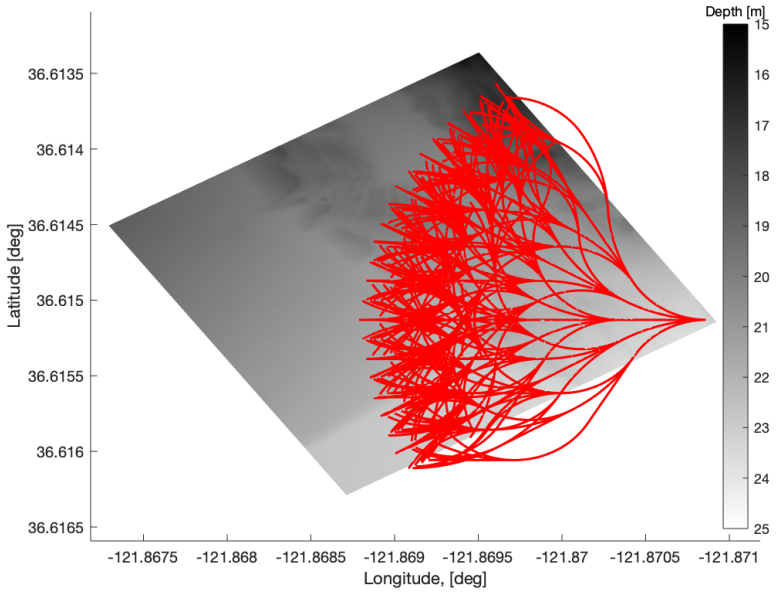


Figure 5.5. Possible Trajectories for AUV Shown in Red.

One disadvantage of the MAP estimate is the lack of an uncertainty measurement within the calculation [85], specifically the uncertainty with the generated Boltzmann entropy map (Ξ). However, the lack of uncertainty within the generated Boltzmann entropy map (Ξ) is small since the particle filter is accurate to within one meter and the vehicle map resolution is one meter. Therefore, in this thesis we utilize a deterministic approach to the Boltzmann entropy estimation and leave the stochastic system representation of Boltzmann entropy for future work.

Through a deterministic approach, Boltzmann entropy and the posterior distribution of explored areas can lead to each trajectory (Figure 5.5) being quantified with a MAP estimate to give an *exploitation* score. The optimal *exploitation* trajectory θ^* over all trajectories Θ

can be written as

$$\theta^* = \underset{\theta \in \Theta}{arg\,min} \Big(- \log P(X^*|\theta_i) \Big), \tag{5.3}$$

in which $P(X^*|\theta_i)$ specifies the localization probability along trajectory θ_i consisting of n IID (second order stationarity) points $\mathbf{m}_{i,\,n}$. Equation 5.3 can be written as a function of $\mathbf{m}_{i,\,n}$ through averaging [87],

$$P(X^*|\theta_i) = \frac{1}{N} \sum_{n=1}^{N} P(X^*|\mathbf{m}_{i,n}), \tag{5.4}$$

in which N is the number of discretized points along trajectory θ_i. Averaging does provide drawbacks for the resulting distribution mean and variance if the combined distributions are dissimilar [87]; however, in this instance all probability distributions $P(X^*|\mathbf{m}_{i,n})$ are identical as each is derived from equation 5.2. Each state $\mathbf{m}_{i,\,n}$ can be quantified through Boltzmann entropy such that equation 5.4 can be rewritten as

$$P(X^*|\theta_i) = \frac{1}{N} \sum_{n=1}^{N} P(\xi(\mathbf{m}_n)). \tag{5.5}$$

The likelihood of localization over trajectory θ_i depends on whether the terrain has been explored and the Boltzmann entropy of the terrain. Terrain that has not been explored provides no point of reference to the AUV and therefore, no chance of localization. The Boltzmann entropy provides a heterogeneity measure of the terrain which can be used to determine the best trajectory of the AUV. However, when a full trajectory covers a previously explored region, areas of localization (high entropy) have the possibility of being marginalized by lower entropy areas on the trajectory. Additionally, if a trajectory has high entropy over a majority of the trajectory, that trajectory should be looked at more favorably than a trajectory with a singular high entropy position. Therefore, in this work we propose the following weighted average approach

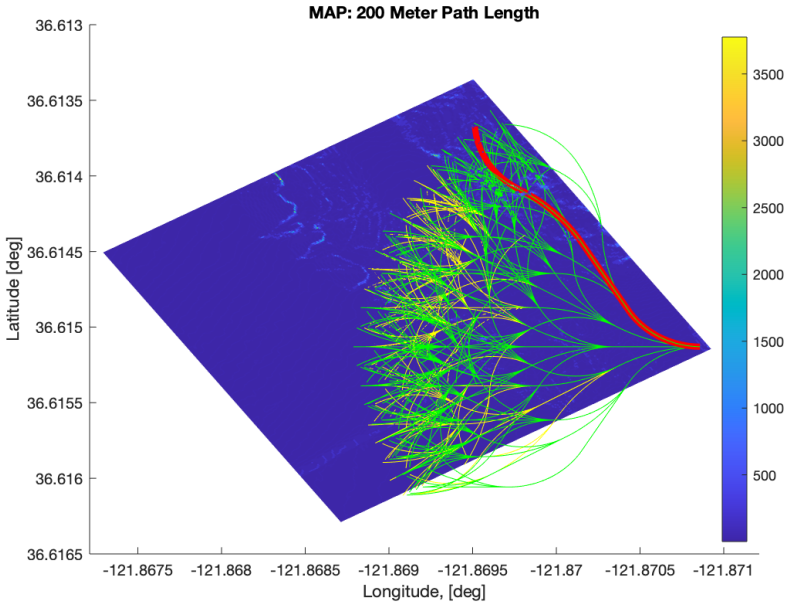
$$P(X^*|\theta_i) = \begin{cases} \frac{\sum_{n=1}^{N} \varepsilon(\mathbf{m}_{i,n})\ P(\xi(\mathbf{m}_n))}{\sum_{n=1}^{N} \varepsilon(\mathbf{m}_{i,n})}, & \text{if } \sum_{n=1}^{N} \varepsilon(\mathbf{m}_{i,n}) > 0 \\ 0, & \text{otherwise,} \end{cases} \tag{5.6}$$

in which

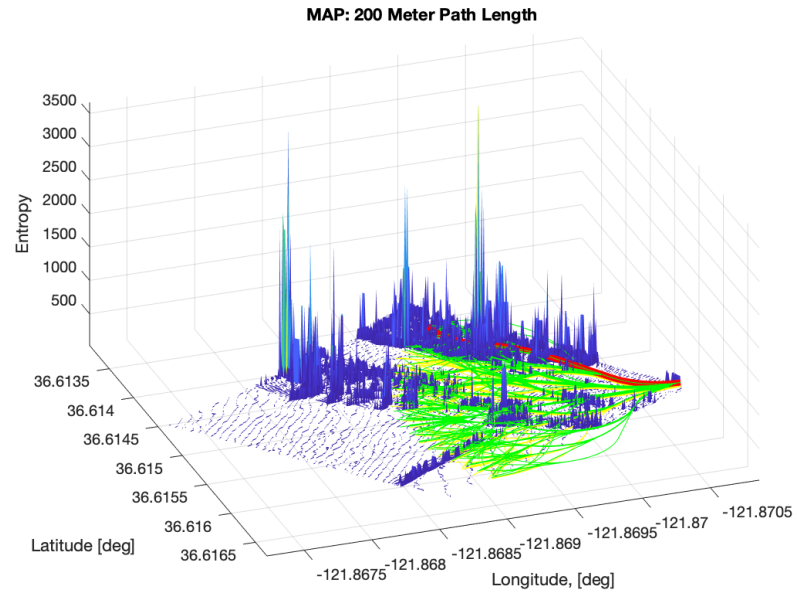
$$\varepsilon(\mathbf{m}_n) = \begin{cases} \xi^{\chi}(\mathbf{m}_n)\,, & \text{if } (\mathbf{m}_n) \text{ explored} \\ 0\,, & \text{otherwise.} \end{cases} \tag{5.7}$$

Multiple values of χ (equation 5.7) were investigated and we selected $\chi = 2$ as it led to the best results for prioritizing large individual entropy values but also giving overall precedence to multiple high entropy values ($\xi > 300$) over an entire trajectory.

Applying equation 5.6 to the trajectories generated in Figure 5.5 yields Figure 5.6 in which θ^* is shown in red with above and below average *exploitation* scores shown in green and yellow, respectively.

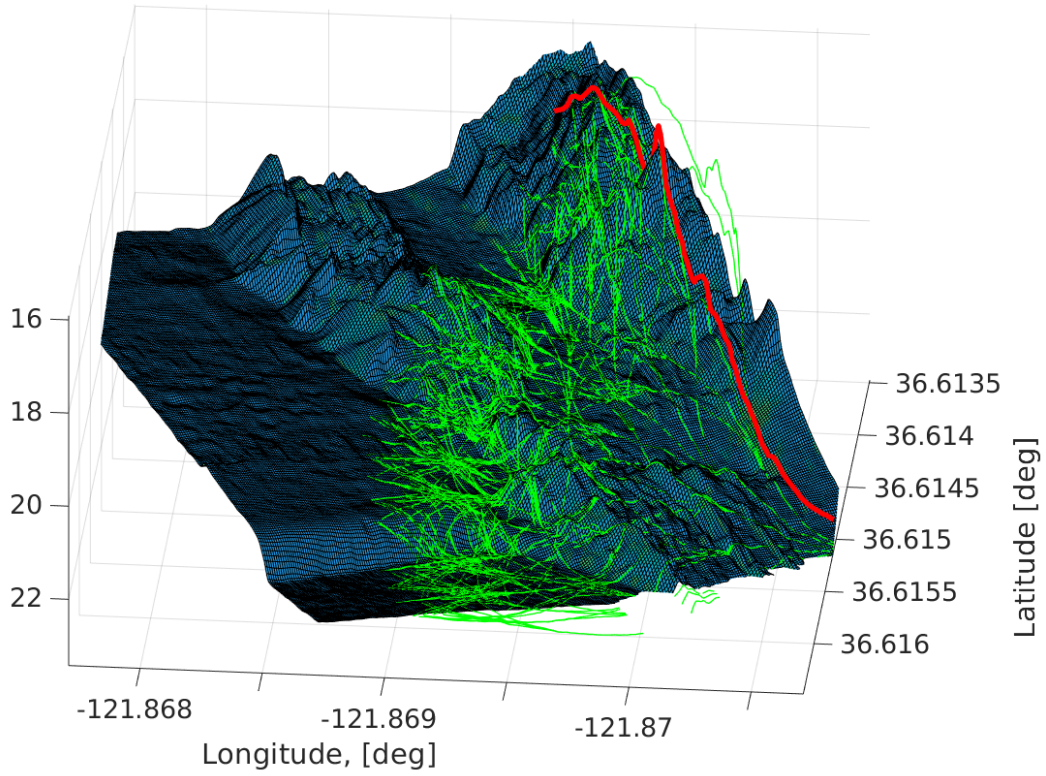


(a)



(b)

Figure 5.6. Exploitation Score Trajectory Analysis. The best trajectory (red) which corresponds to highest MAP estimate with above (green) and below average (yellow) *exploitation* scores shown in green and yellow, respectively. (a) Trajectory analysis top-down view over Boltzmann entropy map (Ξ). (b) Trajectory analysis projected onto a 3D entropy map.



(c)

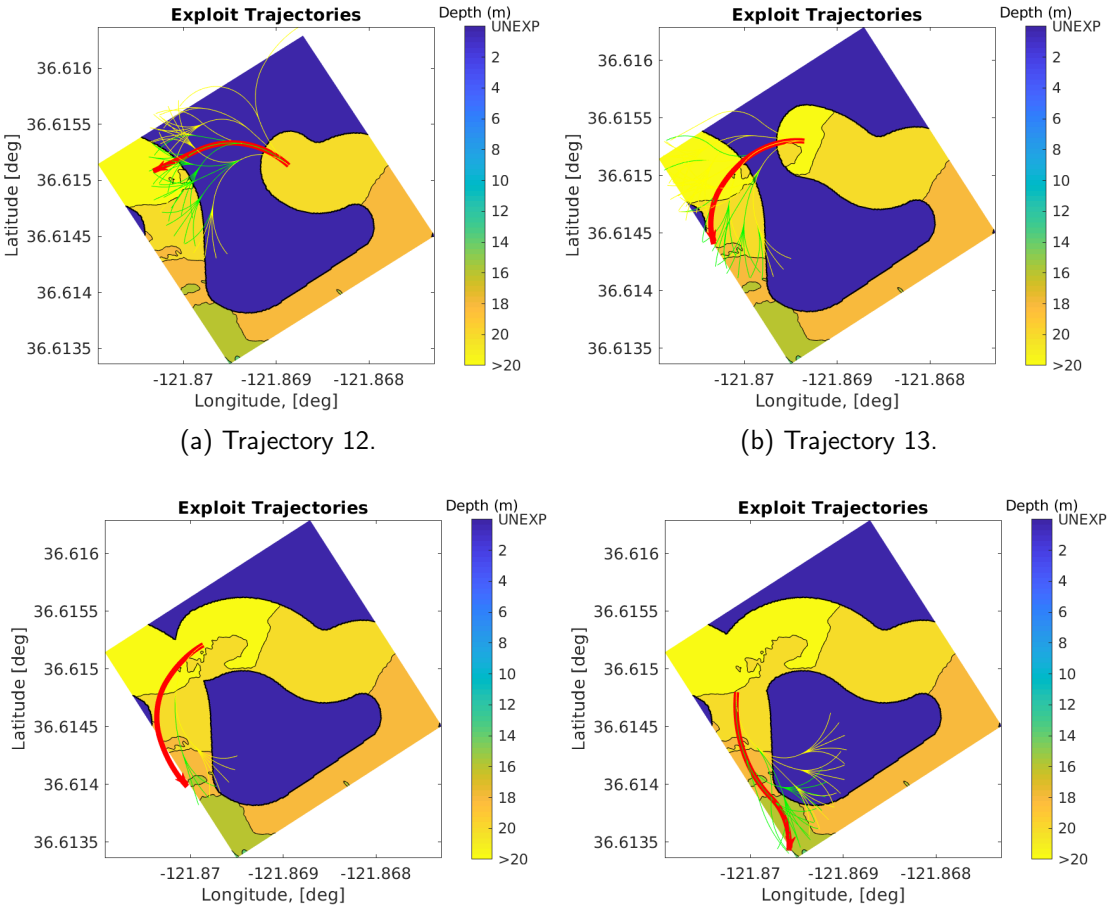
Figure 5.7. Exploitation Score Trajectory Analysis Using Bathymetry. Four horizons consisting of 50m trajectories utilizing heading rates of [-2,-1,0,1,2] degrees/second. The best trajectory (θ^*) is shown in red which corresponds to highest MAP estimate with all other *exploitation* scores shown in green. Trajectory analysis over terrain map.

Results show the entropy classification and MAP estimate can be shown effective, as the best MAP trajectory (red) encounters the most irregular terrain, as illustrated in Figure 5.7.

5.4 Application of the Maximum *A Posteriori* Estimate

Within ATAN, the UV will not have an *a priori* map, which was used in the derivation and implementation of the MAP. This section replaces the fully observable prior map used in Section 5.3 with the map generated through the *exploration* of the UV. Furthermore, the

mean and variance inputs to determine the probability of localization (equation 5.2) are based off of only previously explored areas. The resulting MAP estimate is shown in Figure 5.8.



(a) Trajectory 12.

(b) Trajectory 13.

(c) Trajectory 14.

(d) Trajectory 15.

Figure 5.8. Trajectory Evaluation over Trajectories 12-15 to Maximize MAP with No *A Priori* Map. AUV only knows explored regions (yellow) with unknown regions represented in purple. The maximum MAP is shown in red, with above average and below average MAP shown in green and yellow, respectively.

Each projected 150m path in Figure 5.8 shows the optimal path (red) with the highest MAP score. In trajectories 12-13, the MAP is fighting to get back to previously explored areas, while in trajectories 14-15, the best trajectory goes over higher entropy areas (jagged terrain)

which allow the AUV to localize. As more of the map is explored, the MAP will have more trajectories to choose from, resulting in higher *exploitation* scores.

5.5 Exploitation Conclusions

The *exploitation* score is calculated using projected trajectories of the UV coupled with the MAP, which uses the Boltzmann entropy classification of previously explored terrain. By having a quantifiable measurement of the *exploitation* aspect, the UV has the possibility of factoring the *exploitation* score into the determination of its next trajectory, in which the UV can determine the best trajectory for localization. Through *exploitation* of the terrain, both the map and UV position errors are reduced, accomplishing the reduction of positional uncertainty objective stated in Section 1.1. Furthermore, the *exploitation* score feeds into the determination of the best trajectory for the UV, whether that be *exploration*, *exploitation*, or a combination of the two.

CHAPTER 6:
Reward Function

Exploration and *exploitation* are the two factors that must be optimized to efficiently and effectively map an unknown area. *Exploration* is seeking out new areas while *exploitation* is minimizing the position of uncertainty (PUC) to enable accurate navigation. This push and pull between *exploration* and *exploitation* is managed through a reward function, a mathematical framework that ensures that *exploration* and *exploitation* occur at the proper time. However, *exploration* (spatial estimator) and *exploitation* (temporal estimator) have different units and describe different qualities with different metrics, making it difficult to determine the proper weighting between them. This chapter introduces the overall format of the reward function and discusses appropriate gains for *exploration* and *exploitation*.

6.1 Reward Function

The reward function is a result of the dual optimization calculation that determines the reward, $R(x_s)$, of state x_s as a function of three components: the *exploration* score, Υ_1; the *exploitation* score, Υ_2; and a penalty for going outside the boundaries of the desired area, (P_{OFF}). The relationship between the *exploration* score, *exploitation* score, and out of area penalty can be formulated into an equation,

$$R(x_s) = k_1\Upsilon_1 + k_2\Upsilon_2 - \mathrm{P_{OFF}}, \tag{6.1}$$

where k_1 and k_2 are the respective gain values for *exploration* and *exploitation*, and P_{OFF} is the out-of-area penalty. The subsequent sections showcase different methods for selecting values for k_1 and k_2, specifically constant gains, adaptive heuristic gains, and optimal estimator gains.

6.1.1 Constant Gain Model

Constant gains are the easiest to model, in which the magnitude of the gain relates to the importance of the metric. In this case, *exploration* and *exploitation* are equally weighted, $k_1 = k_2 = 0.5$, as both are equally important.

However, constant gains do not allow for *exploration* nor *exploitation* to take precedence over the other. In this thesis, the constant gains of $k_1 = k_2 = 0.5$ set a minimum baseline for the interaction between *exploration* and *exploitation*.

6.1.2 Adaptive Heuristic Gain Model

The prioritization between *exploration* or *exploitation* can be determined through adaptive gains. For example, when *exploration* takes precedence, the *exploration* gain will be higher than that of the *exploitation* gain, resulting in a higher reward for paths that have higher *exploration* scores.

The *exploration* heuristic adaptive gain is a function of the map area covered. As a larger map area becomes explored, it becomes more difficult to find unexplored areas and, in order to prioritize *exploration*, the *exploration* gain needs to be increased. The following function is used for the *exploration* heuristic adaptive gain:

$$k_1 = \begin{cases} \frac{\zeta}{1-\zeta}, & \text{for } k_1 < k_{1,max} \\ k_{1,max}, & \text{otherwise,} \end{cases} \tag{6.2}$$

where ζ is the fraction of the map area covered. Since effective *exploration* cannot occur without localization, the explore score is capped at a maximum gain value $k_{1,max}$ (this thesis uses $k_{1,max} = 10$) such that *exploitation* can still occur if both are required.

The heuristic adaptive gain for *exploitation* is twofold: first, the algorithm changes to prioritize closer localization points, and second, k_2, increases without bound until localization occurs. First, to prioritize closer high entropy areas, the MAP estimate utilizes an additional weighting component,

$$w(\xi(\mathbf{m}_n)) = \text{N, N-1,...2, 1} \tag{6.3}$$

in which N is the number of discretized points along trajectory θ_i, to calculate the likelihood of localization,

$$P(L|\theta_i) = \begin{cases} \frac{\sum_{n=1}^{N} c(\mathbf{m}_{i,n})\ w(\xi(\mathbf{m}_n))\ P(\xi(\mathbf{m}_n))}{\sum_{n=1}^{N} c(\mathbf{m}_{i,n})\ w(\xi(\mathbf{m}_n))}, & \text{if } \sum_{n=1}^{N} c(\mathbf{m}_{i,n}) > 0 \\ 0, & \text{otherwise,} \end{cases} \tag{6.4}$$

in which

$$c(\mathbf{m}_{i,n}) = \begin{cases} \xi^2(\mathbf{m}_{i,n}), & \text{if } \mathbf{m}_{i,n} \text{ explored} \\ 0, & \text{otherwise.} \end{cases} \tag{6.5}$$

Second, the *exploitation* heuristic adaptive gain, k_2, increases without bound until localization occurs and is a function of the AUV position error,

$$k_2 = s_f \omega^2 \tag{6.6}$$

in which s_f is a scaling factor, ω is the position error and a function of the position estimate x_k and dead reckoning system model x_{DR}, defined as $\omega = ||x_k - x_{DR}||$. For this thesis, $s_f = 15$ was used to prioritize *exploitation* over *exploration*. Since *exploitation* is critical in order to create an accurate map, the *exploitation* gain increases without bound such that no matter how large the *exploration* gain (maximum of 10 for this thesis), the *exploitation* score will eventually gain precedence and localization will occur.

6.1.3 Adaptive Optimal Estimator Gain Model

The optimal estimator gain values k_1 and k_2 are derived in Appendix D as a function of uncertainty between the PF and the GPM,

$$k_1 = \frac{\hat{\sigma}_2^2}{\hat{\sigma}_1^2 + \hat{\sigma}_2^2} \tag{6.7}$$

$$k_2 = \frac{\hat{\sigma}_1^2}{\hat{\sigma}_1^2 + \hat{\sigma}_2^2}, \tag{6.8}$$

where $\hat{\sigma}_1^2$ is the normalized variance of the GPM and $\hat{\sigma}_2^2$ is the normalized variance of the PF.

The *exploration* gain constant, k_1, is dependent on the variance of the position estimator,

and is calculated as the variance of the distribution of the particles,

$$\sigma_2^2 = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (x_i - x_j)^2, \tag{6.9}$$

in which N is the number of particles and $x_i - x_j$ is the distance between particles i and j. To normalize equation 6.9, an exponential decay model is used,

$$\hat{\sigma}_2^2 = e^{-\sigma_2^2}. \tag{6.10}$$

If the PF has a low distribution variance, the UV has a low positional uncertainty, which makes $\hat{\sigma}_2^2$ larger, increasing k_1 and prioritizing *exploration*. If the PF distribution is large, then $\hat{\sigma}_2^2$ is small which lowers the *exploration* gain so that *exploitation* can be performed.

The global estimator variance, σ_1^2, combines the variances of the GPM and is defined as [80]

$$\sigma_1^2 = \frac{1}{M^2} \sum_{i=1}^{M^2} \sigma_i^2 + \frac{1}{M^2} \sum_{i=1}^{M} \sum_{j \neq i}^{M} \mathbb{E}[(Z_{Vi} - Z_{Vi}^*)(Z_{Vj} - Z_{Vj}^*)], \tag{6.11}$$

in which M^2 is the number of elements ($M \times M$ grid), σ_i^2 is the predictive variance per cell of the GPM, Z_V is the mean value per cell , and Z_V^* is the predictive mean value per cell. Taking into account second order stationarity, equation 6.11 simplifies to

$$\sigma_1^2 = \frac{1}{M^2} \sum_{i=1}^{M^2} \sigma_i^2. \tag{6.12}$$

Equation 6.12 can be re-written for an L by H area as the average variance of the GPM of the map:

$$\sigma_1^2 = \frac{1}{LH} \sum_{l=1}^{L} \sum_{h=1}^{H} \sigma_{l,h}^2. \tag{6.13}$$

To normalize the global estimator variance, the exponential decay model is used,

$$\hat{\sigma}_1^2 = e^{-\sigma_1^2}. \tag{6.14}$$

As the map becomes more explored, the overall GPM variance (σ_1^2) decreases, increasing the normalized GPM variance ($\hat{\sigma}_1^2$) and raising the *exploitation* gain constant (k_2). As a larger map area gets explored, a higher *exploitation* gain value is provided as *exploitation* becomes more necessary to ensure accuracy of the UV position.

6.1.4 Out-of-Area Penalty

The out-of-area (OOA) penalty solves two issues: (1) minimize the time the UV is outside the prescribed map and (2) ensure that the UV is able to map the sides and corners of an area without incurring the OOA penalty. The UV will not abide by the boundaries of the map if the penalty is too small, and the UV will not approach the boundary if the penalty is too large. In this thesis we propose an OOA penalty of the form,

$$P_{OFF} = \begin{cases} \frac{d}{d_A}, & \text{for } d < d_A \\ d^2, & \text{otherwise,} \end{cases} \tag{6.15}$$

in which d is the distance off the map and d_A is the allowed distance. For this thesis, $d_A = 10$ to allow for access to the corners and edges of the map.

6.2 Reward Conclusions

The reward function utilizes three different methods to best determine the weighting between *exploration* and *exploitation* through a constant gain model, an adaptive heuristic model, and an optimal estimator model. These three different methods for determining *exploration* and *exploitation* gains allow for a comparison between each approach through a trajectory evaluator (discussed in Chapter 7), to determine the best way to leverage *exploration* and *exploitation*.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 7:
Trajectory Planner

Only about one percent of the ocean floor has been explored [88], leaving a massive, unknown search space with minimal areas to perform *exploitation*. In order to decide which trajectory is the best of the set, the trajectory planner leverages *exploration* and *exploitation* through the reward function, as previously discussed in Chapter 6. However, this problem is computationally complex, known specifically as PSPACE-complete [89], which makes solutions intractable. Therefore, limitations and constraints are placed on the UV and a partially observable Monte Carlo planning process (POMCP) — a POMDP coupled with a MCTS — is used to significantly reduce the computational complexity to make this problem solvable in real time on a computationally limited UV.

7.1 Computational Perspective

The world doesn't yet possess the computational firepower to fully solve the dual optimization problem between *exploration* and *exploitation* with infinite trajectories to evaluate over infinite horizons, even for an area that is as small as 255m $\times$ 255m. Using vehicle sensors, the minimum path length can be computed using the search area and sensor size. For example, if the search area A is a 255m $\times$ 255m area and the sensor size S equals 60m, the minimum path length L for complete coverage with no sensor overlap can be calculated though $A = SL$ which results in a path length of $L = 1093$ meters. Constraining the vehicle to a 50 meter path length that allow for constant heading rates changes of $[-2, -1, 0, 1, 2]$ degrees per minute, the computational complexity over the total path length reduces to 5^{22} or $\approx 10^{15}$ computations solely for the trajectory planner. While there are computers that are on the petascale computing level [90], they are very expensive, too large for an AUV, and overall not feasible.

The REMUS sensor size is around ten meters which translates to a minimum path length of 3277 meters and a computational complexity equal to 5^{66} or $\approx 10^{46}$. Thus, the computational complexity of this problem is too large as the problem stands, but recent Machine Learning advances in Reinforcement Learning have provided a path to indirectly solve for the solution, using the combination of Decision Theory and Game Theory called a partially observable

Monte Carlo planning process (POMCP), which is used as the trajectory planner.

7.2 Decision Theory

Decision Theory combines probability theory and a utility function to calculate a reward value for a framework for decision sequences that involve an aspect of uncertainty [7]. The basic building block for Decision Theory is a Markov Decision Process (MDP), which can be expanded into a partially observable Markov Decision Process (POMDP) for partially observable states. Further, the same methodology can be performed over a finite region when the POMDP computational complexity becomes too large.

7.2.1 Markov Decision Process (MDP)

A Markov Decision Process (MDP) defines a model that can be used to solve a sequential decision problem for a stochastic, fully observable environment that consists of four components [86]:

- A set of states, $s \in \mathbb{S}$ with initial state s_0.
- A set of actions, $a \in \mathbb{A}$, for each state.
- A transition state model, P(s'|s,a), that calculates the probability of reaching state s' if action a is applied to state s.
- A reward function, $R(s) = \mathbb{E}[r_{t+1}|s_t = s, a_t = a]$ [6].

The solution, known as a policy, π_i, consists of an action sequence that the agent should employ from the initial state. The Optimal Policy, π^*, is the policy that produces the highest expected reward.

For example, consider the problem from [86], as shown in Figure 7.1, in which the agent starts at an initial state, s_0, with the goal of reaching either terminal states (+1,-1), with the (+1) terminal state being preferred. The stochastic environment may lead to a different policy, $\pi_i(a)$, for each iteration of the planning process that may or may not achieve the terminal state. After the planned policy is generated, the policy is evaluated based on the reward of the policy: the combination of the utility and probability of reaching the desired state.

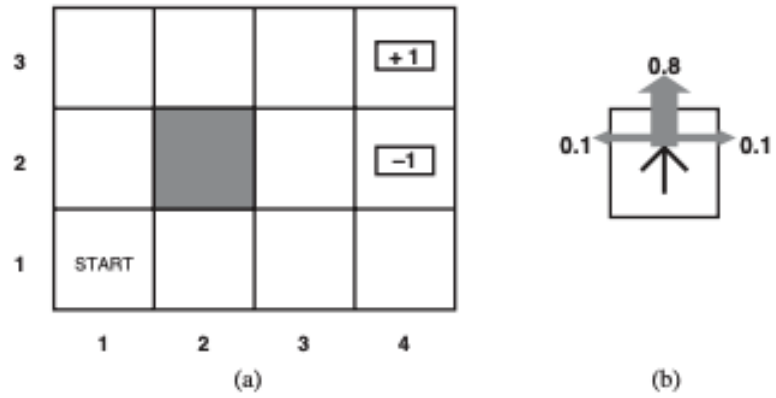


Figure 7.1. (a) Simplified Environment for Sequential Decision Problem with terminal states +1 and -1 and all other states yielding a reward of -0.04, with exception of dark square which is an obstacle. (b) Transition Model Illustration: 0.8 probability of intended movement, 0.2 probability agent moves in unintended direction. Source: [86].

The optimal policy, $\pi^*(a)$, would be the policy with the best expected reward, for example $a = [\,UP, UP, RIGHT, RIGHT, RIGHT\,]$ [86]. The utility of the optimal policy is the sum of the individual utility of each cell in the path, calculated as $U_T = -0.04 \times 5 + 1 = 0.8$. The -0.04 reward for each state gives the agent an incentive to move, as opposed to a +0.04 reward where the agent stays without the risk of being penalized and accumulate utility. Each action of the optimal policy, $\pi^*(a)$, has a success probability of 0.8 of moving correctly, $P_S = 0.8^5$. There is also a possibility that the agent plans to follow the optimal policy and accidentally reaches the terminal state, for example a = $[\,RIGHT, RIGHT, UP, UP, RIGHT]$, which gives a failure probability of $P_F = 0.1^4$. The total probability of success in this instance would be the combination of the success and failure, $P_T = 0.8^5 + 0.1^4 = 0.32776$. The other probabilities of failure are not possible as either they go through the (-1) state, or take more than five moves to achieve the terminal (+1) state. The reward can then be calculated as $R = U_T P_T = (0.8)(0.32776) = 0.262208$.

MDPs work well in fully observable environments where the utility of each state is known and the actions are probabilistically reliable. However, as the problem referenced in Figure 7.1 grows in size, the computational requirements to determine the optimal policy increase exponentially. Further, in most environments, the actions will not be as reliable, and the states for each action also may not be known. Mapping an underwater area of 255m $\times$ 255m

meters where the initial, interim, and terminal states as well as their respective utilities and probabilities are unknown makes the MDP incapable of determining the optimal solution, as the problem is no longer fully observable.

7.2.2 Partially Observable Markov Decision Process (POMDP)

Relaxing the fully observable assumption, the agent no longer knows its present state and the problem shifts to a partially observable environment and requires a partially observable Markov Decision Process (POMDP) to develop a solution. The POMDP still utilizes the same basic elements of the MDP—a set of states $\mathbf{s} \in \mathbb{S}$, a set of actions $\mathbf{a} \in \mathbb{A}$, a transition model $P(\mathbf{s}'|\mathbf{s}, \mathbf{a})$, and a reward function R($\mathbf{s}$)—while adding an observation model $P(o|\mathbf{s})$ to calculate the probability of an observation o at state $\mathbf{s}$ [6], [86]. The observation model consists of the observation, $o \in \mathbb{O}$, which is then used to calculate the observation probabilities $P(o|\mathbf{s})$ [6].

The history, $\mathbf{h}_t = [\mathbf{a}_1, o_1, ...\mathbf{a}_t, o_t]$ is the sequence of actions coupled with their respective state observations [6], as opposed to the MDP use of the utility of that state. The policy, $\pi(\mathbf{h}, \mathbf{a})$ correlates the history to the probability distribution of the actions. The reward is the total reward accumulated over the policy, $R_t = \Sigma_{n=t}^{\infty} \gamma^{n-t} r_n$, [6] where γ is the discount factor and r_n is the reward for each state of policy length n. The value function $V^{\pi}(\mathbf{h}) = \mathbb{E}_{\pi}[R_t|\mathbf{h}_t = \mathbf{h}]$ is the expected reward from state $\mathbf{s}$ while performing policy π. From the value function, the optimal value function is the maximum value function, $V^*(\mathbf{h}) = max_{\pi} V^{\pi}(\mathbf{h})$ [6].

For every POMDP, there exists at least one optimal value function, $V^*(\mathbf{h})$, and corresponding policy, $\pi^*(\mathbf{h}, \mathbf{a})$, that gives the best solution [6].

7.2.3 Finite-Horizon Partially Observable Markov Decision Process (POMDP)

The computational power required to compute the POMDP for all states to determine the best policy is not always available. In the case of the computationally limited REMUS vehicle, the POMDP requires an adjustment so that the REMUS can perform path planning in real time to map a 255m $\times$ 255m area. The Finite-Horizon POMDP limits the computational complexity of POMDP by only looking over a finite horizon, and the optimal solution can

be computed with the following equations [4]:

$$Q_t(\mathbf{b}, \mathbf{a}) = \rho(\mathbf{b}, \mathbf{a}) + \gamma \int_{\mathbf{z}' \in \mathbb{Z}} \eta(\mathbf{z}'|\mathbf{b}, \mathbf{a}) V^*_{t-1} \tau(\mathbf{b}, \mathbf{a}, \mathbf{z}') d\mathbf{z}' \tag{7.1}$$

$$V^*_t(\mathbf{b}) = \sup_{\mathbf{a} \in \mathbb{A}} Q_t(\mathbf{b}, \mathbf{a}) \tag{7.2}$$

$$\pi^*_t(\mathbf{b}) = \arg\sup_{\mathbf{a} \in \mathbb{A}} Q_t(\mathbf{b}, \mathbf{a}), \tag{7.3}$$

where $Q_t(\mathbf{b}, \mathbf{a})$ gives the total expected reward for t discrete periods when action $\mathbf{a}$ is performed at belief state $\mathbf{b}$ as a function of the initial condition, $\rho(\mathbf{b}, \mathbf{a})$ at $t = 1$; discount factor $\gamma \in [0, 1]$; prior probability of observing $\mathbf{z}'$ given belief state $\mathbf{b}$ and action $\mathbf{a}$, $\eta(\mathbf{z}'|\mathbf{b}, \mathbf{a})$; and the Bayes Rule update $\tau(\mathbf{b}, \mathbf{a}, \mathbf{z}')$ [4]. Equations 7.2 and 7.3 show the smallest element of $\mathbf{a} \in \mathbb{A}$ that maximizes Q_t.

7.3 Game Theory

Game Theory, an extension of Decision Theory, accounts for the interactions between multiple agents [7], and has recently been used in computer-based versions of large-branching, turn-based games such as Go, Chess, and Checkers. For a coverage mission objective using ATAN with the REMUS vehicle, exploration and exploitation can be leveraged to search beyond the finite horizon discussed in Section 7.2.3 by using a MCTS and then coupling the MCTS, resulting in the overall methodology of a POMCP.

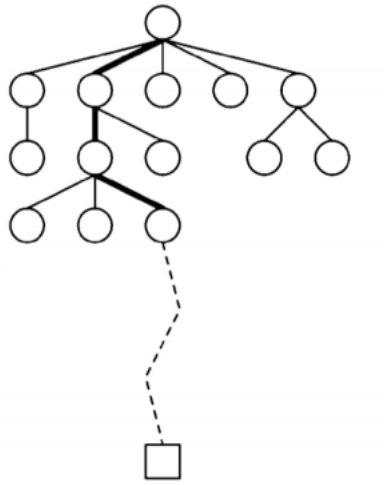
7.3.1 Monte Carlo Tree Search (MCTS)

Intractable integrals like equation 7.1 are approximated through a Monte Carlo tree search (MCTS) [7] that heuristically searches the integral decision space until either a solution is known or a search limit has been reached (e.g., temporal or computational). The MCTS has two fundamental properties: the true value of an action can be estimated through a random search iteration and subsequently be used to bias the policy and result in a best-first search strategy [7].

Monte Carlo Tree Search (MCTS) Algorithm

The MCTS is used to build and search a decision tree by sequentially selecting the best branches. The decision tree (shown in Figure 7.2) is anchored by a root node v_0 at state s_0, that contains the reward value for the node, $Q(s)$, and a visit count for the node, $N(s,a)$, that is connected to its child nodes based on actions $a \in A$. Those child nodes become the parent node of new child nodes related by the same actions $a \in A$.

Each node is initialized to $Q(s) = 0$ and $N(s,a) = 0$ and each MCTS iteration starts at the root node. Each iteration consists of three policies— tree, default, and backpropagation— which interact using Algorithm 4 shown in Figure 7.2.



(a)

Algorithm 4: MCTS [7]

```
1 function MCTS(s_0)
2 create root node v_0 with state s_0
3 while Within computational budget
    do
4   | v_l ← TREEPOLICY(v_0)
5   | Δ ← DEFAULTPOLICY(s(v_1))
6   | BACKPROPAGATION(v_l, Δ)
7   | return a(BESTCHILD(v_0, 0))
8 end
```

(b)

Figure 7.2. (a) MCTS Visual Description of the Decision Tree. Source: [7]. (b) MCTS Algorithm. Source: [7].

The relationships between the tree policy, default policy, and backpropagation are illustrated in Algorithm 4 and Figure 7.3. The tree policy selects and expands the search tree, the default policy produces a reward estimate of one random iteration of the tree search, and backpropagation adds the rewards of descendent children to the parent node.

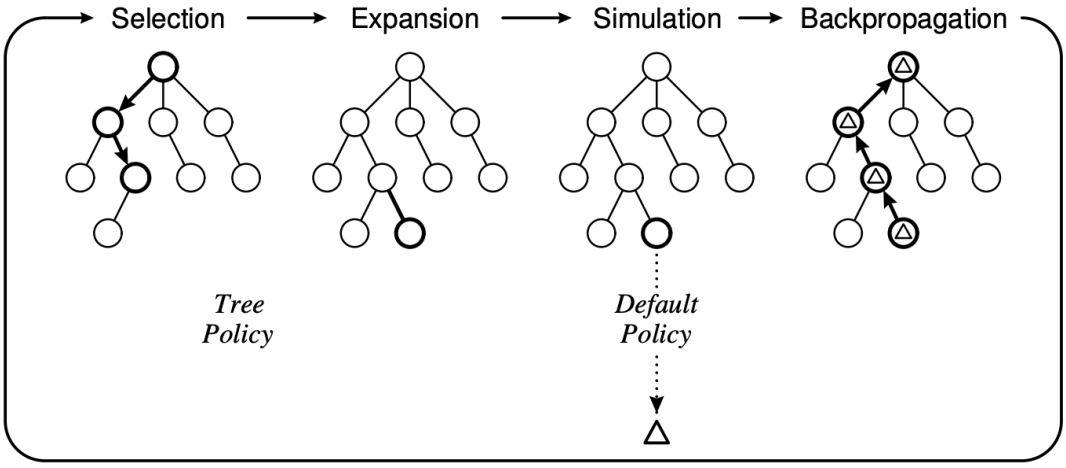


Figure 7.3. MCTS Algorithm. Source: [7].

In selection, the tree policy recursively chooses the best branch at each node tree through the upper confidence bound for trees (UCT) defined as [6], [7]

$$Q^*(s,a) = Q(s,a) + c\sqrt{\frac{lnN(s)}{N(s,a)}}. \tag{7.4}$$

As $c \to 0$, the UCT algorithm acts more greedily. Once every action from state s has been evaluated, the algorithm selects the action maximizing Q^*.

In expansion, the search is no longer bounded by the tree, and new branches need to be formed. More child nodes are added to the tree based on the actions available, $a \in A$, by uniformly picking the subsequent action (modeled as a uniform random variable) [6]. In simulation, each node is evaluated based on the default policy. In backpropagation, the evaluation result is backtracked to the root node, where it updates the value $Q(s)$ and the visit count $N(s,a)$ for each node visited.

The search terminates when either the search is complete or the computational budget is reached. At this point, the next action is determined using one of the following criteria: max child (highest value), robust child (highest visit count), max-robust child (both highest value and visit count), or secure child (maximize lower confidence bound) [7].

The search tree then conducts a pruning operation, removing all nodes not descendent from

the selected action child node, and the MCTS process commences again.

Note there are two places where biasing can affect the outcome of the MCTS search — child selection policy and action selection criteria.

Improving MCTS: Child Selection Policy

The next action the AUV executes is determined through the MCTS child selection policy. There are four standard options [7]:

- maximum reward child
- most visited child
- highest reward, most visited child
- highest low confidence child.

In this work we utilize the highest reward, most visited child with a secondary option on maximum reward child. As the MCTS will be completed as the AUV traverses a trajectory, the AUV requires a new command at the end of its current trajectory prior to the beginning of the new trajectory. If the highest reward, most visited child is not a viable option, the AUV will use the maximum reward child since there is no additional time for more iterations of the MCTS. The secondary option gives the AUV the best child through the maximum reward value, represented by the accumulation of the *exploration* and *exploitation* scores over the MCTS.

Improving MCTS: Biasing

Biasing the MCTS affects the action selection criterion by changing the c value in the MCTS equation 7.4. As the c value increases, the MCTS selects new areas to explore, weighting exploration of the tree more heavily. As the c value decreases, the MCTS acts more greedily, and instead chooses the next child based on which gives the highest reward (exploitation).

Previous work selected $c = 1/\sqrt{2}$ to satisfy the Hoeffding inequality for a reward within [0,1] [7], which also is in line with the scores of *exploration* and *exploitation*.

In this thesis we explore an avenue to improve the results of the MCTS by using an adaptive

c value, specifically,

$$c = \left(\frac{k_1}{k_2}\right)^{\eta}, \tag{7.5}$$

where k_1 is the gain value for *exploration*, k_2 is the gain value for *exploitation*, and η is a constant. By making the c value proportional to the ratio of *exploration* to *exploitation*, the MCTS parallels the desire of the AUV for *exploration* or *exploitation*.

7.4 Partially Observable Monte Carlo Planning

The combination of the POMDP and MCTS yield the POMCP, as depicted in Figure 7.4. The search horizon, illustrated with the red, yellow, green, and pink trajectories, depicts the POMDP consisting of four trajectories and the planning horizon portrayed in blue represents the MCTS equal to nine trajectories. The large decision space up to the planning horizon leads to a large computational complexity, with each additional search horizon exponentially adding more computational cost. However, the MCTS reduces the computational complexity to within the computational limit by advantageously searching the large decision space starting at the root node (AUV position) and terminating at the planning horizon through the child selection criterion and biasing. When one iteration of the search is complete, the reward for each individual trajectory is compiled using a discount factor, γ, to ensure that the cumulative reward is more dependent on closer trajectories.

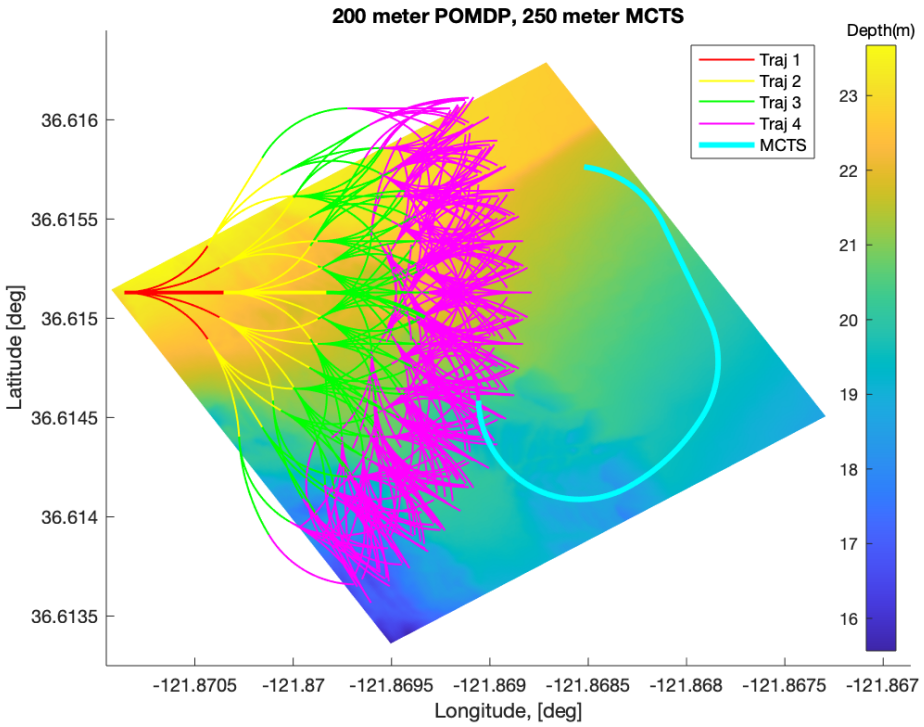


Figure 7.4. POMCP Visual Description of POMDP trajectories (red, yellow, green, pink) and one MCTS trajectory (teal).

Using a POMCP, a combination of the POMDP and MCTS, allows for a large decision space to be advantageously searched within a computational limit, and permits the planning process to look over a larger distance. With longer possible paths to evaluate, the AUV has the ability to better position itself to explore or exploit the environment and make a better decision for the current move based on potential future moves.

7.5 Conclusions

The POMCP allows for the determination of a viable solution within a large decision space that is formed while conducting trajectory planning by significantly lowering the computational complexity. By selectively biasing the POMCP, the trajectory planner becomes better at finding the best solutions in a more expedited timeframe, a concept that can be better portrayed by results shown in the next chapter.

CHAPTER 8:
Results and Evaluation

To accomplish autonomous navigation, the AUV must primarily be able to navigate without external localization sources or an *a priori* map. Additional secondary objectives include minimizing total positional error, completely covering a defined region with a designated bathymetric sensor, creating an accurate bathymetric map, and finishing the task within time and energy constraints. The primary and secondary objectives can be accomplished through the use of a position estimator and leveraging *exploration* and *exploitation* through a mathematical framework called a reward function. The reward function is then employed through a POMCP, which scales down the computational complexity while biasing the search to focus on the most advantageous areas. This chapter establishes a baseline metric for a coverage problem, compares the three reward functions discussed in Chapter 6, and evaluates the trajectory planner using results of the POMDP, unbiased POMCP, and the biased POMCP.

8.1 Autonomous Underwater Vehicle Specifications

The AUV specifications used to compile the results are listed in Table 8.1.

Table 8.1. AUV Model Parameters.

Heading Rates	$[-2\ ,-\frac{4}{3}\ ,-\frac{2}{3}\ ,0\ ,\frac{2}{3}\ ,\frac{4}{3}\ ,2]\ ^{o}/s$
POMDP Horizon	3 trajectories
MCTS Horizon	10 trajectories
AUV INS Error	0.5% distance traveled
AUV speed	3.5 knots
Trajectory length	50 meters
AUV Allowed Error	1.5 m
Number of Particles	1000

8.2 Benchmark

There must be metrics in place on which to judge the trajectory planner prior to discussing the results of the trajectory planner. This section discusses the appropriate benchmark through calculating the minimum time, discussing common mapping techniques and their respective abilities, and using more computationally complex methods with a perfect map to determine the best solution.

8.2.1 Optimal Solution

The best coverage path length can be determined with the equation $A = SL$, where S is the sensor coverage size, L is the number of discretized points over the entire path length, and A is the search area. The sensor size can be represented as two intersecting circles. As the AUV moves from one discretized step to the next on a trajectory, there will be some margin of sensor overlap. Therefore, the sensor size can be represented as the change in area of two intersecting circles [91],

$$S = \pi R^2 - 2R^2 \cos^{-1}\left(\frac{d}{2R}\right) + \frac{1}{2} d\sqrt{4R^2 - d^2}, \tag{8.1}$$

where R is the radius of the circle, and d is distance between the center of each circle. The overall best time to search an area can be determined using the speed of the AUV and number of discretized points L. In this case, the optimal time is $\approx$ 17 min with a sensor radius of $10m$ and a speed of 3.5 knots. The optimal time is most likely not achievable based on vehicle motion dynamics and further does not allow the AUV to localize as no area of the map is re-visited. However, it is still a useful performance metric for the AUV trajectory planner as the ideal solution without localization.

8.2.2 Common Techniques

Three prevalent techniques to perform mapping of an area exist: box in mapping methodology (BI), spiral out mapping methodology (SO), and mow-the-lawn mapping methodology (MtL). Each method was performed with an AUV with identical speed and sensor specifications, specifically 3.5 knots and a 10 meter sonar sensor range, that performed the mapping of the 255m $\times$ 255m area with no sensor overlap of previously mapped areas and another with half sensor overlap of previously mapped areas to provide a change for

localization, as shown in Figure 8.1.

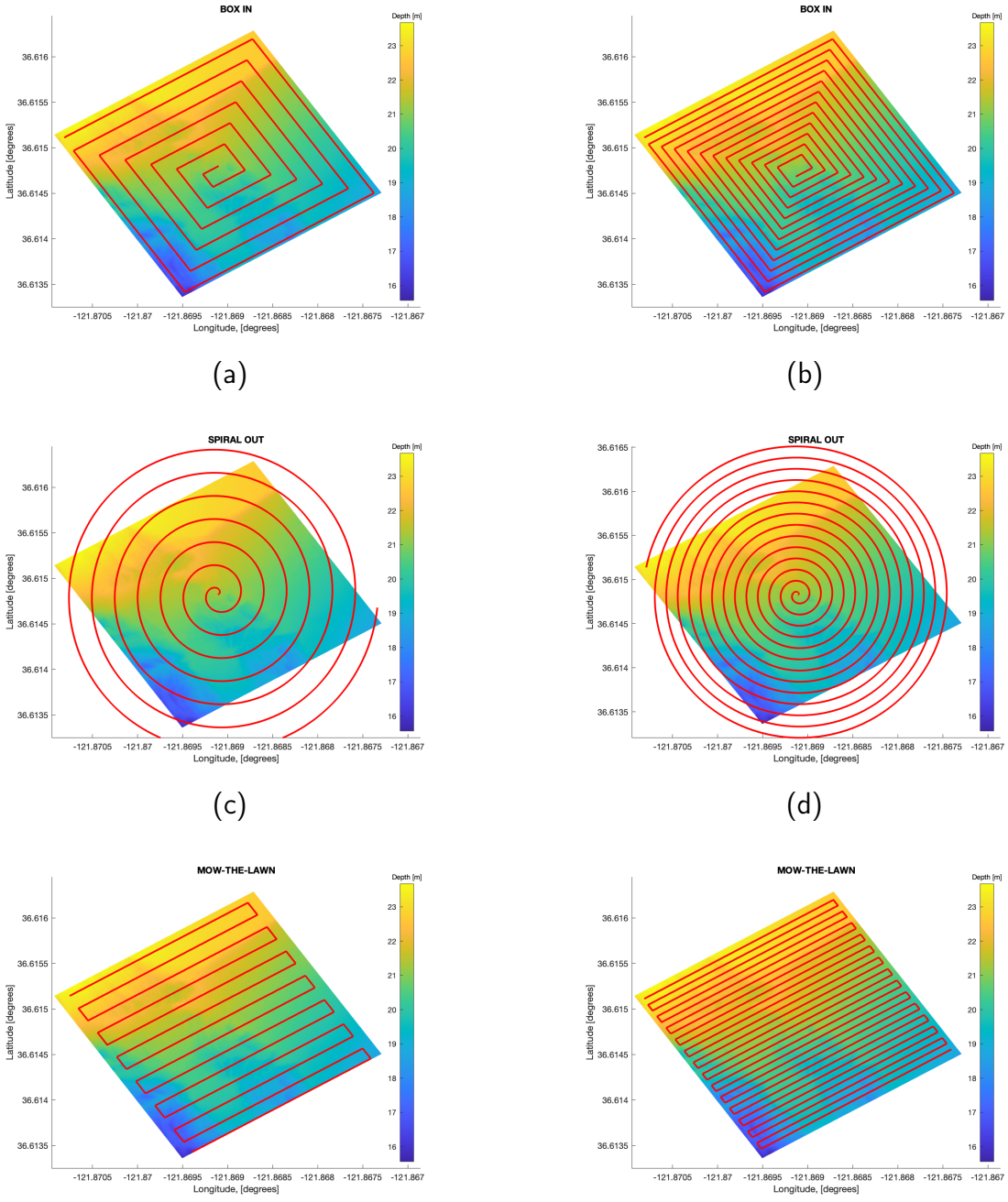
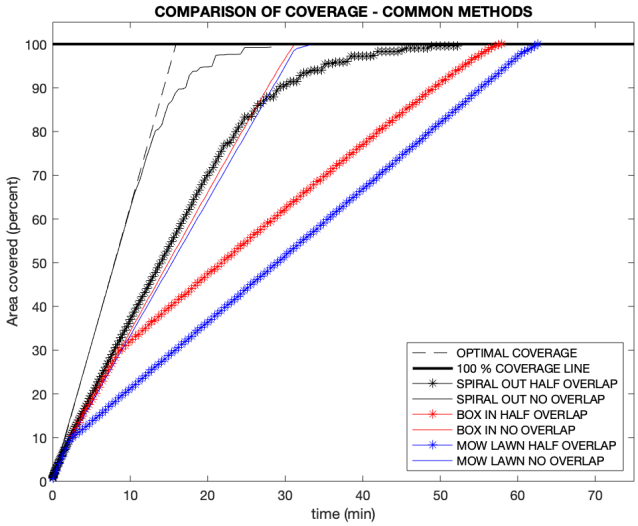
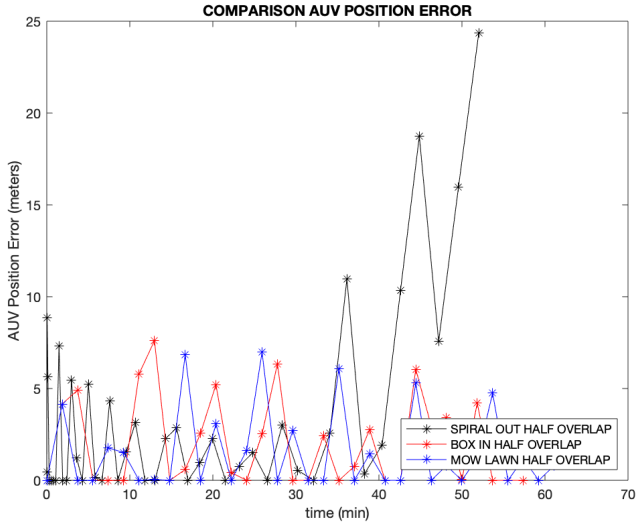


Figure 8.1. Common Mapping Techniques: (ab) BI, (cd) SO, and (ef) MtL. (ace) No Sensor Overlap (bdf) 50% Sensor Overlap.

The map area gets fully covered by each mapping technique, as illustrated in Figure 8.1. The left column of Figure 8.1 shows no sensor overlap, the quickest way to cover the entire area, whereas the right column shows a 50 % sensor overlap, an attempt to provide a balance between *exploration* and *exploitation*. The coverage at each time in the trajectory, along with the AUV position error is shown in Figure 8.2.



(a)



(b)

Figure 8.2. Common Mapping Techniques Comparison in (a) Coverage and (b) AUV Position Error.

Looking at Figure 8.2 (a), the optimal solution (discussed in Section 8.2.1) maintains the best time to cover the specified area. The SO method with no overlap provides the best initial coverage by always exploring new regions and keeping the optimal solution until the AUV starts going over areas that are not required to be mapped. The BI and MtL methods with no desired sensor overlap are very similar, as each has minimal sensor overlaps when moving from one trajectory to another. However, there is no ability for the AUV to localize with no overlap, leading to higher AUV position error. Therefore, 50% sensor coverage trajectories were implemented to give the ability for localization, with SO performing the best for coverage, followed by BI and MtL, respectively. The BI method performs better in coverage than MtL as BI utilizes a full sensor over unexplored space for a longer duration prior to starting the 50% sensor coverage routine (AUV goes around the map fully in BI as opposed to back-and-forth in MtL).

Figure 8.2 (b) depicts the AUV position error as a function of time, in which only 50% sensor overlap methodologies are included. The 100 % sensor coverage over new areas does not provide the ability to localize position, and therefore the error increases linearly as a function of distance. Of the three 50 % sensor coverage methods, BI and MtL are similar, with SO not performing as well. However, for these cases, the UV error is a function of the map terrain. Since the UV has a predetermined path, the UV cannot exploit areas that it already knows, and must get "lucky" for localization to occur. Regardless, results show AUV position errors obtained for the three 50% sensor coverage methods are above the 1.5m specification indicated in Table 8.1, showing that predetermined paths are not a viable methodology.

Difficulties with Common Techniques

The difficulties presented with the common techniques are two-fold: maneuverability of the AUV and map dependency.

First, the AUV cannot pivot ninety degrees instantaneously, requiring at a minimum, a rudder angle based arc to maneuver which leads to the inability of the AUV to follow the BI or MtL trajectories. Further, the shape of the search area adds an additional hindrance to the AUV, providing corners and straight edges that the AUV doesn't have the maneuverability to properly handle. The only feasible trajectory, SO, covers areas outside the desired region, and once the AUV gets to the outermost edge of the spiral, the time to fully spiral around

grows significantly, making it a less desirable approach for larger areas.

In the 50% sensor coverage case, the AUV has the ability to localize using previously explored areas of the map. However, by performing a predetermined pattern of movement, the AUV is unable to find and exploit the environment and is therefore dependent on the map features.

8.2.3 Brute Force Results

The computational requirement to solve this problem, even with the restrictions listed in Table 1.1, leads to an evaluation of $\approx 10^{46}$ trajectories, as shown in Section 7.1. The immense computational cost to determine the optimal solution was too large using resources available and instead we employed an alternative to determine an optimal trajectory within computational requirements of an intel XEON processor. Instead of determining the optimal solution, a six-horizon POMDP—more horizons than computationally feasible in real time for the REMUS vehicle—coupled with a perfect map was used to determine an optimal trajectory within computational limits. For comparison, one iteration of the six-horizon POMDP was completed in 32.2 min while an AUV is able to traverse a 50m trajectory in 32.3 seconds. The six-horizon POMDP was run using all three reward functions starting at the same location, with the optimal estimator reward function providing the best coverage and AUV position error.

8.2.4 Establishing the Benchmark

Results obtained with the methods previously described are compiled in Figure 8.3 and Table 8.2 and are compared to evaluate an appropriate benchmark.

Figure 8.3 depicts the coverage of the six-horizon POMDP with a perfect prior map, SO 50% sensor overlap, BI 50% sensor overlap, and MtL 50% sensor overlap, with the six-horizon POMDP with a perfect prior map performing the best from a coverage standpoint.

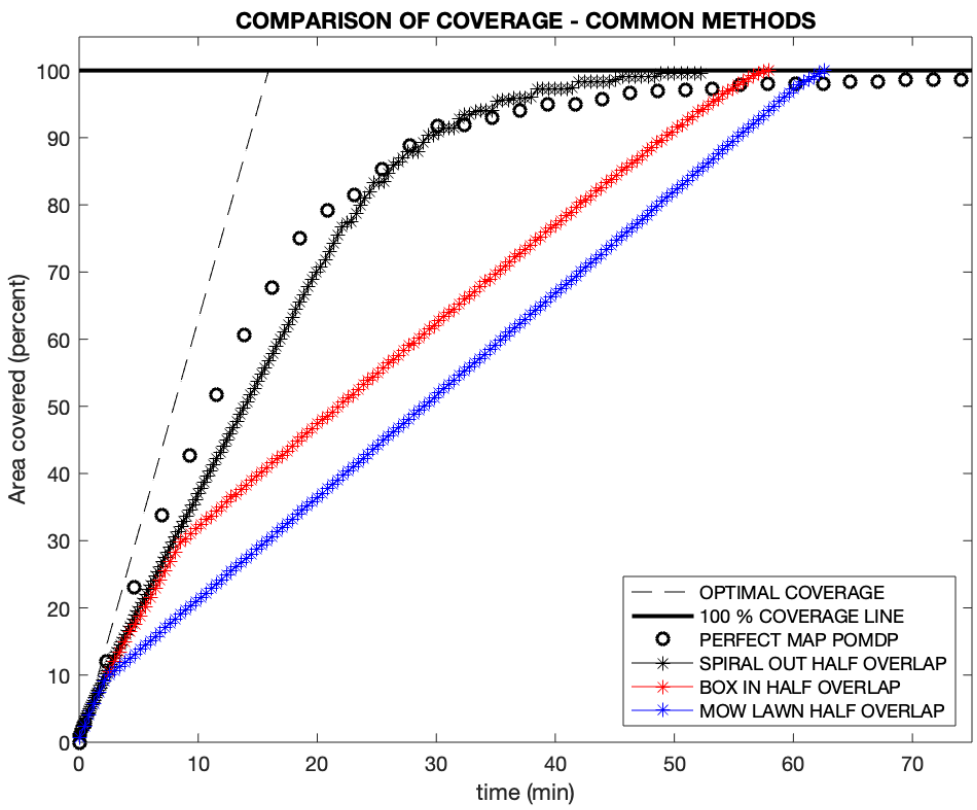


Figure 8.3. Comparison of Coverage Techniques between the six-horizon POMDP with perfect map, box in (BI), spiral out (SO), and mowing the lawn (MOW) methods with 50% sensor overlap.

Referring to Figure 8.3, the optimal solution and six-horizon POMDP with a perfect map were chosen as the appropriate benchmarks for coverage comparison. While the optimal solution is likely not feasible, it provides the best solution in terms of coverage based on AUV parameters. The six-horizon POMDP was also selected as it led to the best solution among all techniques investigated that allow for localization.

Table 8.2. Comparison of AUV Positional Error Using Coverage Techniques listed in Figure 8.3.

AUV Error (meters)		
Method	Avg	Max
POMDP	0.19	1.2
SO	-	>10 *
SO 50%	3.5	>10 *
BI	-	>10 *
BI 50%	1.8	7.6 *
MtL	-	>10 *
MtL 50%	1.5	6.9 *

Table 8.2 shows the AUV position error for each method used in Figure 8.3. Results show the six-horizon POMDP gives the most realistic answer in terms of coverage and AUV position error, and will be used as the benchmark for this thesis.

8.3 Reward Function Determination

This Section discusses the three different reward functions presented in this thesis and shows results of each reward function using the POMDP and the unbiased POMCP.

8.3.1 POMDP

Ten simulations of the POMDP for each reward function was implemented using random starting points. The coverage and AUV error obtained for ten simulations was recorded to first evaluate the coverage and AUV position error. Shown in Figure 8.4 and Table 8.3, the mean and 95% confidence interval (CI) were determined for the coverage and AUV error for each simulation, and the maximum AUV error was also recorded.

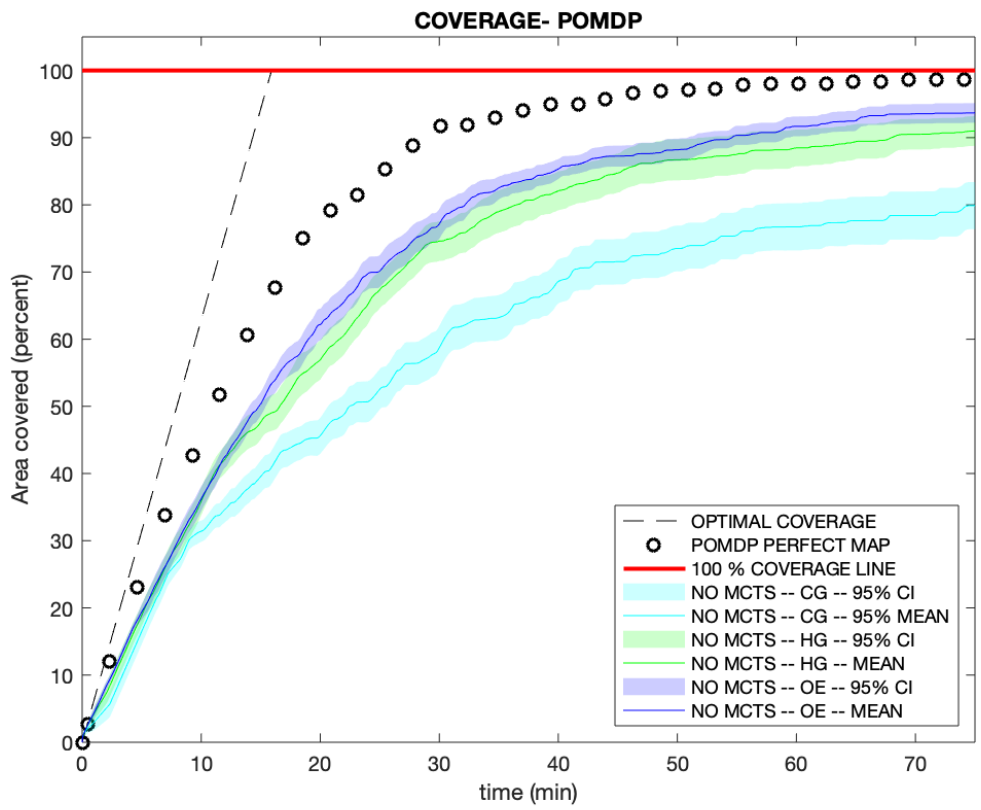


Figure 8.4. POMDP Results for Coverage of Three Reward Functions Presented in Chapter 6: CG – Constant Gain (Section 6.1.1), HG – Heuristic Gain (Section 6.1.2), and OE – Optimal Estimator (Section 6.1.3).

From a coverage standpoint results show the heuristic gain (HG) and the optimal estimator (OE) have similar performance (OE performing slightly better) and both methods surpass constant gain coverage as illustrated in Figure 8.4. Further, the POMDP coverage results plateau at around 90% coverage, struggling to find new areas to explore from 30 min to 80 min due to the inability to look further ahead toward trajectories that achieve complete coverage.

Table 8.3. POMDP Results AUV Position Error of Three Reward Functions Presented in Chapter 6: CG – Constant Gain (Section 6.1.1), HG – Heuristic Gain (Section 6.1.2), and OE – Optimal Estimator (Section 6.1.3).

POMDP AUV Error (meters)			
Gain Type	Avg	95% CI	Max
CG	0.25	0.46	1.47
HG	0.22	0.34	1.40
OE	0.19	0.29	1.14

All three reward functions led to a maximum position error below the required 1.5m maximum. However, the average, 95 % CI, and maximum position error between the three different reward functions, as shown in Table 8.3, display that the OE gain type leads to the lowest average, smallest 95% CI, and lowest maximum.

8.3.2 POMCP

Ten simulations of the POMCP were then run from random poses to determine coverage and AUV position error, as shown in Figure 8.5 and Table 8.4, respectively.

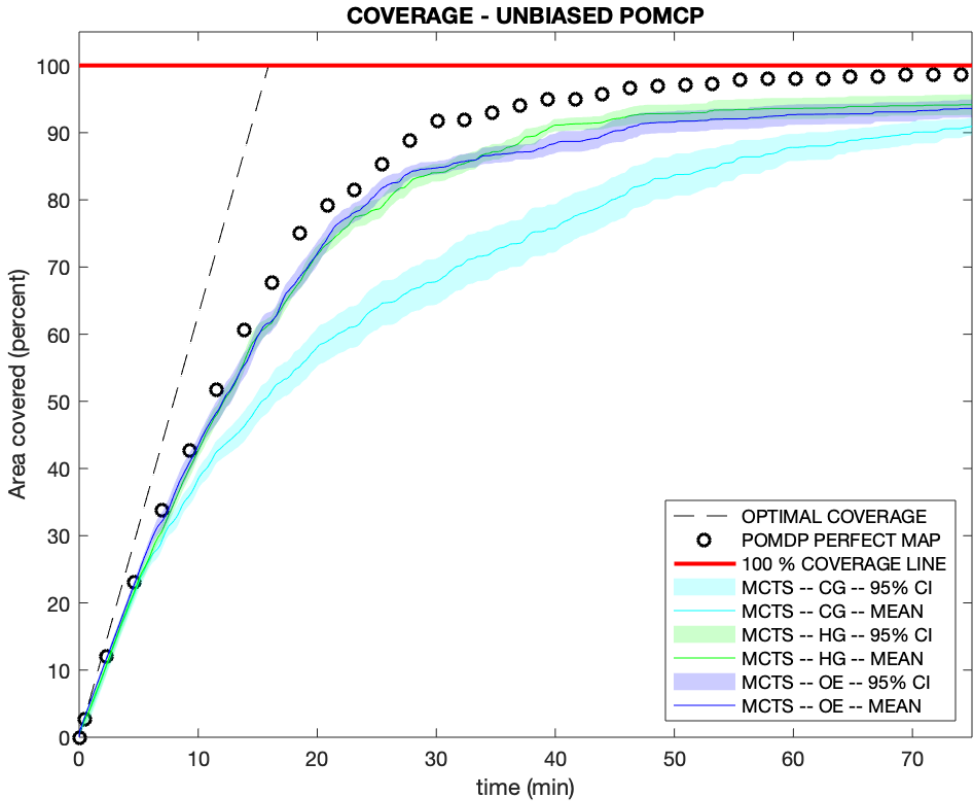


Figure 8.5. Unbiased POMCP Results for Coverage of Three Reward Functions Presented in Chapter 6: CG – Constant Gain (Section 6.1.1), HG – Heuristic Gain (Section 6.1.2), and OE – Optimal Estimator (Section 6.1.3).

Results show HG provided the best coverage, with OE very close behind. However, the map area coverage still only reaches a maximum of around 93 % at around 80 min of the AUV search.

Table 8.4. Unbiased POMCP Results for AUV Position Error of Three Reward Functions Presented in Chapter 6: CG – Constant Gain (Section 6.1.1), HG – Heuristic Gain (Section 6.1.2), and OE – Optimal Estimator (Section 6.1.3).

Unbiased POMCP AUV Error (meters)			
Gain Type	Avg	95% CI	Max
CG	0.20	0.32	1.46
HG	0.22	0.35	1.61
OE	0.18	0.31	1.19

The main difference between results obtained with the HG and OE gain values comes down to the AUV position error, shown in Table 8.4. The OE maintains significantly lower AUV position error while also not exceeding the maximum specified error of 1.5 meters.

Overall, the best method for the reward function determination is through optimal estimator based on the coverage and AUV position error of the three reward functions within the POMCP framework, .

8.3.3 Reward Function Conclusions

Three different reward functions were evaluated through three different scenarios:

- *a priori* perfect map, six-horizon POMDP (Section 8.2.3)
- unknown map, three trajectory POMDP (Section 8.3.1)
- the unknown map, POMCP utilizing three trajectory POMDP and ten trajectory MCTS (Section 8.3.2).

Results showed the OE to be the best overall reward function in each of the three scenarios based on the coverage and AUV performance.

8.4 Partially Observable Monte Carlo Planning

It is also important to determine whether the MCTS is worth the additional computational load by comparing unbiased POMDP and unbiased POMCP results. The coverage comparison for both the unbiased POMDP and unbiased POMCP utilizing all reward functions is shown in Figure 8.6.

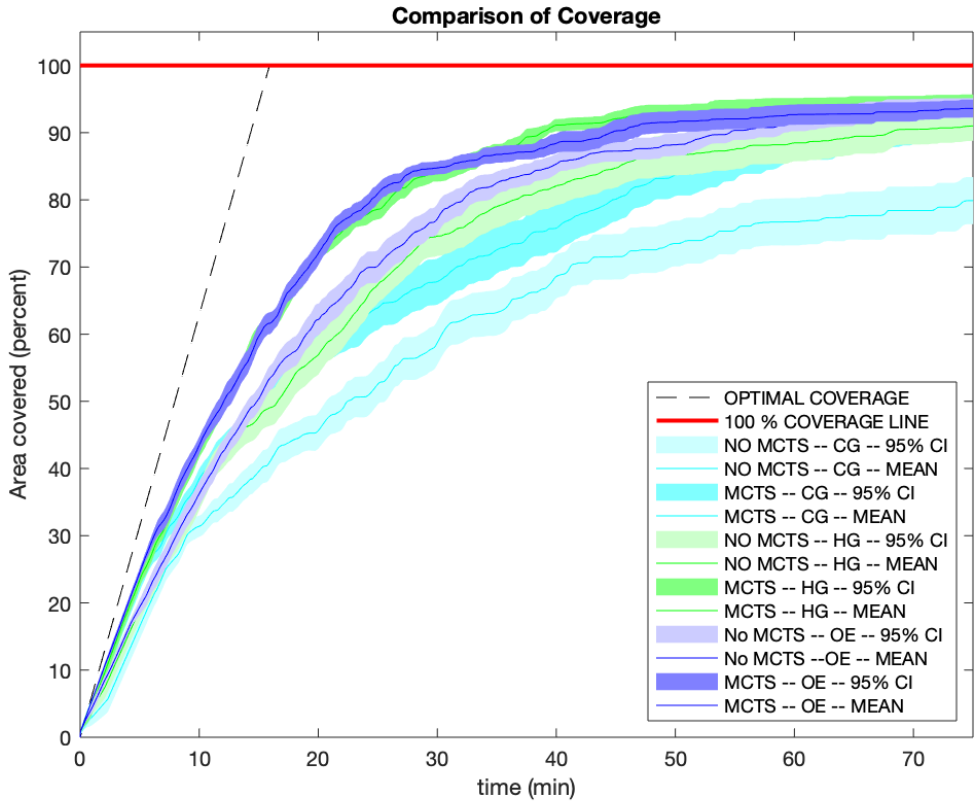


Figure 8.6. Comparison of POMDP and POMCP Coverage Performance.

As seen in Figure 8.6, the addition of the MCTS allows the POMCP to attain better coverage than that without the MCTS. Furthermore, the addition of the MCTS also lowered or maintained the AUV position mean error throughout the entire trajectory.

Overall, the MCTS only adds a computational cost as opposed to a temporal cost since the computational cost does not exceed the time required for the AUV to traverse a 50m trajectory. The MCTS only searches using the time remaining prior to the next decision,

and therefore is an improvement over a non-MCTS approach.

8.5 Biased Partially Observable Monte Carlo Planning

The MCTS balances *exploration* and *exploitation* through a constant c [6],

$$Q^*(s,a) = Q(s,a) + c\sqrt{\frac{lnN(s)}{N(s,a)}}. \tag{8.2}$$

By increasing the c value, more *exploration* of the MCTS occurs. By decreasing the c value, more *exploitation* of known areas of the current search occur. Using ten scenarios starting from random poses and the optimal estimator (OE) method for the reward function, this thesis proposes a MCTS biasing option through a ratio of *exploration* and *exploitation* gain values,

$$c = \left(\frac{k_1}{k_2}\right)^{\eta}. \tag{8.3}$$

Equation 7.5 is evaluated for each of the following η values, $\eta = [-5, -1, 0.5, 1, 2, 3, 4, 10]$, in which the coverage and AUV position errors were recorded. For clarity, only η values of $\eta = [-5, 0.5, 1, 10]$ were plotted in Figure 8.7.

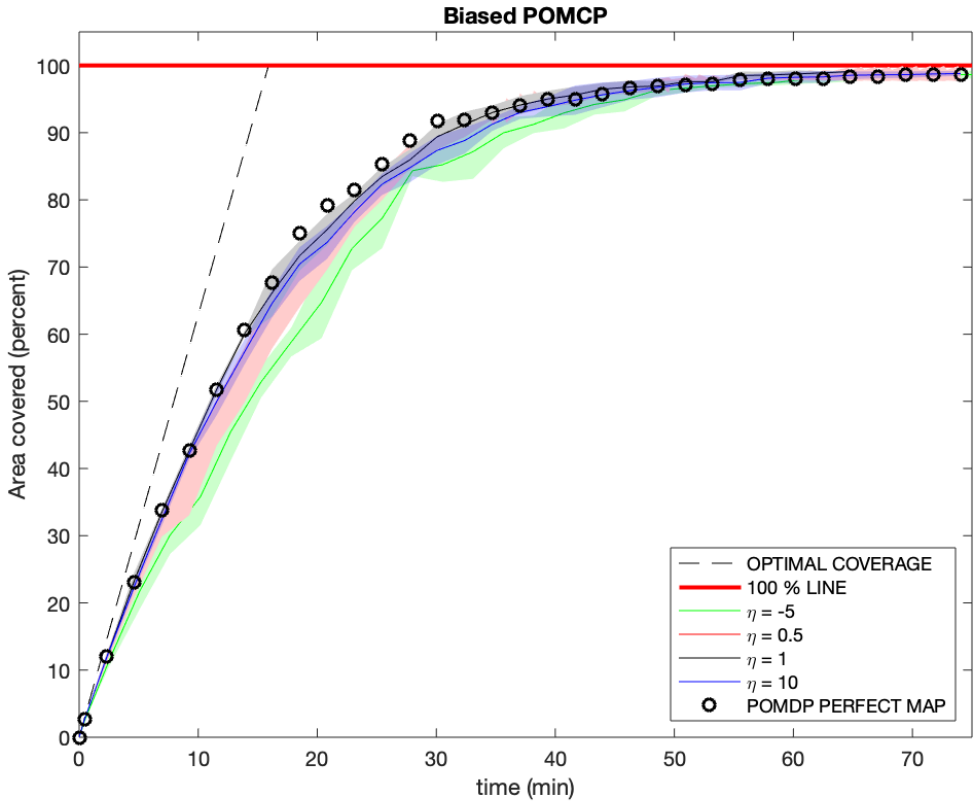


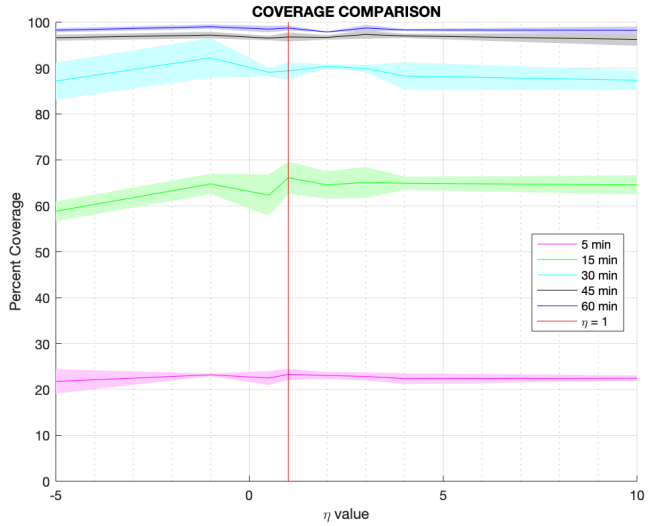
Figure 8.7. Biased MCTS Mean and 95% CI Coverage Results for $\eta = [-5, 0.5, 1, 10]$.

Table 8.5. Biased POMCP Results for AUV Position Error.

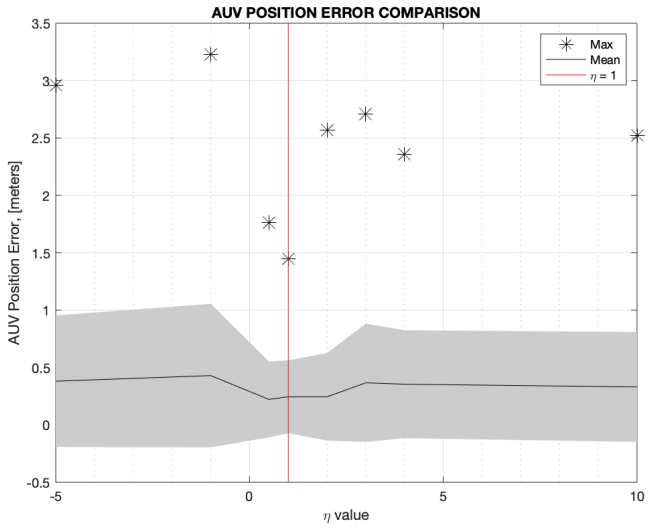
Biased POMCP AUV Error (meters)			
η	Avg	95% CI	Max
0.5	0.22	0.33	1.76
1	0.23	0.31	1.45
10	0.33	0.47	2.52
-5	0.38	0.57	2.96

Referring to Figure 8.7 and Table 8.5, the best coverage and AUV position error occur for $\eta = 1$, closely following the six-horizon, perfect map POMDP in coverage and maintaining a slightly higher average and maximum AUV position error (POMDP AUV position mean 0.19 and average 1.2), but still within the overall specification of 1.5 meters.

All η values, $\eta = [-5, -1, 0.5, 1, 2, 3, 4, 10]$, were then used for a basis of comparison for coverage and AUV position error, as shown in Figure 8.8.



(a)



(b)

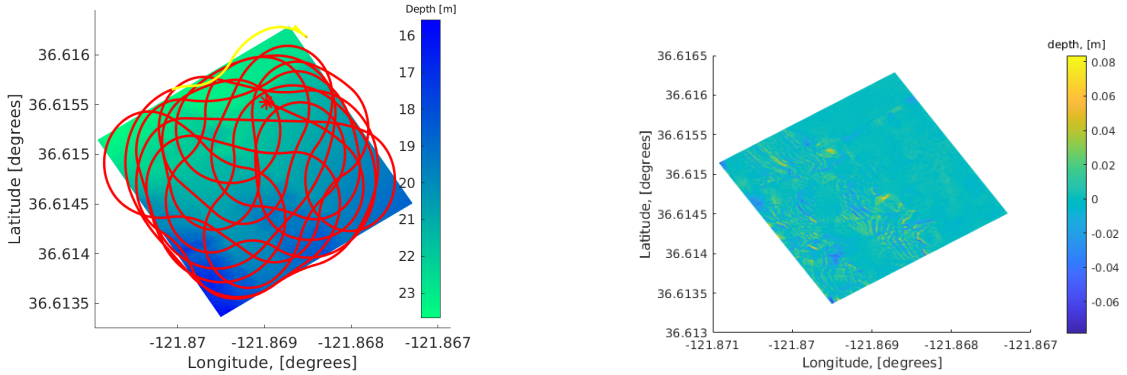
Figure 8.8. MCTS Bias η-Value Comparison. (a) Mean and 95% CI Coverage for $\eta = [-5, -1, 0.5, 1, 2, 3, 4, 10]$. (b) Mean, 95% CI, and maximum AUV Position error for $\eta = [-5, -1, 0.5, 1, 2, 3, 4, 10]$.

Referring to Figure 8.8, it is noted that as the η values get further from $\eta = 1$, the coverage 95 % CI, the AUV position error 95% CI width, and the maximum value AUV position error, all increase. Therefore, $\eta = 1$ was chosen as the best biasing option for coverage and

AUV position error considerations.

8.6 Best Trajectory from Trajectory Planner

The POMCP was run for $\eta = 1$ using the OE reward function, achieving full coverage at 7250 meters, or approximately one hour. The step-by-step trajectory is provided in Appendix E, with the final trajectory and the AUV generated map error shown in Figure 8.9.



(a) (b)

Figure 8.9. (a) Biased POMCP Result Trajectory and (b) AUV Map Error comparison to CSUMB Map [78].

The overall trajectory resulted in a maximum AUV position error of 1.47 meters, mean AUV error of 0.22 meters, with a 95 % CI of 0.33 meters, narrowly achieving the 1.5 meter error specification. Furthermore, the total map error, shown in Figure 8.9(b), has a maximum error of 0.08 meters, which is within the specified sensor error.

CHAPTER 9:
Conclusions and Future Work

The requirement for accurate bathymetric charts for the open oceans, ports, and channels has never been more apparent. The oceans cover seventy-one percent of the surface of the Earth [92], provide a means of transportation for ninety percent of world goods and resources [93], and play a part in approximately twenty one trillion U.S. dollars per year to human welfare (sixty percent from coastal and shelf systems and forty percent from open oceans) [92]. This thesis proposes ATAN within the underwater domain as a solution to quickly, efficiently, and accurately map areas while not requiring an *a priori* map or external localization sources.

9.1 Thesis Conclusions

This thesis demonstrates the ability of an AUV to navigate autonomously without external localization sources or an *a priori* map while also fulfilling the following objectives:

- minimizing total positional error
- completely covering a specified area with a designated sensor
- creating an accurate map within certain AUV position error
- completing the mapping problem within time and energy constraints.

Results show that (1) the AUV total positional error was minimized to under 1.5 meters; (2) the full area was accurately covered, having a maximum error of 0.08 meters when compared to the real map; and (3) the AUV was able to map the entire area in approximately one hour, outperforming larger computational cost methods.

9.2 Thesis Contributions

This thesis demonstrated the viability of ATAN for autonomous vehicles and provides the following contributions to the area:

- A real-time information-theoretic trajectory planner that adaptively leverages *exploration* and *exploitation* through a POMCP

- A real-time implementation of the GPM that maximizes the information gain of the selected trajectory
- A real-time mapping solution that does not require a prior map or external localization systems
- An analytical framework that allows for inspection and evaluation of different approaches within ATAN.

The autonomy of a UV can be measured through three constructs: energy, position estimation, and the ability to make decisions [94]. ATAN coupled with the REMUS system provides a capable energy source, accurate position estimation, and the ability to interpret, interact, and decide trajectories in new environments. Through this definition, ATAN emphasizes greater levels of autonomy for UVs and improves upon the foundation and application of TAN.

9.3 Recommendations for Future Work

This thesis covers multiple subject areas. Therefore, considerations for future work are broken up into sections parallel to the chapters of this thesis and also includes a section on computational complexity of this problem and obtaining in-water results.

9.3.1 Position Estimator

The position estimator uses a particle filter since ATAN is a non-linear, non-Gaussian problem. The particle filter characteristics can be potentially improved by refining its parameters.

The particle filter was modeled using a circular sonar sensor as opposed to a line sensor that replicates the BlueView MB 2250. Using a line sensor instead of a circular sensor makes the particle filter correlation method a much harder problem, requiring a more accurate initialization and update process. This thesis attempted to fulfill the line sensor model; however, we were unable to implement it with reasonable accuracy. Results showed the initialization and first 1000 meters was fairly consistent with position estimation (error under three meters). However, the PF was unable to recover once the PF converged to an incorrect solution. Future work within the linear sonar sensor could include the modeling of the linear sonar sensor using a ray trace algorithm coupled with the INS system to better

correlate terrain while minimizing the probability of a false PF convergence, using [29] as a starting point.

This thesis considered Gaussian probability distributions within the modeling of the particle filter, yet the particle filter is capable of handling non-Gaussian, multi-modal functions. Utilizing different types of probability distributions, or different selection criteria for the PF modes could increase PF performance.

For different terrain classifications (e.g., flat, rough, soft, and hard terrain), an adaptive particle filter approach may lead to better position estimation. Using a combination of adaptive probability distributions with the terrain coupled with [31] may improve the overall PF performance.

Lastly, there are numerous types of particle filters. In this thesis we considered the MCL variety, but a comparative analysis between different particle filters like the MCMC method or Rao-Blackwellized PF from a computational cost standpoint and positional estimate accuracy would help decide the optimal position estimation techniques to be used within ATAN.

9.3.2 Exploration

The *exploration* component of ATAN comprises of a GPM and a trajectory evaluation component that uses the GPM and KLD to determine an *exploration* score. Future improvements on the *exploration* component include different modeling of the semi-variance, utilizing different covariance matrix functions, and clustering and combining areas of likeness together into separate GPMs.

The semi-variance can be represented using an epi-spline which can then be optimized using individual sections [95], which may lead to a more accurate GPM.

In this thesis we selected the Matérn function to calculate the covariance matrix based on its spatial properties [81]. However, there are various covariance matrices that can be used which may be more effective for the ATAN problem.

Similar terrain can be modeled in a similar fashion. Using a different GPM for areas may lower computational cost and also increase the overall spatial estimation that occurs within

the GPM. By modeling smaller areas as separate GPMs, the computational complexity decreases. Further, with different models, each model can be refined to its specific terrain type, possibly making it more accurate. A possible approach to combining GPMs is through a Bayesian Committee Machine (BCM) [41]. Using the combination of multiple smaller GPMs would reduce the overall computational complexity and potentially result in a more accurate GPM.

9.3.3 Exploitation

Exploitation uses a MAP estimate to evaluate trajectories using the best hypothesis for position estimation to determine localization through terrain evaluation. This thesis uses a CDF function based on entropy values of the terrain to determine localization probability. Refining this model to a linear or non-linear function that quantifies the localization probability as a function of entropy using actual terrain data would potentially increase the accuracy of ATAN within all environments.

Additionally, the terrain evaluation through the exploitation phase can be broken up into two different methodologies. In this thesis we considered a distance approach, sampling the terrain at constant distances over the trajectory to determine the best trajectory for localization. Another methodology is using cell decomposition and using every cell the trajectory passes over to determine the exploitation score which may yield more accurate results.

Lastly, in this work, we utilized a MAP estimate for trajectory evaluation instead of a MLE which accounts for uncertainty of the vehicle position. Using a stochastic system to quantify the error associated with the Boltzmann entropy through the use of a GPM could give higher fidelity on exploitation planning. However, the computational complexity of the problem will increase dramatically by adding an additional GPM.

9.3.4 Trajectory Planner

The trajectory planner utilizes a POMCP to generate and evaluate trajectories. While this approach was shown to be successful, there are multiple ways to increase the ability of the UV to complete the coverage problem in a faster time and with lower vehicle error.

First, ATAN can be improved with inspection planning in the terminal stages. When the coverage reaches a certain limit, the POMCP can switch over to an inspection planning algorithm to selectively cover the remaining areas.

Additionally, in this thesis we used a constant trajectory of fifty meters and constant heading rate changes throughout the entire problem. Another improvement of the POMCP can be the biasing of the trajectory length and heading rates as a function of *exploration* and *exploitation*, or the optimization of trajectory lengths and heading rates as a function of *exploration* and *exploitation*.

Furthermore, the POMCP utilizes heading rates instead of the preferred straight line paths for conducting mapping. The POMCP can be easily adapted to evaluate trajectories that perform quick maneuvers followed by straight paths to optimize the sensor aspect. The harder development would be the implementation of a sensor system that can perform mapping while the sensor is turning.

Lastly, in this thesis we used the maximum reward, most visited child as the selection criterion for the MCTS. The ability to quickly cover an area while minimizing AUV position error may improve if an adaptive approach to selection is used instead of a fixed child.

9.3.5 Computational Complexity

In this thesis we utilized multiple platforms ranging from Intel i9 and Intel i10 processors to Amazon Web Services (AWS) Elastic Computer Cloud (EC2) nodes in order to compile the data used for analysis. After multiple runs on the Intel i9 processor, an average number of simulations within the MCTS was determined and used for all data collection on the various platforms, with only ten simulations for each trial achieved. Future work could consider a higher number of simulations to determine a better biasing for the MCTS which may include a function for c based on time or the requirement for *exploration* or *exploitation*.

Future work is needed to evaluate the optimal trajectory based on running all trajectories discussed in Section 8.2.3. In this thesis, we attempted to determine the optimal trajectory but were unable to due to limited resources and time.

9.3.6 In-Water Results

Lastly, and most importantly, future work can be used to provide in-water results on the REMUS vehicle, using ATAN. While the simulation of ATAN proved successful, an in-water result would solidify ATAN as the way to achieve autonomy in the future.

9.3.7 Applications Outside of the Undersea Domain

In this thesis we model ATAN through the use of an AUV. However, ATAN is not limited to the undersea domain and has applicable approaches to any environment. Another aspect of future work revolves around incorporating ATAN into different types of UVs. The *exploration-exploitation* interaction remains the same, but the parameters of the vehicle and terrain will be matched to different environmental and technological constraints.

9.4 Concluding Remarks

ATAN takes advantage of the *exploration-exploitation* dilemma in real time through the use of a POMCP. In this thesis we provide notable contributions to the field of robotics, and provide a path forward to further improve ATAN. Through ATAN and these improvements, greater levels of autonomy are exhibited, improving upon the TAN framework but also providing a larger offering to the field of robotics.

APPENDIX A: Variogram

This appendix shows the mathematical relationship between the variogram and the covariance.

A.1 Variables

The following variables are used:

- h is the relative distance between the test point and the measured points
- $C(h)$ is the covariance as a function of h
- x is the location of the test or measured points
- $2\gamma(h)$ is the variogram function
- $Z(x)$ is the observation at location x
- m_x is the mean observation, z, based on location, x

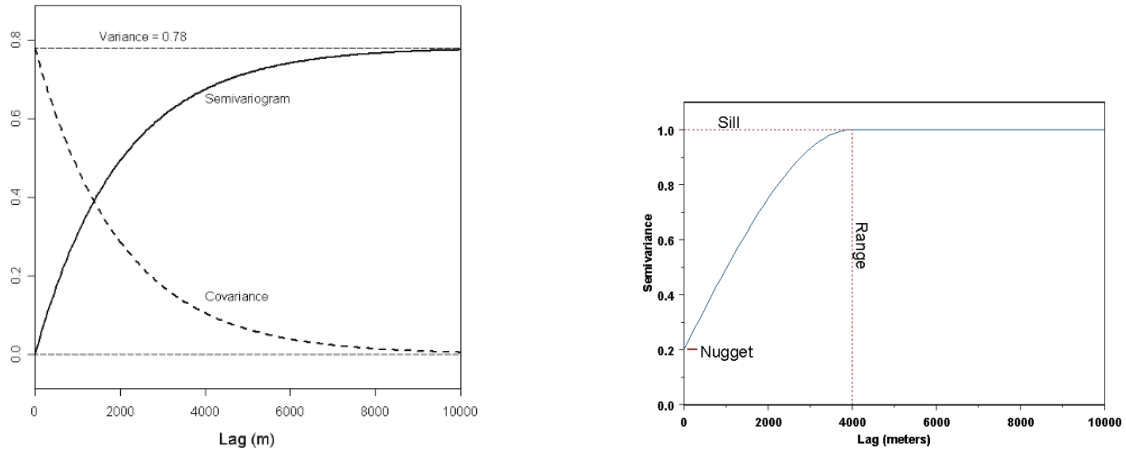
A.2 Variogram

With the assumption of second-order stationarity, the covariance $C(h)$ and variogram $2\gamma(h)$ can be written as [80]

$$C(h) = \mathbb{E}[Z(x+h) \cdot Z(x)] - m_x^2, \qquad \forall x \tag{A.1}$$

$$2\gamma(h) = \mathbb{E}[Z(x+h) - Z(x)]^2 = C(0) - C(h) \qquad \forall x. \tag{A.2}$$

The variogram asymptotically approaches the sill, which is equal to the initial covariance [80], as shown in Figure A.1.



(a) (b)

Figure A.1. (a) Variogram-Covariance Relationship. (b) Variogram Terminology. Source: [82]

Looking at Figure A.1, the maximum value of the variogram can be written as [80],

$$\gamma(\infty) = \mathbb{V}(Z(x)) = C(0). \tag{A.3}$$

This can then be utilized to solve for the covariance as a function of the variogram,

$$C(h) = \gamma(\text{inf}) - \gamma(h). \tag{A.4}$$

APPENDIX B:
Kullback Leibler Divergence

This appendix derives the KLD for two Gaussian distributions.

Distributions p(**x**) and q(**x**) are given as Gaussian distributions:

$$\mathrm{p}(\mathbf{x}) \sim N(\mu_1, \sigma_1^2) \tag{B.1}$$

$$\mathrm{q}(\mathbf{x}) \sim N(\mu_2, \sigma_2^2). \tag{B.2}$$

The KLD of two distributions can be written as:

$$KL(\mathrm{p}||\mathrm{q}) = \int \mathrm{p}(\mathbf{x}) \log\left(\frac{\mathrm{p}(\mathbf{x})}{\mathrm{q}(\mathbf{x})}\right) d\mathbf{x}. \tag{B.3}$$

A Gaussian distribution $\mathrm{y}(\mathbf{x}) \sim N(\mu, \sigma^2)$ can be represented through its probability density function:

$$\mathrm{y}(\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{\mathbf{x}-\mu}{\sigma})^2}. \tag{B.4}$$

Using equation B.4, $p(\mathbf{x})$ and $q(\mathbf{x})$ can be written using their mean and variance (equations B.1 and B.2):

$$\mathrm{p}(\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{1}{2}(\frac{\mathbf{x}-\mu_1}{\sigma_1})^2} \tag{B.5}$$

$$\mathrm{q}(\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{1}{2}(\frac{\mathbf{x}-\mu_2}{\sigma_2})^2}. \tag{B.6}$$

Inputting equations B.5 and B.6 into equation B.3,

$$KL(\mathrm{p}||\mathrm{q}) = \int \mathrm{p}(\mathbf{x}) \log\left(\frac{\frac{1}{\sqrt{2\pi\sigma_1^2}}\, e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\frac{1}{\sqrt{2\pi\sigma_2^2}}\, e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}}\right) d\mathbf{x}. \tag{B.7}$$

Simplifying equation B.7,

$$KL(\mathrm{p}||\mathrm{q}) = \int \mathrm{p}(\mathbf{x})\log\left(\left(\frac{\sigma_2^2}{\sigma_1^2}\right)^{0.5} \frac{e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}}\right)d\mathbf{x}. \tag{B.8}$$

Using the properties of the logarithm, equation B.8 can be re-written as

$$KL(\mathrm{p}||\mathrm{q}) = \frac{1}{2}\int \mathrm{p}(\mathbf{x})\log\left(\frac{\sigma_2^2}{\sigma_1^2}\right)d\mathbf{x} + \int \mathrm{p}(\mathbf{x})\left[-\frac{(x-\mu_1)^2}{2\sigma_1^2} + \frac{(x-\mu_2)^2}{2\sigma_2^2}\right]d\mathbf{x}. \tag{B.9}$$

The first term in equation B.9 includes a constant, $\log(\sigma_2^2/\sigma_1^2)$, which can be moved outside the integral, which leaves $\int p(\mathbf{x})d\mathbf{x} = 1$. Using the properties of logarithms, the first term can then be written as

$$\frac{1}{2}\int \mathrm{p}(\mathbf{x})\log\left(\frac{\sigma_2^2}{\sigma_1^2}\right)d\mathbf{x} = \frac{1}{2}\log\left(\frac{\sigma_2^2}{\sigma_1^2}\right). \tag{B.10}$$

The second term in equation B.9 is more complex:

$$\begin{aligned}\int \mathrm{p}(\mathbf{x})\left[-\frac{(\mathbf{x}-\mu_1)^2}{2\sigma_1^2} + \frac{(\mathbf{x}-\mu_2)^2}{2\sigma_2^2}\right]d\mathbf{x} =& \frac{1}{2\sigma_1^2}\left[-\int (\mathbf{x}-\mu_1)^2\mathrm{p}(\mathbf{x})d\mathbf{x}\right] \\ &+ \frac{1}{2\sigma_2^2}\left[\int (\mathbf{x}-\mu_2)^2\mathrm{p}(\mathbf{x})d\mathbf{x}\right].\end{aligned} \tag{B.11}$$

By definition, $\int(\mathbf{x}-\mu_1)^2 p(\mathbf{x})d\mathbf{x} = \sigma_1^2$ which leads to equation B.11 to be:

$$\int \mathrm{p}(\mathbf{x})\left[-\frac{(\mathbf{x}-\mu_1)^2}{2\sigma_1^2} + \frac{(\mathbf{x}-\mu_2)^2}{2\sigma_2^2}\right]d\mathbf{x} = -\frac{\sigma_1^2}{2\sigma_1^2} + \frac{1}{2\sigma_2^2}\left[\int (\mathbf{x}-\mu_2)^2\mathrm{p}(\mathbf{x})d\mathbf{x}\right]. \tag{B.12}$$

Adding in a positive and negative μ_1 to the square term of equation B.12

$$\int \mathrm{p}(\mathbf{x})\left[-\frac{(\mathbf{x}-\mu_1)^2}{2\sigma_1^2}+\frac{(\mathbf{x}-\mu_2)^2}{2\sigma_2^2}\right]d\mathbf{x} = -\frac{1}{2}+\frac{1}{2\sigma_2^2}\left[\int(\mathbf{x}-\mu_1+\mu_1-\mu_2)^2\mathrm{p}(\mathbf{x})d\mathbf{x}\right]. \quad \text{(B.13)}$$

Equation B.13 can be re-written as:

$$\begin{aligned}\int \mathrm{p}(\mathbf{x})\left[-\frac{(\mathbf{x}-\mu_1)^2}{2\sigma_1^2}+\frac{(\mathbf{x}-\mu_2)^2}{2\sigma_2^2}\right]d\mathbf{x} = -\frac{1}{2}+\frac{1}{2\sigma_2^2}\Bigg[&\int(\mathbf{x}-\mu_1)^2\mathrm{p}(\mathbf{x})d\mathbf{x}\\ &+(\mu_1-\mu_2)^2\int\mathrm{p}(\mathbf{x})d\mathbf{x}\\ &+2(\mu_1-\mu_2)\int(x-\mu_1)\mathrm{p}(\mathbf{x})d\mathbf{x}\Bigg].\end{aligned} \quad \text{(B.14)}$$

Equation B.14 can be simplified using the following definitions,

$$\begin{aligned}&\textstyle\int(\mathbf{x}-\mu_1)^2\mathrm{p}(\mathbf{x})d\mathbf{x} = \sigma_1^2\\ &\textstyle\int\mathrm{p}(\mathbf{x})d\mathbf{x} = 1\\ &\textstyle\int(\mathbf{x}-\mu_1)\mathrm{p}(\mathbf{x})d\mathbf{x} = 0,\end{aligned}$$

which results in

$$\int \mathrm{p}(\mathbf{x})\left[-\frac{(\mathbf{x}-\mu_1)^2}{2\sigma_1^2}+\frac{(\mathbf{x}-\mu_2)^2}{2\sigma_2^2}\right]d\mathbf{x} = -\frac{1}{2}+\frac{1}{2\sigma_2^2}\left[\sigma_1^2+(\mu_1-\mu_2)^2\right]. \quad \text{(B.15)}$$

Inputting equations B.10 and B.15 into equation B.9 gives

$$KL(\mathrm{p}||\mathrm{q}) = \frac{1}{2}\log\left(\frac{\sigma_2^2}{\sigma_1^2}\right)+-\frac{1}{2}+\frac{1}{2\sigma_2^2}\left[\sigma_1^2+(\mu_1-\mu_2)^2\right]. \quad \text{(B.16)}$$

Equation B.16 can be simplified to the equation used in Section 4.3:

$$KL(\mathrm{p}||\mathrm{q}) = \log\left(\frac{\sigma_2}{\sigma_1}\right)+\frac{\sigma_1^2+(\mu_1-\mu_2)^2}{2\sigma_2^2}-\frac{1}{2}. \quad \text{(B.17)}$$

THIS PAGE INTENTIONALLY LEFT BLANK

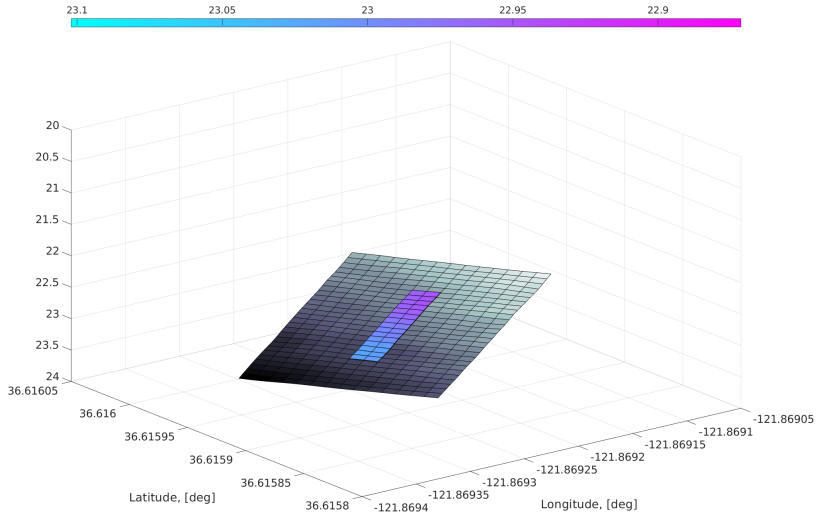
APPENDIX C:
Boltzmann Entropy

This appendix shows all Boltzmann entropy values shown in Table C.1.

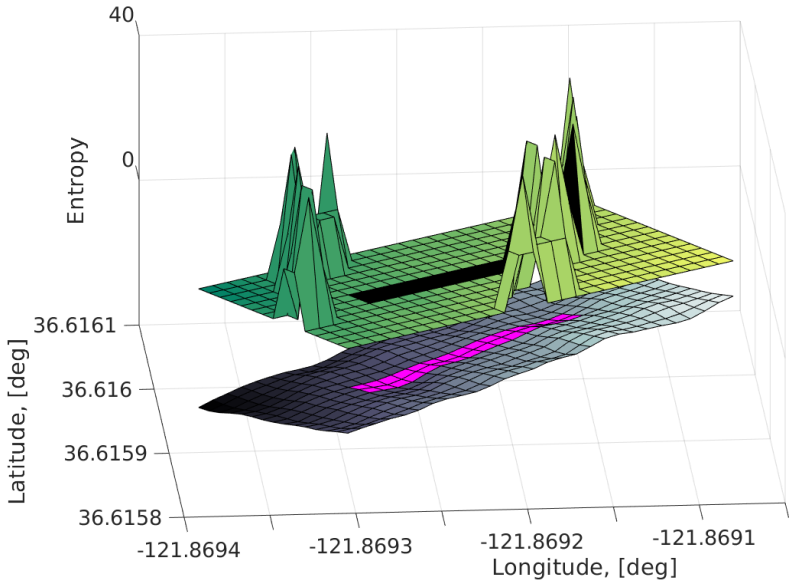
Table C.1. Boltzmann Entropy and Relative Localization Probability.

Boltzmann Entropy $\xi(\mathbf{m}_n)$	**Localization Probability** $p(X^*\|\xi(\mathbf{m}_n))$
1	13.7 %
50	32.9 %
180	90.2 %
300	99.8 %
2160	100 %

C.1 Boltzmann Entropy Maximum of 1



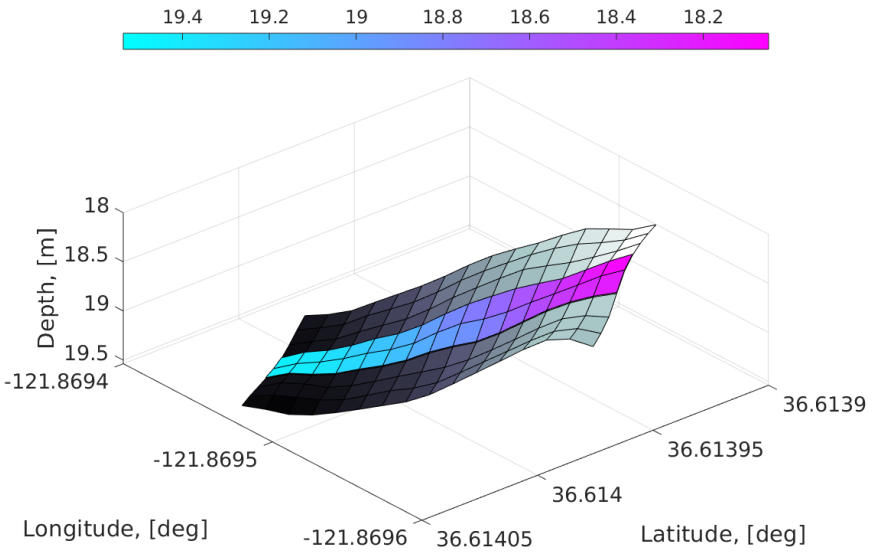
(a) Depth Profile.



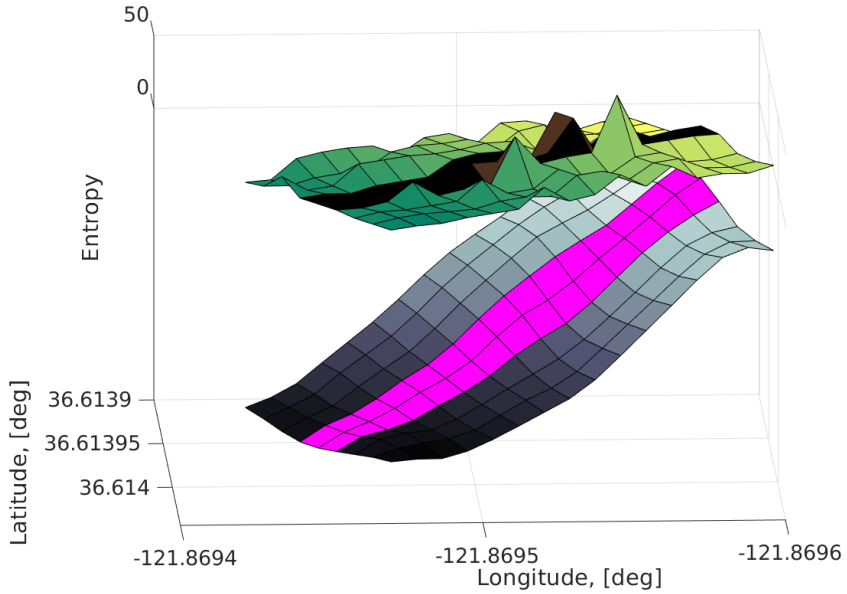
(b) Entropy Profile.

Figure C.1. Boltzmann Entropy Depth Profile for $\max(\xi(\mathbf{m}))= 1$.

C.2 Boltzmann Entropy Maximum of 50



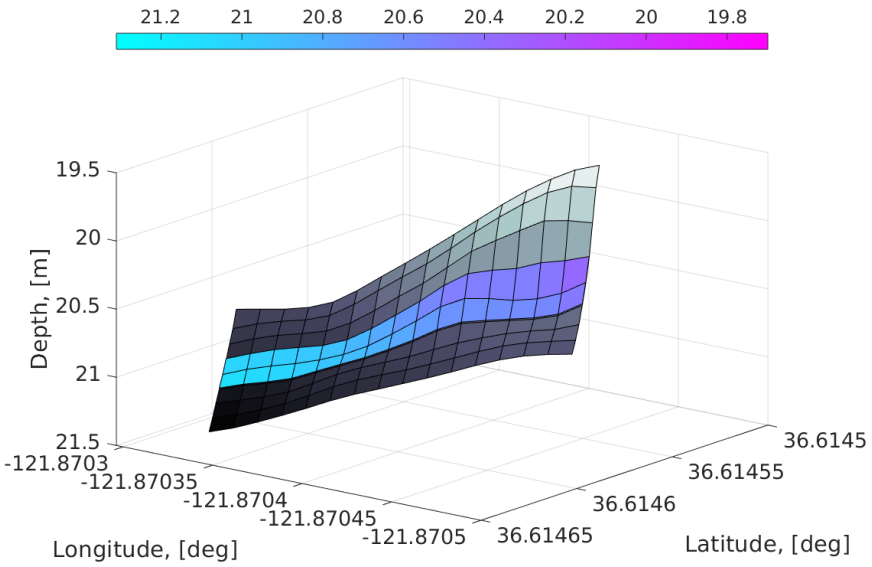
(a) Depth Profile.



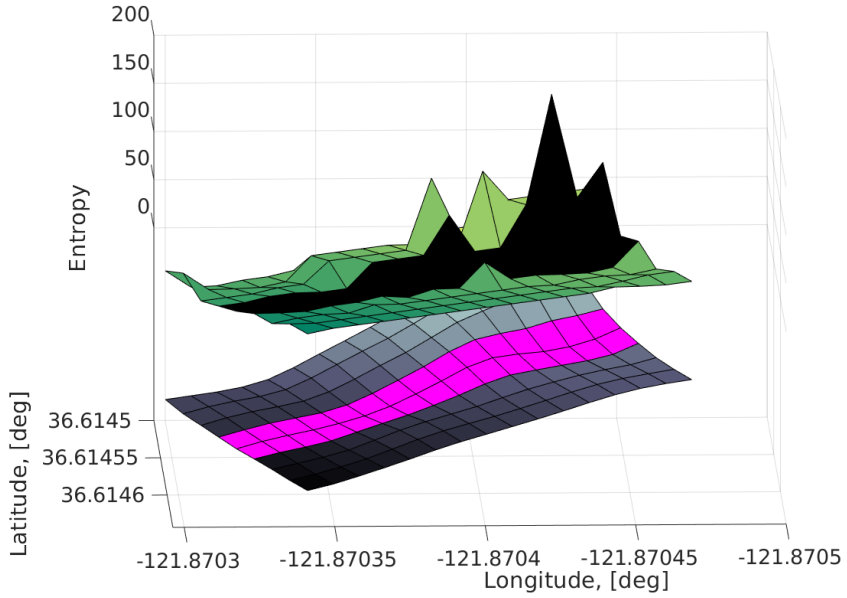
(b) Entropy Profile.

Figure C.2. Boltzmann Entropy Depth Profile for max($\xi(\mathbf{m})$)= 50.

C.3 Boltzmann Entropy Maximum of 180



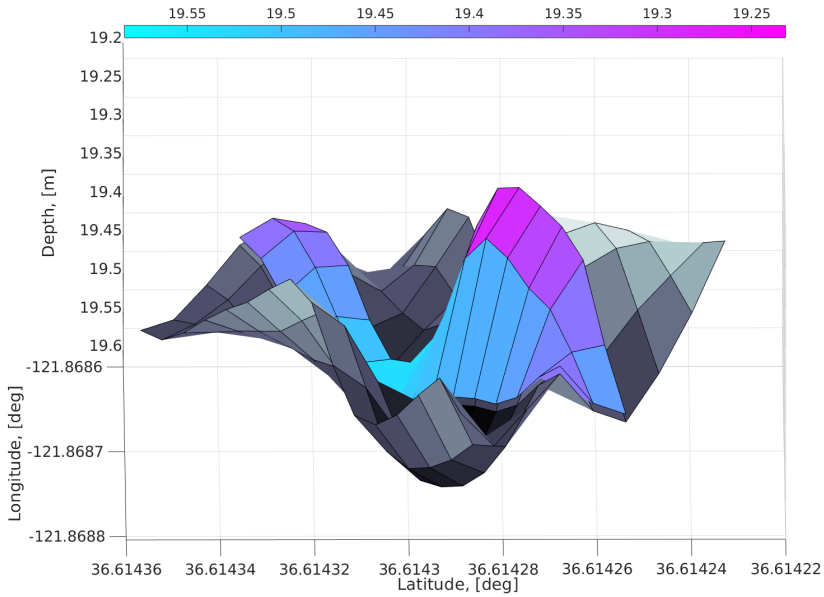
(a) Depth Profile.



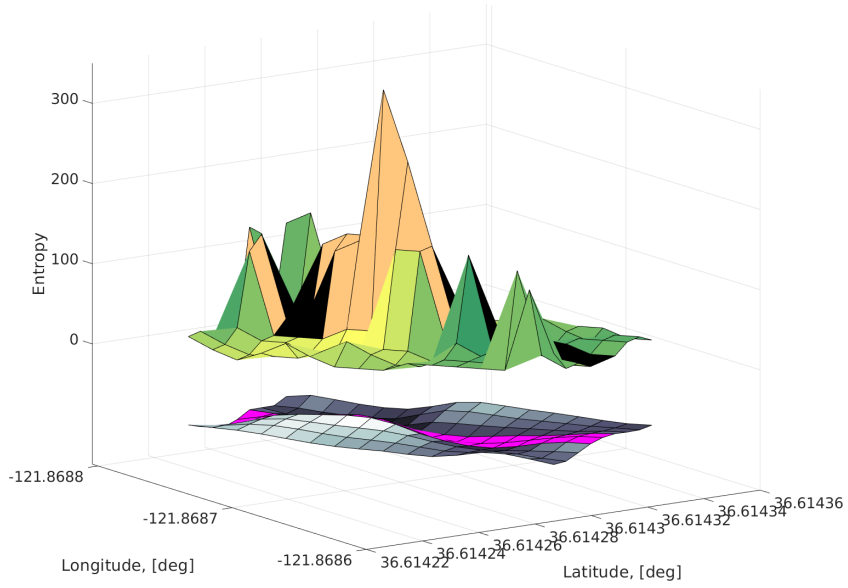
(b) Entropy Profile.

Figure C.3. Boltzmann Entropy Depth Profile for max($\xi(\mathbf{m})$)= 180.

C.4 Boltzmann Entropy Maximum of 300



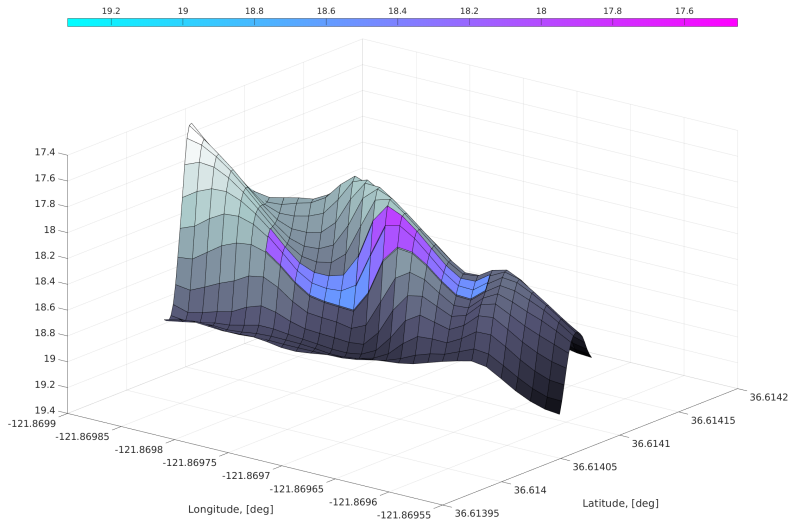
(a) Depth Profile.



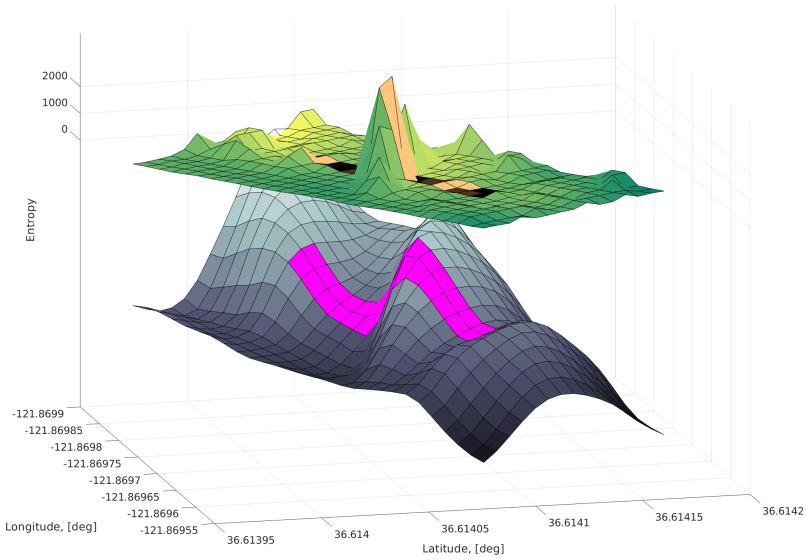
(b) Entropy Profile.

Figure C.4. Boltzmann Entropy Depth Profile for max($\xi(\mathbf{m})$)= 300.

C.5 Boltzmann Entropy Maximum of 2160



(a) Depth Profile.



(b) Entropy Profile.

Figure C.5. Boltzmann Entropy Depth Profile for max($\xi(\mathbf{m})$)= 2160.

APPENDIX D:
Derivation Of Optimal Estimator

This appendix shows the derivation of the optimal estimator $\hat{r}$ used in the reward function shown in Chapter 6. This dual optimization is between a spatial estimator, *Exploration*, and a temporal model, *Exploitation*, to best choose the trajectory of the AUV.

D.1 Variables

The following variables are used:

- r is the reward, the true change of uncertainty
- k_1 is the *Exploration* gain
- ν_1 is the uncertainty around of the *Exploration* score and is modeled as zero mean white noise
- z_1 is *Exploration* score with uncertainty ν_1, $z_1 = r + \nu_1$
- k_2 is the *Exploitation* gain
- ν_2 is the uncertainty of the *Exploitation* score and is modeled as zero mean white noise
- z_2 is the *Exploitation* score with error ν_2, $z_2 = r + \nu_2$
- $\hat{r}$ is the estimated reward
- $\tilde{r}$ is estimated reward, $\tilde{r} = \hat{r} - r$

D.2 Solution

Forming a possible solution,

$$\hat{r} = k_1 z_1 + k_2 z_2, \tag{D.1}$$

the mean value of the estimated reward $\tilde{r}$ can be calculated by:

$$\mathbb{E}[\tilde{r}] = \mathbb{E}[\hat{r} - r] = 0. \tag{D.2}$$

Inputting equation D.1 into D.2,

$$0 = \mathbb{E}[k_1 z_1 + k_2 z_2] - \mathbb{E}[r] \tag{D.3}$$

Since r is the true reward value, $\mathbb{E}[r] = r$, so equation D.3 becomes

$$\begin{aligned} 0 &= \mathbb{E}[k_1 z_1 + k_2 z_2] - r \\ 0 &= \mathbb{E}[k_1(r + \nu_1) + k_2(r + \nu_2)] - r \\ 0 &= \mathbb{E}[k_1(r + \nu_1)] + \mathbb{E}[k_2(r + \nu_2)] - r \end{aligned} . \tag{D.4}$$

Evaluating each term in D.4 individually,

$$\begin{aligned} \mathbb{E}[k_1(r + \nu_1)] &= r\,\mathbb{E}[k_1] + \mathbb{E}[k_1 \nu_1] \\ \mathbb{E}[k_1(r + \nu_1)] &= r k_1 \end{aligned} . \tag{D.5}$$

$$\begin{aligned} \mathbb{E}[k_2(r + \nu_2)] &= r\,\mathbb{E}[k_2] + \mathbb{E}[k_2 \nu_2] \\ \mathbb{E}[k_2(r + \nu_2)] &= r k_2 \end{aligned} . \tag{D.6}$$

In equation D.5 and D.6, $\mathbb{E}[k_1 \nu_1] = \mathbb{E}[k_2 \nu_2] = 0$ since ν_1 and ν_2 are zero mean Gaussian noise. Inputting the results from equations D.5 and D.6 back into equation D.4,

$$0 = k_1 r + k_2 r - r. \tag{D.7}$$

Solving for k_2 as a function of k_1,

$$\begin{aligned} k_2 &= \frac{r - k_1 r}{r} \\ k_2 &= 1 - k_1. \end{aligned} \tag{D.8}$$

The estimated reward, $\hat{r}$, can be written using equation D.1. Inputting D.8 into equation

D.1,

$$\begin{aligned} \hat{r} &= k_1 z_1 + k_2 z_2 \\ \hat{r} &= k_1 z_1 + (1 - k_1) z_2. \end{aligned} \tag{D.9}$$

Computing the estimated reward error, $\tilde{r}$,

$$\begin{aligned} \tilde{r} &= k_1(r + \nu_1) + (1 - k_1)(r + \nu_2) \\ \tilde{r} &= k_1 \nu_1 + \nu_2 - k_1 \nu_2. \end{aligned} \tag{D.10}$$

Computing the variance of the system, $\mathbb{E}[\tilde{r}^2]$,

$$\mathbb{E}[\tilde{r}^2] = \mathbb{E}[k_1^2 \nu_1^2 + 2\nu_2(1 - k_1)k_1 \nu_1 + \nu_2^2(1 - k_1)^2]. \tag{D.11}$$

Equation D.11 can be simplified since $\mathbb{E}[\nu_1] = \mathbb{E}[\nu_2] = \mathbb{E}[\nu_1 \nu_2] = 0$ since ν_1 and ν_2 are uncorrelated. Additionally, the variance of the systems, $\mathbb{E}[\nu_1^2] = \sigma_1^2$ and $\mathbb{E}[\nu_2^2] = \sigma_2^2$, leading to

$$\mathbb{E}[\tilde{r}^2] = \mathbb{E}\, k_1^2 \sigma_1^2 + (1 - k_1)^2 \sigma_2^2. \tag{D.12}$$

Finding the minimum variance of the system with respect to k_1,

$$\frac{\partial\, \mathbb{E}[\tilde{r}^2]}{\partial k_1} = 0, \tag{D.13}$$

and applying equation D.12:

$$0 = 2k_1 \sigma^2 - 2\sigma_2^2 + 2k_1 \sigma_2^2. \tag{D.14}$$

Simplifying equation D.14 and solving for k_1:

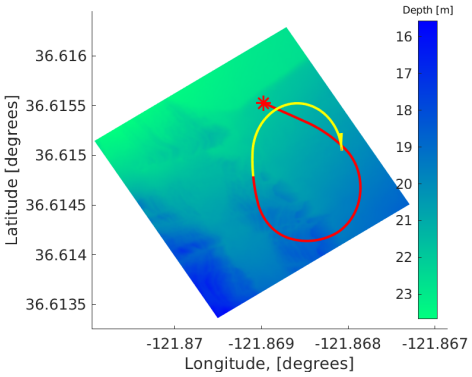
$$k_1 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}. \tag{D.15}$$

Using equations D.8 andD.15 k_2 can be solved for,

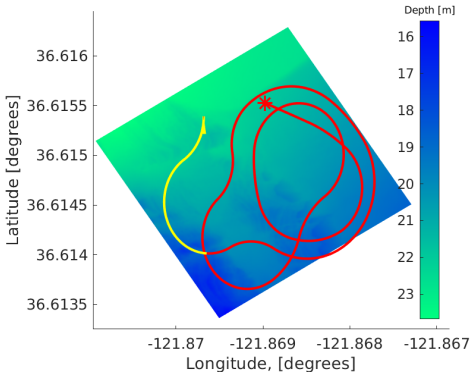
$$k_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}. \tag{D.16}$$

APPENDIX E:
Full Trajectory

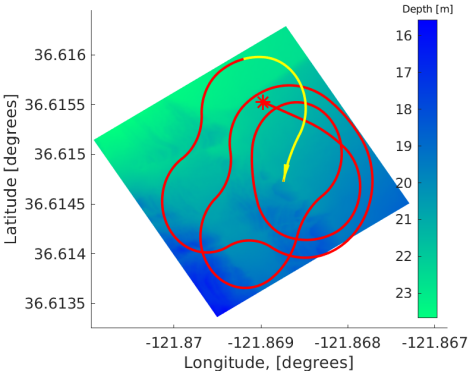
E.1 Result Trajectory



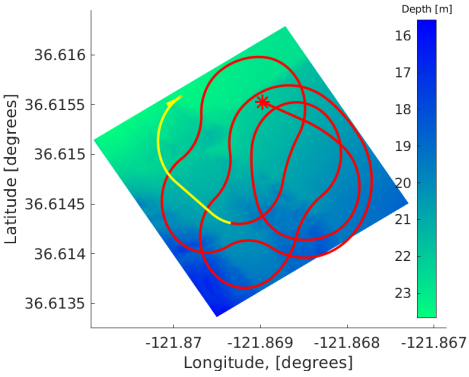
(a) 500m (Cov 21.0, posit. error 0.23m).



(b) 1500m (Cov 55.3, posit. error 0.36m).

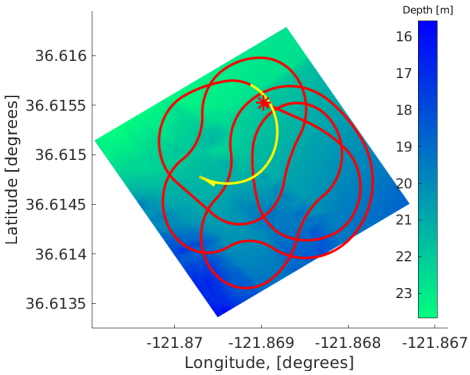


(c) 1750m (Cov 63.6, posit. error 0.17m).

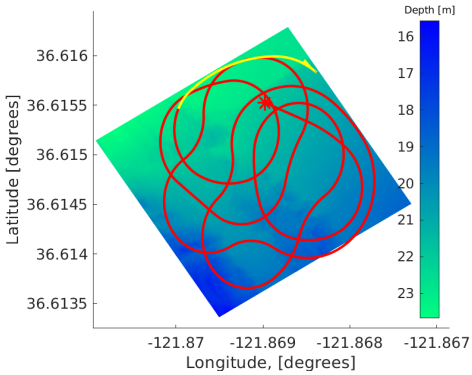


(d) 2000m (Cov 71.2, posit. error 0.63m).

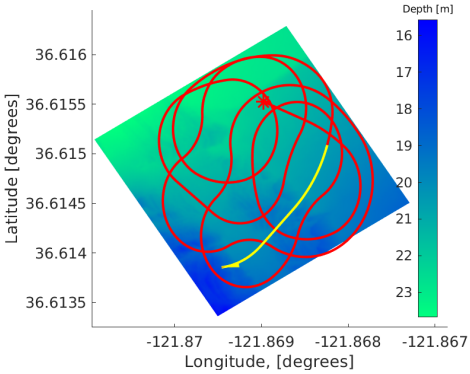
Figure E.1. POMCP Result Trajectories for 500m to 2000m.



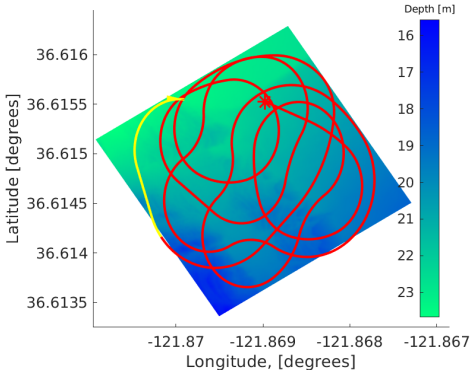
(a) 2250m (Cov 75.5, posit. error 0.11m).



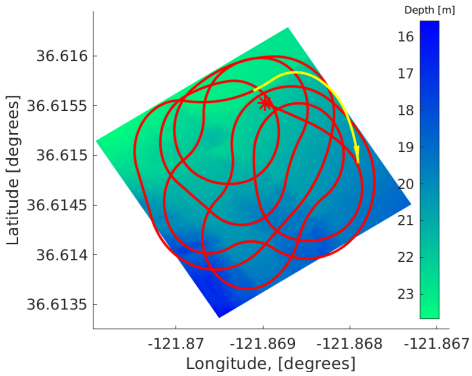
(b) 2500m (Cov 77.6, posit. error 0.18m).



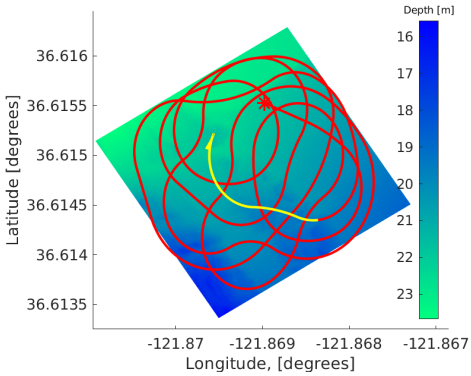
(c) 2750m (Cov 81.4, posit. error 0.71m).



(d) 3000m (Cov 86.7, posit. error 0.94m).

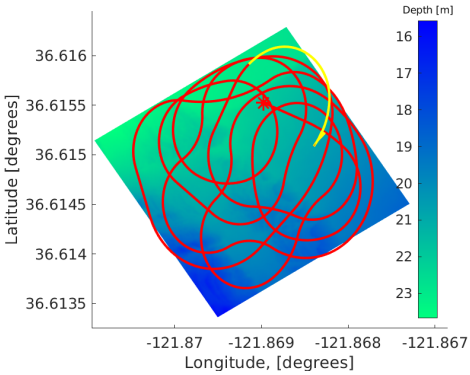


(e) 3250 (Cov 87.5, posit. error 0.33m).

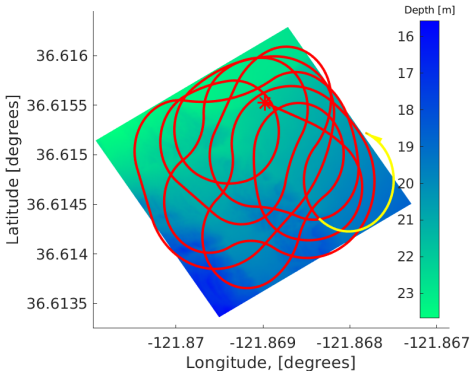


(f) 3500m (Cov 90.5, posit. error 1.1m).

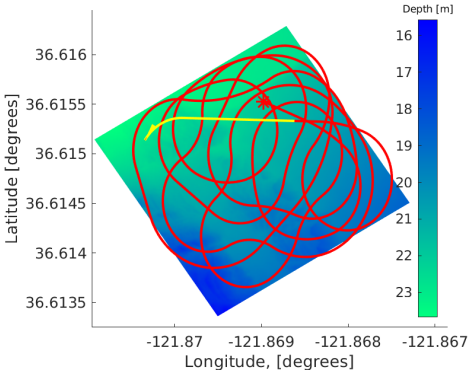
Figure E.2. POMCP Result Trajectories for 2250m to 3500m.



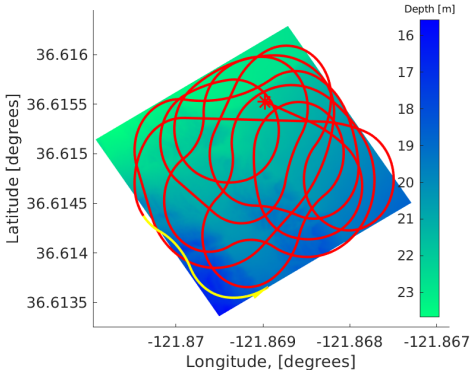
(a) 3750m (Cov 91.7, posit. error 0.1m).



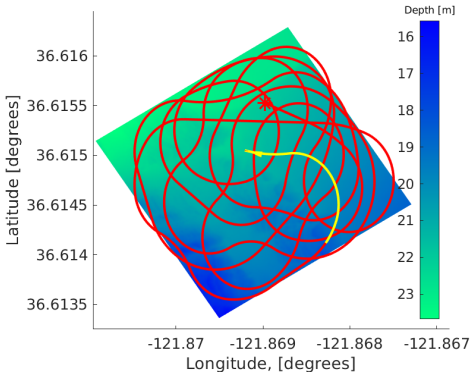
(b) 4000m (Cov 93.7, posit. error 0.22m).



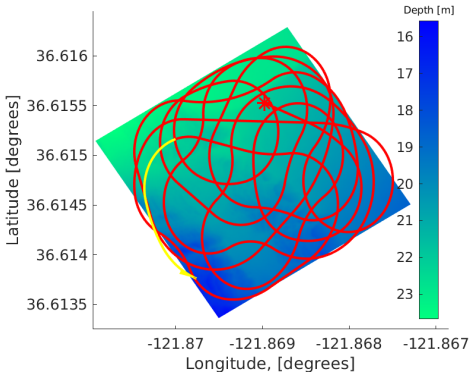
(c) 4250m (Cov 94.1, posit. error 0.25m).



(d) 4500m (Cov 95.7, posit. error 0.72m).

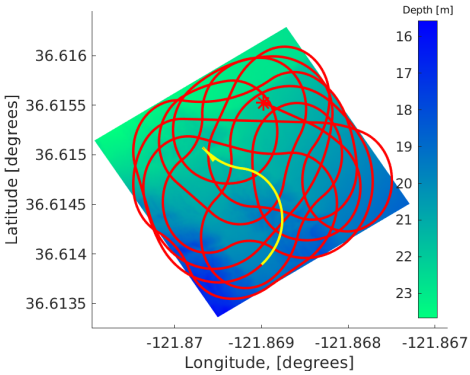


(e) 4750 (Cov 95.8, posit. error 0.13m).

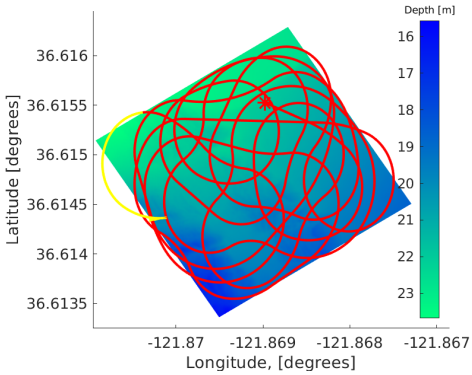


(f) 5000m (Cov 96.1, posit. error 0.24m).

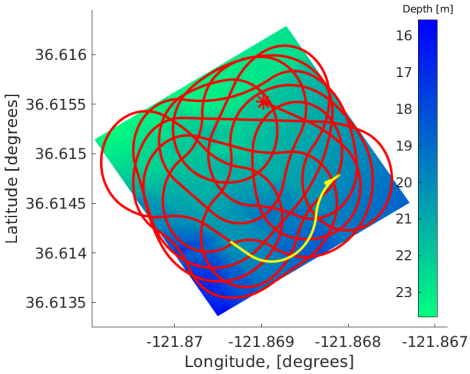
Figure E.3. POMCP Result Trajectories for 3750m to 5000m.



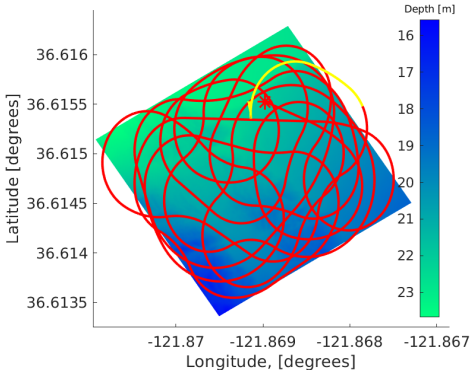
(a) 5250m (Cov 96.6, posit. error 0.17m).



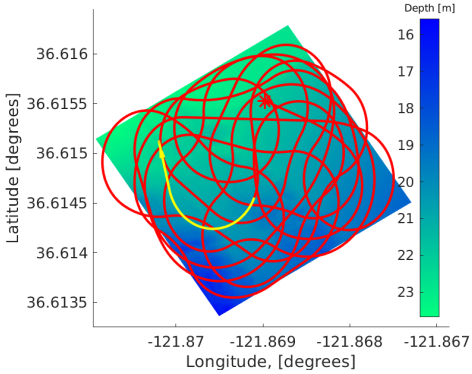
(b) 5500m (Cov 97.8, posit. error 1.1m).



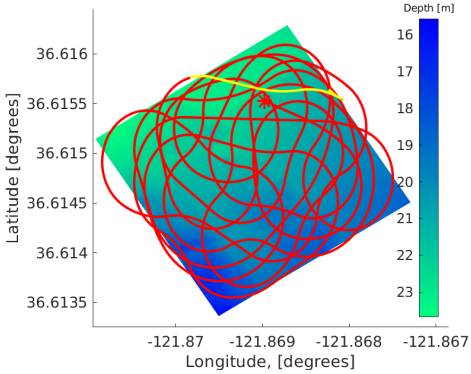
(c) 5750m (Cov 98.6, posit. error 0.28m).



(d) 6000m (Cov 98.7, posit. error 0.13m).

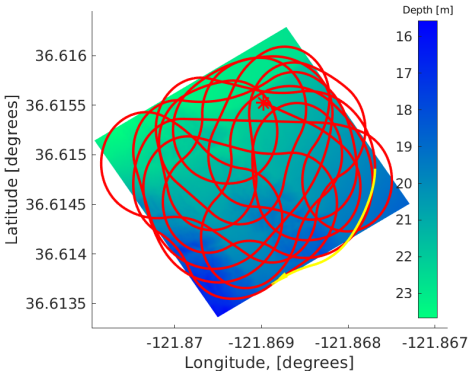


(e) 6250m (Cov 98.8, posit. error 0.16m).

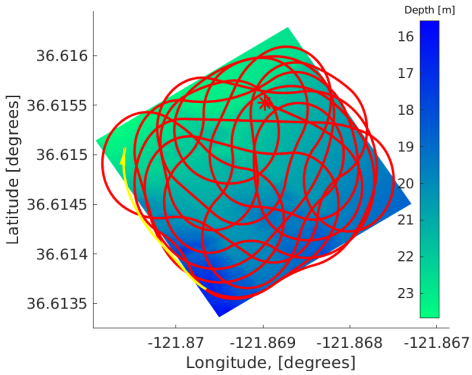


(f) 6500m (Cov 98.8, posit. error 0.09m).

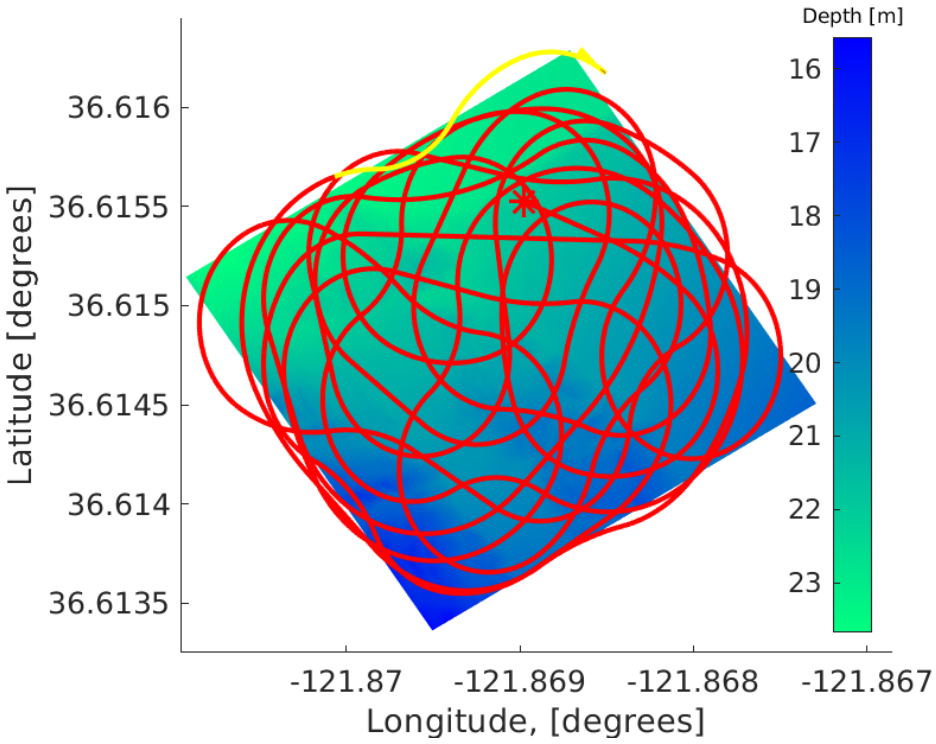
Figure E.4. POMCP Result Trajectories for 5250m to 6500m.



(a) 6750m (Cov 98.8, posit. error 0.11m).

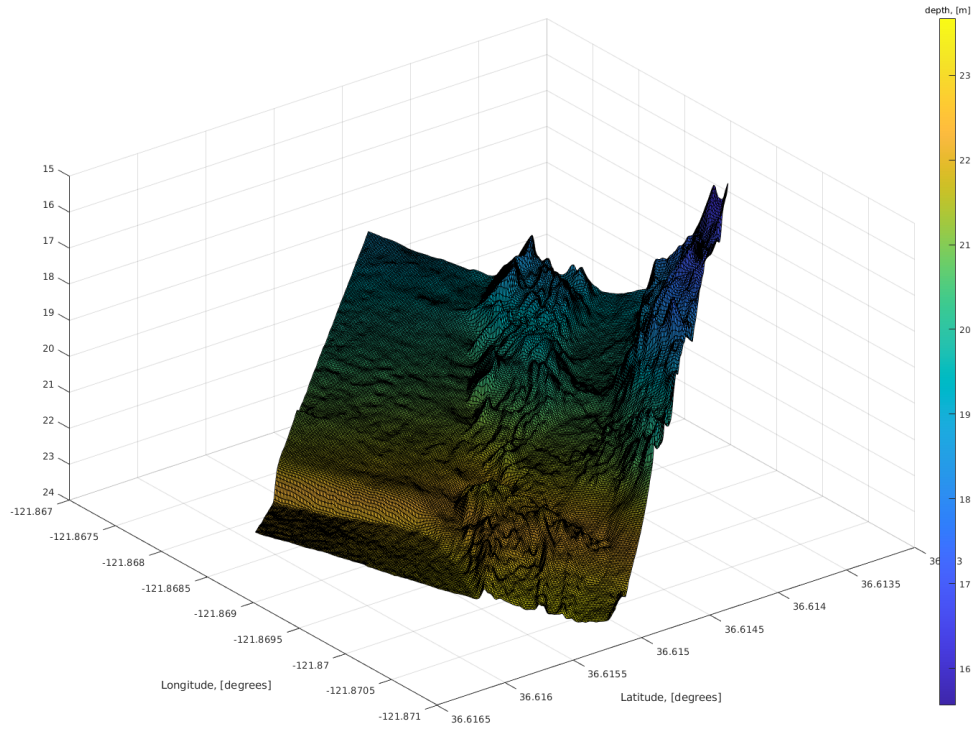


(b) 7000m (Cov 99.4, posit. error 0.29m).

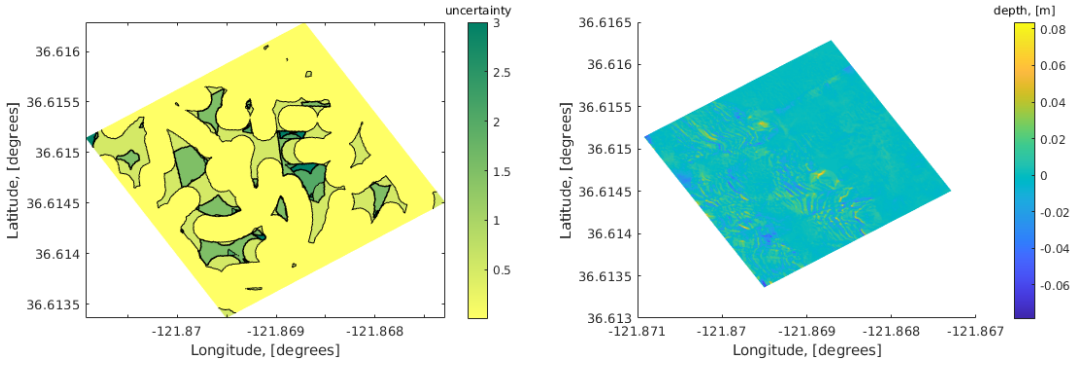


(c) 7250m (Cov 100, posit. error 0.34m).

Figure E.5. POMCP Result Trajectories for 6750m to 7250m.



(a)



(b) (c)

Figure E.6. (a) AUV Generated Map. (b) AUV GPM at Completion of Trajectory. (c) AUV Map Error Comparison to CSUMB Map [78].

List of References

[1] D. Glass, "What happens if gps fails?" *The Atlantic*, Jun 2016. [Online]. Available: https://www.theatlantic.com/technology/archive/2016/06/what-happens-if-gps-fails/486824/

[2] S. Carreño, P. Wilson, P. Ridao, and Y. Petillot, "A survey on terrain based navigation for AUVs," in *Oceans 2010 MTS/IEEE Seattle*, Sep. 2010, pp. 1–7.

[3] J. P. Golden, "Terrain Contour Matching (TERCOM): A cruise missile guidance aid," in *Image Processing for Missile Guidance*, T. F. Wiener, Ed., International Society for Optics and Photonics. SPIE, 1980, vol. 0238, pp. 10 – 18. Available: https://doi.org/10.1117/12.959127

[4] M. Lauri and R. Ritala, "Planning for robotic exploration based on forward simulation," *Robot. Auton. Syst.*, vol. 83, no. C, pp. 15–31, Sep. 2016.

[5] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017. [Online]. Available: https://doi.org/10.1038/nature24270

[6] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 2164–2172. [Online]. Available: http://papers.nips.cc/paper/4031-monte-carlo-planning-in-large-pomdps.pdf

[7] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, Mar. 2012.

[8] L. G. Baehr, "Swimming with sharks: an underwater robot learns how to track great whites," *Oceanus*, vol. 50, no. 2, pp. 42+, 2013.

[9] K. L. Dodge, A. L. Kukulya, E. Burke, and M. F. Baumgartner, "Turtlecam: A "smart" autonomous underwater vehicle for investigating behaviors and habitats of sea turtles," *Frontiers in Marine Science*, March 2015. [Online]. Available: https://doi.org/10.3389/fmars.2018.00090

[10] J. C. Hodgson, R. Mott, S. M. Baylis, T. T. Pham, S. Wotherspoon, A. D. Kilpatrick, R. Raja Segaran, I. Reid, A. Terauds, and L. P. Koh, "Drones count wildlife more accurately and precisely than humans," *Methods in Ecology and Evolution*, vol. 9, no. 5, pp. 1160–1167, 2018. [Online]. Available: https://doi.org/10.1111/2041-210X.12974

[11] L. E. Llewellyn and S. J. Bainbridge, "Getting up close and personal: the need to immerse autonomous vehicles in coral reefs," in *Oceans 2015 - MTS/IEEE Washington*, Oct 2015, pp. 1–9.

[12] M. A. Hassan, M. Yang, L. Fu, A. Rasheed, B. Zheng, X. Xia, Y. Xiao, and Z. He, "Accuracy assessment of plant height using an unmanned aerial vehicle for quantitative genomic analysis in bread wheat," *Plant Methods*, vol. 15, no. 1, p. 37, 2019. [Online]. Available: https://doi.org/10.1186/s13007-019-0419-7

[13] Kongsberg, "Autonomous underwater vehicles–REMUS 100," 2019. Available: https://www.kongsberg.com/maritime/products/marine-robotics/autonomous-underwater-vehicles/AUV-remus-100/

[14] V. Howard, J. Mefford, L. Arnold, B. Bingham, and R. Camilli, "The unmanned port security vessel: An autonomous platform for monitoring ports and harbors," in *Oceans'11 MTS/IEEE Kona*, Sep. 2011, pp. 1–8.

[15] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robotics Automation Magazine*, vol. 19, no. 3, pp. 46–56, Sep. 2012.

[16] J. T. Juriga, "Terrain Aided Navigation for REMUS Autonomous Underwater Vehicle," Master's thesis, Department of Mechanical and Aerospace Engineering, Naval Postgraduate School, Monterey, CA, USA, 2014.

[17] G. Kearfott Corporation and N. Division, *Kearfott SeaNav*, Kearfott, 2016. [Online]. Available: http://www.kearfott.com/wp-content/uploads/2016/08/Kearfott_SeaNav.pdf

[18] T. BlueView, *Company Product Guide: High-Resolution Underwater Acoustic Imaging, Measurement, and Automation Systems*, Teledyne Marine Company, 2015. [Online]. Available: https://www.m-b-t.com/fileadmin/redakteur/Hydrographie/Multibeam/Blueview/tbv-product-guide-2015-lowres.pdf

[19] T. Westbrook, "The global positioning system and military jamming: geographies of electronic warfare," *Journal of Strategic Security*, vol. 12, no. 2, pp. 1–16, 2019. [Online]. Available: https://www.jstor.org/stable/26696257

[20] M. L. Psiaki, T. E. Humphreys, and B. Stauffer, "Attackers can spoof navigation signals without our knowledge. here's how to fight back gps lies," *IEEE Spectrum*, vol. 53, no. 8, pp. 26–53, August 2016.

[21] H. Rice, S. Kelmenson, and L. Mendelsohn, "Geophysical navigation technologies and applications," in *PLANS 2004. Position Location and Navigation Symposium (IEEE Cat. No.04CH37556)*, April 2004, pp. 618–624.

[22] K. Vickery, "Acoustic positioning systems. a practical overview of current systems," in *Proceedings of the 1998 Workshop on Autonomous Underwater Vehicles (Cat. No.98CH36290)*, Aug 1998, pp. 5–17.

[23] K. Osborn, "Darpa discovers 'GPS-like' undersea drone connectivity," *Defense Systems*, Feb. 2017. Available: https://defensesystems.com/articles/2017/02/14/darpauuv.aspx

[24] M. Hammond, S. Houts, S. Dektor, and S. Krukowski, "Terrain relative navigation," *Stanford Aerospace Robotics Lab*, Jan 2017. [Online]. Available: https://web.stanford.edu/group/arl/projects/terrain-relative-navigation

[25] I. Nygren and M. Jansson, "Terrain navigation using the correlator method," in *PLANS 2004. Position Location and Navigation Symposium (IEEE Cat. No.04CH37556)*, April 2004, pp. 649–657.

[26] I. Nygren, "Terrain navigation for underwater vehicles," Ph.D. dissertation, Dept. of Signals, Sensors and Systems, Electrical Engineering, Royal Institute of Technology (KTH), Stockholm, Sweden, 2005.

[27] P. Kimball and S. M. Rock, "Sonar-Based Iceberg-Relative AUV Navigation," in *Proceedings Of AUV2008*. Woods Hole, MA: IEEE, October 2008. [Online]. Available: http://www.stanford.edu/group/arl/cgi-bin/drupal/sites/default/files/public/publications/KimballR2008.pdf

[28] P. Kimball and S. M. Rock, "Sonar-Based Iceberg-Relative AUV Localization," in *UUST: Unmanned Untethered Submersible Technology*, R. Blidberg, Ed. Durham, NH: Autonomous Undersea Systems Institute, August 2009. [Online]. Available: http://www.stanford.edu/group/arl/cgi-bin/drupal/sites/default/files/public/publications/KimballR2009.pdf

[29] J. Gao, D. Peng, T. Zhou, T. Wang, and C. Xu, "Terrain matching localization for underwater vehicle based on gradient fitting," *Journal of Sensors*, vol. 2018, Jan 2018. [Online]. Available: http://search.proquest.com/docview/2166681671/

[30] S. Dektor and S. Rock, "Improving robustness of terrain-relative navigation for AUVs in regions with flat terrain," in *2012 IEEE/OES Autonomous Underwater Vehicles (AUV)*. IEEE, Sep. 2012, pp. 1–7.

[31] S. Dektor and S. Rock, "Robust adaptive terrain-relative navigation," in *2014 Oceans - St. John's*. IEEE, Sep. 2014, pp. 1–10.

[32] J. Melo and A. Matos, "Survey on advances on terrain based navigation for autonomous underwater vehicles," *Ocean Engineering*, vol. 139, pp. 250–264, July 2017.

[33] D. Meduna, S. M. Rock, and R. McEwen, "Low-Cost terrain relative navigation for long-range AUVs," in *Proceedings of the Oceans 2008 MTS/IEEE Quebec Conference*, Quebec City, Canada, Sept. 2008. [Online]. Available: http://www.stanford.edu/group/arl/cgi-bin/drupal/sites/default/files/public/publications/MedunaRM2008.pdf

[34] M. Sualeh and G.-W. Kim, "Simultaneous localization and mapping in the epoch of semantics: A survey," *International Journal of Control, Automation and Systems*, vol. 17, no. 3, pp. 729,742, 2019-03.

[35] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE Robotics Automation Magazine*, vol. 13, no. 2, pp. 99–110, June 2006.

[36] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, p. 1309–1332, Dec 2016. [Online]. Available: http://dx.doi.org/10.1109/TRO.2016.2624754

[37] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," *IEEE Robotics Automation Magazine*, vol. 13, no. 3, pp. 108–117, Sep. 2006.

[38] M. L. Rodríguez-Arévalo, J. Neira, and J. A. Castellanos, "On the importance of uncertainty representation in active SLAM," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 829–834, June 2018.

[39] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[40] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[41] M. Prágr, P. Cizek, J. Bayer, and J. Faigl, "Online incremental learning of the terrain traversal cost in autonomous exploration," in *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*, June 2019. [Online]. Available: https://doi.org/10.15607/RSS.2019.XV.040

[42] L. Nardi and C. Stachniss, "Actively improving robot navigation on different terrains using gaussian process mixture models," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 4104–4110.

[43] G. Rui and M. Chitre, "Path planning for bathymetry-aided underwater navigation," in *2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*. IEEE, 2018-11, pp. 1–6.

[44] A. Z. Zambom, B. Seguin, and F. Zhao, "Robot path planning in a dynamic environment with stochastic measurements.(report)," *Journal of Global Optimization*, vol. 73, no. 2, pp. 389–410, Sept. 2018.

[45] M. Ornik and U. Topcu, "Learning and planning for time-varying mdps using maximum likelihood estimation," Nov. 2019.

[46] J. Zhang, C. Hu, R. G. Chadha, and S. Singh, "Maximum likelihood path planning for fast aerial maneuvers and collision avoidance," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Nov 2019, pp. 2805–2812.

[47] M. Fu, A. Kuntz, O. Salzman, and R. Alterovitz, "Toward asymptotically-optimal inspection planning via efficient near-optimal graph search," *Robotics: Science and Systems XV*, Jun 2019. [Online]. Available: http://dx.doi.org/10.15607/RSS.2019.XV.057

[48] E. Ayvali, H. Salman, and H. Choset, "Ergodic coverage in constrained environments using stochastic trajectory optimization," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 5204–5210.

[49] J. Santos, T. Krajník, and T. Duckett, "Spatio-temporal exploration strategies for long-term autonomy of mobile robots," *Robotics and Autonomous Systems*, vol. 88, Nov. 2016.

[50] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, July 1948.

[51] E. C. Cherry, "A history of the theory of information," *Proceedings of the IEE - Part III: Radio and Communication Engineering*, vol. 98, no. 55, pp. 383–393, Sep. 1951.

[52] S. Höst, *Information and communication theory* (IEEE Series on Digital Mobile Communication). Hoboken, New Jersey: IEEE Press, 2019.

[53] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," in *Robotics Science and Systems*, 2005, pp. 65–72.

[54] L. Carlone, J. Du, M. K. Ng, B. Bona, and M. Indri, "An application of kullback-leibler divergence to active slam and exploration with particle filters," *IEEE Intelligent Robots and Systems*, vol. 18, no. 22, pp. 287–293, October 2010.

[55] P. Cai, Y. Luo, A. Saxena, D. Hsu, and W. S. Lee, "Lets-drive: Driving in a crowd by learning from tree search," *Robotics Science and Systems*, 2019.

[56] Z. Liu, M. Zhou, W. Cao, Q. Qu, H. W. F. Yeung, and V. Y. Y. Chung, "Towards understanding chinese checkers with heuristics, monte carlo tree search, and deep reinforcement learning," Mar. 2019.

[57] A. Castellini, G. Chalkiadakis, and A. Farinelli, "Influence of state-variable constraints on partially observable monte carlo planning," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, July 2019., pp. 5540–5546. [Online]. Available: https://doi.org/10.24963/ijcai.2019/769

[58] A. Bai, F. Wu, and X. Chen, "Posterior sampling for monte carlo planning under uncertainty," *Applied Intelligence*, vol. 48, no. 12, pp. 4998,5018, Dec. 2018.

[59] *Summary of the 2018 Department of Defense Artificial Intelligence Strategy*, Department of Defense, Washington D.C., USA, 2019.

[60] S. A. Cushman, "Calculating the configurational entropy of a landscape mosaic," *Landscape ecology*, vol. 31, no. 3, pp. 481–489, 2016.

[61] P. Gao, H. Zhang, and Z. Li, "A hierarchy-based solution to calculate the configurational entropy of landscape gradients," *Landscape ecology*, vol. 32, no. 6, pp. 1133–1146, 2017.

[62] S. Thrun, "Particle filters in robotics," in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (UAI'02). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002, p. 511–518.

[63] X. Wang, T. Li, S. Sun, and J. Corchado Rodríguez, "A survey of recent advances in particle filters and remaining challenges for multitarget tracking," *Sensors*, vol. 17, p. 2707, Nov. 2017.

[64] A. Akca and M. Önder Efe, "Multiple model Kalman and particle filters and applications: A survey," *IFAC-PapersOnLine*, vol. 52, no. 3, pp. 73 – 78, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896319300977

[65] T. Tung and T. Matsuyama, "Visual tracking using multimodal particle filter," in *Computer Vision*, 2018, pp. 1072–1090.

[66] A. Murangira, C. Musso, K. Dahia, and J.-M. Allard, "Robust regularized particle filter for terrain navigation," *Fusion 2011 - 14th International Conference on Information Fusion*, Jan. 2011.

[67] S. Thrun, W. Burgard, D. Fox, and R. Arkin, *Probabilistic Robotics* (Intelligent Robotics and Autonomous Agents series). MIT Press, 2005.

[68] D. Peng, T. Zhou, J. Folkesson, and C. Xu, "Robust particle filter based on huber function for underwater terrain-aided navigation," *IET Radar, Sonar Navigation*, vol. 13, no. 11, pp. 1867,1875, Nov. 2019.

[69] K. B. Anonsen and O. Hallingstad, "Terrain aided underwater navigation using point mass and particle filters," in *2006 IEEE/ION Position, Location, And Navigation Symposium*, April 2006, pp. 1027–1035.

[70] R. Karlsson and F. Gustafsson, "Bayesian surface and underwater navigation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4204,4213, Nov. 2006.

[71] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, *Particle Filters for Mobile Robot Localization*. New York, NY: Springer New York, 2001, pp. 401–428.

[72] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *The Oxford Handbook of Nonlinear Filtering*, Oxford University. The Oxford Press, 2009, pp. 656–704.

[73] M. A. Nicely and B. E. Wells, "Improved parallel resampling methods for particle filtering," *IEEE Access*, vol. 7, pp. 47 593–47 604, 2019.

[74] T. Li, M. Bolic, and P. M. Djuric, "Resampling methods for particle filtering: Classification, implementation, and strategies," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 70–86, May 2015.

[75] F. Daum and J. Huang, "Nonlinear filters with particle flow induced by log-homotopy," in *Signal Processing, Sensor Fusion, and Target Recognition XVIII*, I. Kadar, Ed., International Society for Optics and Photonics. SPIE, 2009, vol. 7336, pp. 76 – 87. [Online]. Available: https://doi.org/10.1117/12.814241

[76] A. Murangira, C. Musso, and K. Dahia, "A mixture regularized rao-blackwellized particle filter for terrain positioning," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1967–1985, August 2016.

[77] H. Breit and G. Rigoll, "A flexible multimodal object tracking system," Oct. 2003, vol. 3, pp. 111 – 133.

[78] *South/Central Monterey Bay 2009-10*, Sea Floor Mapping Lab, California State University, Monterey Bay. [Online]. Available: http://seafloor.otterlabs.org/SFMLwebDATA_mb.htm

[79] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[80] A. G. Journel and C. J. Huijbregts, *Mining geostatistics / [by] A. G. Journel and Ch. J. Huijbregts*. Academic Press London ; New York, 1978.

[81] M. L. Stein, *Interpolation of spatial data* (Springer Series in Statistics). New York: Springer-Verlag, 1999.

[82] G. Bohling, "Introduction to geostatistics and variogram analysis," *CPE 940 Kansas Geological Survey*, October 2005.

[83] M. D. Ecker and A. E. Gelfand, "Bayesian variogram modeling for an isotropic spatial process," *Journal of Agricultural, Biological, and Environmental Statistics*, vol. 2, no. 4, pp. 347–369, 1997. [Online]. Available: http://www.jstor.org/stable/1400508

[84] S. Castruccio, L. Bonaventura, and L. Sangalli, "A bayesian approach to spatial prediction with flexible variogram models," *Journal of Agricultural, Biological, and Environmental Statistics*, vol. 17, June 2012.

[85] K. P. Murphy, *Machine learning : a probabilistic perspective*. Cambridge, MA: MIT Press, 2013.

[86] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.

[87] T. P. Hill and J. Miller, "How to combine independent data sets for the same quantity," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 21, no. 3, p. 033102, Sep 2011. Available: http://dx.doi.org/10.1063/1.3593373

[88] Deep Sea Ecology: Sea Floor. (2019). World Wildlife Foundation. [Online]. Available: https://wwf.panda.org/our_work/oceans/deep_sea/sea_floor/

[89] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Mathematics of Operations Research*, vol. 12, no. 3, pp. 441–450, 1987.

[90] Y. You, Z. Zhang, C. Hsieh, J. Demmel, and K. Keutzer, "Fast deep neural network training on distributed systems and cloud tpus," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 11, pp. 2449–2462, Nov. 2019.

[91] E. W. Weisstein, "Circle-circle intersection," *from Wolfram MathWorld*, 2019. [Online]. Available: http://mathworld.wolfram.com/Circle-CircleIntersection.html

[92] R. Costanza, "The ecological, economic, and social importance of the oceans," *Ecological Economics*, vol. 31, no. 2, pp. 199–213, 1999.

[93] J. Castonguay, *International Shipping: Globalization In Crisis*, Vision Project Inc., North Salem, NY, USA, 2019. [Online]. Available: https://www.visionproject.org/images/img_magazine/pdfs/international_shipping.pdf

[94] P. E. Hagen, O. Midtgaard, and O. Hasvold, "Making AUVs truly autonomous," in *OCEANS 2007*, Sep. 2007, pp. 1–4.

[95] M. Mkuseli, "Terminal homing position estimation for autonomous underwater vehicle docking," Master's thesis, Dept. Mechanical and Aerospace Engineering, Naval Postgraduate School, 2017.

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California