Multi-Robot Probabilistic Mapping and Exploration

by Evan Kaufman

B.S. in Mechanical Engineering, May 2012, Bucknell University

A Dissertation submitted to

The Faculty of
The School of Engineering and Applied Science
of The George Washington University
in partial satisfaction of the requirements
for the degree of Doctor of Philosophy

August 31, 2018

Dissertation directed by

Tayeoung Lee
Associate Professor of Mechanical and Aerospace Engineering

The School of Engineering and Applied Science of The George Washington University certifies that Evan Kaufman has passed the Final Examination for the degree of Doctor of Philosophy as of July 13, 2018. This is the final and approved form of the dissertation.

Multi-Robot Probabilistic Mapping and Exploration

Evan Kaufman

Dissertation Research Committee:

Taeyoung Lee, Associate Professor of Mechanical and Aerospace Engineering, Dissertation Director

James Hahn, Professor of Computer Science, Committee Member

Robert Pless, Professor of Computer Science, Committee Member

Chung Hyuk Park, Assistant Professor of Biomedical Engineering, Committee Member

Zhuming Ai, Computer Engineer, U.S. Naval Research Laboratory, Committee Member

Acknowledgments

My academic career has certainly been a long and bumpy road, but I was never alone. I would like thank many people for their support throughout my life for pursuing difficult and interesting work.

First and foremost, I would like to thank my parents. When I was younger, I was not always a great student, but my parents believed in me. When I struggled, they helped every time. I figured that this is what all parents do, but later I realized how lucky I was. Looking back, my academic career not ending prematurely is largely thanks to my mother and father. They have loved me and supported my academic pursuits, and I am extremely grateful.

I would also like to thank all other parts of my family, including my fiancée, brother, grandparents, aunts, uncles, cousins, and close friends. We make it a priority to come together and support each other, even when our careers differ greatly and our lives change unpredictably. Despite the long academic road, they provided consistent support and love.

Finally, I would like to thank my research advisor, Taeyoung Lee, for the excellent guidance over the last six years. Unlike many relationships between advisors and students, he has treated his students like colleagues rather than subordinates. This way, I could focus on completing my research objectives properly rather than trying to impress superior figure. Taeyoung Lee knew just what to say to push my research forward, even when my mind was going another direction. His positive influence over my research has been unparalleled, and I will continue using his valuable lessons.

I think that finding and pursuing a passion is something truly special. Even though I find great satisfaction in doing anything well, regardless of how simple or complicated the task, I find myself extremely fortunate to pursue endeavors that I actually find exciting and stimulating. With tremendous support around me, I begin a career in an amazing field and ready to take on future challenges.

Abstract

Multi-Robot Probabilistic Mapping and Exploration

This dissertation focuses on robotic mapping and exploration of uncertain environments. Computational algorithms are developed to provide complete stochastic information of the environments. These algorithms are designed for real-time implementation for robotic autonomy.

First, probabilistic occupancy grid mapping is developed according to Bayesian framework. A novel approach to this problem is explored, which uses important physical properties of the environment and stochastic properties of depth sensors. We develop an exact solution to occupancy grid probability that can be achieved in real-time using sensor properties and conventional assumptions of an occupancy grid. The rapid computation allows the algorithm to consider large scans of measurements in 2D and 3D environments. The mapping algorithm is demonstrated with several numerical examples and experiments.

The next topic is autonomous exploration, where a robot selects actions to maximize its knowledge about the probabilistic map. We select Shannon's entropy as a metric that represents grid cell uncertainty. Using the earlier contributions on probabilistic occupancy grid mapping, we determine the expected value of entropy change from possible future measurements. This entropy change provides important insights for where a robot should move to maximize its mapping information gain. Dijkstra's search is integral to the algorithm to account for collision-free distances during motion planning. This algorithm is designed for 2D and 3D, where computation time is carefully considered to ensure real-time algorithm performance. Several versions of the exploration algorithm are applied to simulations and experiments.

The final topic of this dissertation relates to multi-vehicle cooperative scenarios. The mapping algorithm is revised to accept measurements from multiple sources with differing sensor properties. More importantly, the exploration algorithm structure is modified with a

bidding-based framework. A series of auctions determines where robots should travel such that the members act together as a team. This solution is further extended to multi-vehicle patrol, where robots begin with an uncertain map, autonomously explore the space, and periodically revisit regions. Autonomous multi-vehicle patrol is accomplished through map degradation, where the probabilistic map becomes more uncertain over time, and the robots must revisit these spaces. These complex algorithms are demonstrated with numerical simulations.

In short, this dissertation proposes novel solutions to probabilistic occupancy grid mapping, autonomous exploration, and patrol in single-vehicle and multi-vehicle scenarios. Real-time implementation is paramount to ensure autonomy during a task. The efficacy of the approaches are shown with several experiments and numerical examples.

Table of Contents

List of Figures

Chapter 1: Introduction

This dissertation explores three key aspects of mobile robotics: occupancy grid mapping, autonomous exploration, and cooperation among robot teams. Several novel contributions are presented on these topics for complex and dynamic situations.

1.1 Motivation and Goals

The study of robotics has expanded tremendously in recent years as robots have replaced humans for dangerous, difficult, and repetitive tasks. Certain objectives, such as search-and-rescue, surveillance, and cleaning, require some level of autonomy, particularly with mobile robots exploring new and uncertain environments. While these tasks provide significant value to our society, they present nontrivial challenges for the robots, particularly in mapping, exploration, and cooperation.

1.1.1 Building Probabilistic Maps

The first topic this dissertation addresses is building accurate probabilistic maps, known as *mapping*. Map generation is crucial for simultaneous localization and mapping (SLAM) because maps provide important information for the robot to determine its pose (localization), which includes its position and attitude. Mapping is also an integral part of vital mobile robot objectives, such as collision avoidance and trajectory planning. In short, mapping is of fundamental importance in mobile robotics.

Therefore, the map *quality* deserves much attention. Robotics is inherently *probabilistic*; no sensor reading or action from an actuator is deterministic. All robotic processes have some degree of uncertainty, and the mapping must account for this. Onboard sensors serve to improve the understanding of surrounding spaces, but the stochastic properties of those sensors must be reflected in the stochastic properties of the map. Therefore, we propose to

develop a probabilistic map that accounts for the history of measurements and poses, and their associated stochastic properties. Here, each element of the map holds a probability based on measured data. In turn, this probability can be applied to tasks such as collision avoidance and predicting future map outcomes.

1.1.2 Exploration in Uncertain Spaces

The second topic of this dissertation focuses on the motions robots must choose to maximize their knowledge about the map while avoiding collisions, known as *autonomous exploration*. Unlike conventional motion planning, where the map of the environment is assumed known, the robot must select motions with only the current knowledge of the environment. Therefore, real-time implementation of an active motion planning scheme is necessary while the probabilistic map is changing.

There are several important aspects to consider in autonomous exploration. Most importantly, the robot motion must be constrained to collision-free paths. The probabilistic map provides collision-free regions for this objective, and it also serves as a graph to determine an ordered set of collision-free waypoints. Additionally, the uncertainty of the map must be considered in the policy governing robotic motion. The method of decreasing map uncertainty can be computationally-expensive, so careful consideration with respect to algorithm complexity is vital. Furthermore, the efficiency of robot motions should be considered to avoid unnecessary travel distances. These objectives must all be achieved for an effective autonomous exploration policy.

1.1.3 Cooperation Among Autonomous Systems

The final topic of this dissertation is cooperation among members of a robot team. All robots share the mutual objective of learning the map as quickly as possible. Each robot must select actions that consider the other team members. These considerations involve avoiding collisions among members, and avoiding actions that provide too much coverage

of one region of the map while another region remains uncertain. Much like single-vehicle autonomous exploration, the multi-vehicle problem requires a computationally-inexpensive algorithm to handle numerous possible robot motions. These nontrivial goals can be handled similarly to single-vehicle autonomous exploration, but following a bidding-based framework.

Multi-vehicle autonomous exploration can be extended further for surveillance purposes with autonomous patrol. The goal is to periodically capture many regions within a large space. Frequently, the robots are subject to dynamic obstacles (e.g. a walking person) or a changing environment (e.g. moving a chair), so the robots must account for these as well. Furthermore, an effective patrol policy can decrease the number of robots required to patrol an environment. The multi-vehicle autonomous exploration can be integrated with probabilistic map degradation to achieve autonomous patrol.

1.2 Literature Review

In this section, we discuss the existing approaches to occupancy grid mapping, autonomous exploration, and multi-vehicle cooperation. Several shortcomings with existing approaches are highlighted, which are addressed in the dissertation contributions.

1.2.1 Occupancy Grid Mapping

There are several existing mapping representations including occupancy grids [60, 60, 44, 56], Octomaps [62, 24], or feature-based maps [45]. In this dissertation, we focus on occupancy grid maps because of their simple structure, efficient memory usage, and natural application to autonomous exploration in later sections. Occupancy grid mapping follows a Bayesian framework that focusses on occupied and free space, not tracking particular systems or mapping features, such as visual servoing [11].

Here we describe the existing approaches to building probabilistic occupancy grid maps. Though several variations have been proposed, no approach uses the stochastic properties

of depth sensors directly to obtain an exact Bayesian solution that can be computed in real-time.

Inverse Sensor Model The key to generating a probabilistic occupancy grid map is known as the *inverse sensor model* [59]. The main idea is that a depth sensor has a probability density (e.g. Gaussian) that describes the stochastic properties of a sensor reading (e.g. a LIDAR range) with respect to the expected value of a reading (e.g. distance to a wall). This probability density is referred to as the *forward sensor model*, as it relates a depth measurement range to the occupancy of the map.

In contrast, the inverse sensor model relates the occupancy of the map to the depth measurement range. This problem is solved using Bayes' theorem. However, the occupancy grid map properties yield a complicated Bayesian probability; the computational cost has exponential complexity with respect to the number of grid cells captured within the depth sensor limits. It was previously-believed that this restriction makes solving the inverse sensor model in real-time impossible. This motivated approximate or estimated solutions, described next.

Approximated Function for the Inverse Sensor Model Moravec and Elfes originally proposed a grid cell-based mapping for sonar sensors in [46], and the probabilistic properties of the occupancy grid were formalized in [46, 18]. This approach models measurements with Gaussian probability density, although the occupancy of grid cells is not exactly known. Instead, the probability of grid cells are estimated with a Gaussian-like *approximate function*. The authors show that occupancy probability converges to a final value following a Bayesian framework, though there is no conclusive proof that this probability is accurate. Nevertheless, this approach garnered much attention because it could efficiently produce robotic maps that nicely-resembled the occupancy of spaces around a robot.

Several other research studies have applied approximate functions in place of the inverse sensor model. For example, [13] proposed a simplified continuous function composed of

lines patched together near a measurement reading. Furthermore, variations of approximate smoother functions in [7, 48, 34] contain Gaussian-like terms to apply occupancy grid mapping to 3D environments with more modern sensors. Numerous research studies have used the proposed approximate functions with questionable accuracy.

Learning the Inverse Sensor Model The approximate nature of the inverse sensor model motivated a solution using machine learning. The main idea is to simulate maps, robot poses, and measurements with simulated noised based on the sensor stochastic properties. Then, an approximation the inverse sensor model is learned using an expectation maximization algorithm [57, 59, 6]. Unlike approximate functions based on intuition, the machine learning approach is based on the physical properties of sensors. However, this approach is undesirable in practice due to complexities associated to implementing a learning algorithm. For example, the accuracy of such inverse sensor models strongly depends on the samples selected for learning, but it is unclear how to select those samples, or how many samples are required to obtain a reasonable approximation. Furthermore, it is challenging to apply any learning algorithm over the large dimensional space composed of maps, poses, and measurements.

Log-Odds Ratio Both approaches described above use the log-odds ratio to update grid cell probabilities. The log-odds ratio formulation follows a popular framework for updating binary random variables with static state within a Bayesian filter. The main idea is that instead of multiplying terms from prior and current time steps, the properties of logarithms allow these terms to be simply added, while avoiding some truncation issues associated with probabilities close to 0 or 1 [59, 58]. This approach is also popular because the inverse sensor model needs not consider prior probabilities. Instead, these approaches typically consider fixed uniform initial probabilities of all grid cells, which makes the formulation simpler.

However, the log-odds ratio formulation makes an assumption that is not consistent

with the occupancy grid mapping problem. The approach yields a simplified solution by assuming that the probability of a measurement is independent of past measurements and robot poses. This assumption is frequently violated when past measurements and poses indicate the occupancy of other cells between the robot and the cell in question. As such, this approach neglects potentially important information when considering grid cell occupancy probability.

In short, other existing solutions to occupancy probability are inexact, and involve a potentially-harmful log-odds ratio assumption. These shortcomings motivate an exact solution to occupancy grid mapping that can be computed in real-time.

1.2.2 Autonomous Exploration

The next topic this dissertation addresses is autonomous exploration. The vast majority of work in simultaneous localization and mapping (SLAM) deals only with the aspect of estimating the environment and the poses of vehicles. These approaches are passive in the sense that SLAM is performed on incoming sensor measurements from vehicles following an arbitrary path (e.g. [59, 17, 10]). As such, human teleoperation and monitoring are often essential to guide the vehicles safely through unknown surroundings. Therefore, it is desirable that an active motion planning scheme is developed and integrated with SLAM such that vehicles are able to determine their paths without human intervention, and explore unknown areas autonomously. The two approaches used to solve this problem are described next.

Frontier-Based Autonomous Exploration The most popular approach to solve the autonomous exploration problem is known as frontier-based exploration, originally-proposed in [63, 64] for 2D applications, and was later extended to 3D with a visibility metric in [51] and [67, 52, 5] using the popular Octomap representation (a variation on occupancy grid mapping). The main idea is that robots move toward the border between certain and uncer-

tain space, referred to as a frontier. Then the robot takes depth measurement at this frontier, thus pushing back the boundary. This process is repeated until the map is well-known. Frontier-based exploration assumes that repeatedly moving toward the closest frontier and taking measurements are the best actions to gain new information about the map. However, these systematic actions, based on ad-hoc rules for what constitutes a frontier, do not consider the future uncertainty of the probabilistic map. Frontier-based approaches provide an intuitive solution, but lack optimality in map uncertainty or exploration time.

Entropy-Based Autonomous Exploration Entropy-based approaches address the suboptimal nature of frontier-based approaches by selecting robotic actions designed to decrease map uncertainty [9, 55, 8]. These approaches use a measure of probabilistic uncertainty known as Shannon's entropy, which becomes smaller as cell probabilities approach 0 or 1 (becoming more certain). Existing entropy-based approaches simulate possible future measurements from various locations with so-called "hallucination measurements," which typically corresponds to expected depth measurements from the robot to the closest grid cell that is possibly occupied. Then, the robot analyzes how this measurement would affect the map uncertainty. However, this approach assumes that expected map entropy is equivalent to map entropy from expected measurements [27]. This is not the case in general, however, as there is typically a nonzero probability that the hallucination measurement and true future measurement are not the same. Entropy-based approaches seek to optimize map uncertainty, but require large approximations which can decrease the accuracy of predictions.

In conclusion, existing autonomous exploration approaches are effective for eliminating the need for humans to control the robot. However, optimality remains quite difficult, so these approaches apply suboptimal or approximated solutions to achieve autonomous motions.

1.2.3 Multi-Vehicle Cooperation

The final topic relates to coordinating robotic efforts to build occupancy grid maps together. These concepts are further extended to autonomous cooperative patrol, where robots periodically revisit regions to monitor a large environment.

Cooperative Autonomous Exploration Multi-vehicle cooperation is essential for robots to explore large spaces autonomously so that robots coordinate their efforts to improve map information and avoid collisions. However, optimizing map coverage simultaneously from multiple vehicles becomes very complicated and computationally intractable. Instead, an auction-based approach addresses the computational restrictions effectively. Bidding-based approaches were introduced with [53, 15], and were extended to several other robotic problems [20, 68, 50, 12] exploring decentralization and task allocation with varying robot capabilities. In [53], a central executive assigns tasks to robots for maximizing overall utility of the group. This is accomplished by minimizing a coverage overlap among the members. Robots submit bids based on expected coverage of uncertain spaces and travel distance. The robot winning the auction is awarded the task, and the remaining bids for subsequent auctions are discounted to prevent robots from covering the same area, thereby coordinating the mapping efforts. Bidding-based approaches simplify multi-vehicle cooperation in a computationally-efficient manner, and integrate effectively with the autonomous mapping and exploration approaches proposed in this dissertation.

Autonomous Patrol Robotic surveillance and cleaning are examples of tasks that require periodic observations of the same regions. Several patrolling approaches have been proposed in recent years to address this need. A popular approach is decomposing the reachable space into Voronoi diagrams [36, 49, 47]. In [36], edges between nodes of the Voronoi graph are heavily sensed to track intruders. In [49], regions of the Voronoi diagram are assigned to different robots, and issues with robots having underperforming surveillance capabilities are

addressed in [47]. These approaches assume the map is well-known in advance, and require a Voronoi graph be generated from the map.

Other patrol approaches create new metrics about the environment. Cell periodicity is estimated in [37, 38] by predicting and observing the long-term dynamics of objects at the same locations on an occupancy grid. This work is extended to exploring and navigating a human-populated environment for long-term autonomy. A bio-inspired approach is proposed in [66], where robots cooperate by dropping digital pheromones based on the importance of events, where the pheromones weaken over time to prioritize future tasks. Another approach is formulating patrol as an optimization problem with respect to robot idleness in [65]. The prior work in autonomous patrol demonstrates that periodic actions can be incentivized with a metric to be optimized.

1.3 Outline of Dissertation

This dissertation is organized as follows. Chapter 2 introduces a novel approach to solve for the exact Bayesian probability of grid cells in real-time. Then, Chapter 3 extends these contributions to autonomous exploration in 2D space, where a computational approach to predict future map uncertainty is presented. Next, Chapter 4 applies the proposed occupancy grid mapping approach to 3D environments, and presents a method to simplify autonomous exploration in 3D environments with map projections. Chapter 5 applies this 3D mapping and exploration to multi-vehicle exploration and patrol scenarios using a bidding-based framework. Then, Chapter 6 extends exploration to complex 3D environments without 2D map projections. In these chapters, several numerical examples demonstrate the efficacies of the approaches. Furthermore, Chapter 7 presents three experiments, which show the key contributions in this dissertation. Finally, these are followed by conclusions in Chapter 8.

1.4 Contributions

There are three key contributions in this dissertation. The first two are novel solutions that find exact occupancy probabilities and predicted entropies, respectively. These contributions improve the quality and efficiency of autonomous mapping and exploration of uncertain environments for single-vehicle missions. The third contribution extends these concepts to multi-vehicle scenarios, where multiple members of a team work together to autonomously explore and patrol large environments.

1.4.1 Summary of Contributions

The first contribution is an exact Bayesian solution to occupancy grid mapping. The proposed approach considers that a depth measurement captures the closest occupied space, which allows certain mapping outcomes to be systematically grouped together. The exact solution to occupancy grid cell probability, which was previously-believed to have exponential complexity, is now possible to solve in real-time. Numerous numerical simulations and experiments demonstrate that the proposed mapping approach is effective and inexpensive in 2D and 3D scenarios.

The second contribution is a new approach to autonomous exploration, where the robot predicts future occupancy grid map uncertainty, using a measure known as Shannon's entropy. This approach computes the expected values of grid cell entropies from potential future measurement rays with a novel approach to predict future measurement probabilities. These predictions are used to evaluate the benefit of possible robot actions to maximize map information gain. Then, we formulate autonomous exploration as an optimization problem to maximize an objective function that includes map uncertainty and travel costs. The probabilistic map is used for collision-avoidance and motion planning. These processes are designed for real-time applications, where the robot makes decisions as it learns information about the surrounding environment.

The final contribution coordinates the mapping and exploration efforts of multiple robots together. The mapping process is multi-threaded such that multiple members update the same occupancy grid simultaneously. The exploration algorithm follows a bidding-based structure. Robots compete for tasks in a series of auctions, where the winning bid of one auction affects the bids of subsequent auctions, thereby coordinating the robot efforts. This cooperative approach is computationally-efficient in large environments, and follows a receding-horizon framework, such that map expected information gains are updated as quickly as possible. The proposed coordination is extended to autonomous patrol, where occupancy grid map probabilities are degraded over time, which promotes the robots to revisit areas periodically.

1.4.2 List of Publications

The following is a list of publications. These cover research included in this dissertation, as well as contributions in control, data association, and estimation.

- E. Kaufman, K. Takami, T. Lee, and Z. Ai, "Multi-Vehicle Cooperative Exploration and Patrol of Uncertain 3D Environments", in preparation.

- E. Kaufman and T. Lee, "Autonomous Aerial Exploration for Topological Mapping of Mars Environments", *2019 AIAA Guidance, Navigation, and Control Conference*, San Diego, CA, January 2019, submitted.

- E. Kaufman, K. Takami, Z. Ai, and T. Lee, "Autonomous Quadrotor 3D Mapping and Exploration Using Exact Occupancy Probabilities," *The Second IEEE International Conference on Robotic Computing*, pp. 49–55, Laguna Hills, CA, January 2018.

- E. Kaufman, K. Takami, T. Lee, and Z. Ai, "Autonomous Exploration with Exact Inverse Sensor Models," *Journal of Intelligent & Robotic Systems*, 2017, doi: 10.1007/s10846-017-0710-7.

- E. Kaufman, T. Lee, and Z. Ai, "Autonomous Exploration by Expected Information Gain from Probabilistic Occupancy Grid Mapping," *IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, pp. 246–251, San Francisco, CA, December 2016.

- E. Kaufman, T. Lee, Z. Ai, and I. S. Moskowitz, "Bayesian Occupancy Grid Mapping via an Exact Inverse Sensor Model," *Proceedings of the American Control Conference*, pp. 5709-5715, Boston, MA, July 2016.

- E. Kaufman, T. A. Lovell, and T. Lee, "Nonlinear Observability Measure for Relative Orbit Determination with Angles-Only Measurements," *The Journal of the Astronautical Sciences*, 63(1): pp. 60-80, 2016, doi: 10.1007/s40295-015-0082-9.

- E. Kaufman, T. A. Lovell, and T. Lee, "Minimum Uncertainty JPDA Filters and Coalescence Avoidance for Multiple Object Tracking," *The Journal of the Astronautical Sciences*, 63(4): pp. 308-334, 2016, doi: 10.1007/s40295-016-0092-2.

- T. Wu, E. Kaufman, and T. Lee, "Globally Asymptotically Stable Attitude Observer on SO(3)," *Proceedings of the 54th IEEE Conference on Decision and Control*, pp. 2164-2168, Osaka, Japan, December 2015.

- E. Kaufman, T. A. Lovell, and T. Lee, "Nonlinear Observability Measure for Relative Orbit Determination with Angles-Only Measurements," *Proceedings of the 25th AAS/AIAA Space Flight Mechanics Meeting*, AAS 15-451, Williamsburg, VA, January 2015.

- E. Kaufman, T. A. Lovell, and T. Lee, "Minimum Uncertainty JPDA Filter and Coalescence Avoidance Performance Evaluations," *Proceedings of the 25th AAS/AIAA Space Flight Mechanics Meeting, Williamsburg*, AAS 15-432, VA, January 2015.

- E. Kaufman, K. Caldwell, D. Lee, and T. Lee, "Design and Development of a Free-Floating Hexrotor UAV for 6-DOF Maneuvers," *Proceedings of the IEEE Aerospace*

Conference, ASC 14-2527, March 2014, Big Sky, MT.

- E. Kaufman, T. A. Lovell, and T. Lee, "Optimal Joint Probabilistic Data Association Filter Avoiding Coalescence in Close Proximity," *Proceedings of the European Control Conference*, pp. 2709-2714, Strasbourg, June 2014.

Chapter 2: Probabilistic Occupancy Grid Mapping

In this chapter, we present the exact solution to occupancy grid map probability for a given depth measurement. This Bayesian approach uses the stochastic properties of sensors, and exploits important patterns in conditional probabilities. Then we show how several measurements can be considered together in 2D. This solution is simulated with numerical examples to show the efficacy of the approach.

2.1 Mapping Problem Definition

Let a map m be decomposed into n_m evenly-spaced grid cells, where the i-th grid cell is assigned to a static binary random variable $\mathbf{m}_i$ for $i \in \{1, 2, \ldots, n_m\}$, that is defined as $\mathbf{m}_i = 1$ when occupied, and $\mathbf{m}_i = 0$ when free. The location and size of each grid cell is assumed known, where a smaller cell size (greater grid resolution) better represents a space, but increases computation and memory. Therefore, a map m is defined by $\{\mathbf{m}_1, \ldots, \mathbf{m}_{n_m}\}$ (2^{n_m} possible maps).

Another random variable is defined as $\bar{\mathbf{m}}_i = 1 - \mathbf{m}_i$ for convenience. The probability that the i-th cell is occupied is $P(\mathbf{m}_i)$, and the probability that it is free is $P(\bar{\mathbf{m}}_i) = 1 - P(\mathbf{m}_i)$. The random variables $\mathbf{m}_i$ are mutually independent, i.e.,

$$P(m) = P(\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_{n_m}) = \prod_{i=1}^{n_m} P(\mathbf{m}_i). \tag{2.1}$$

Consider a range sensor that provides scans of the surrounding environment in order to identify the closest occupied space. The location and the direction of the sensor, referred to as the *pose* at time t, is denoted by $X_t = (x_t, R_t)$. In a 2D environment, the planar position and direction of the robot at t are denoted by $x_t \in \mathbb{R}^2$ and $R_t \in \mathsf{S}^1 = \{q \in \mathbb{R}^2 | \, \|q\| = 1\}$, i.e., the attitude corresponds to a direction on a 2D plane.

Similarly in a 3D environment, the spacial position and attitude are denoted by $x_t \in \mathbb{R}^3$ and $R_t \in \mathsf{SO}(3) = \left\{R \in \mathbb{R}^{3\times 3} | R^{\mathsf{T}} R = I, \det R = 1\right\}$. Let $X_{1:t}$ denote the history of poses from the initial time to the current time, i.e., $X_{1:t} = \{X_1, X_2, \ldots, X_t\}$. At each pose, the robot receives a 2D measurement *scan*, composed of n_z measurement *rays*. These rays are 1D depth measurements of known direction from the current pose to the closest occupied space, subject to a known *forward sensor model* $p(z_{t,l}|m, X_t)$, where $z_{t,l}$ is the l-th measurement ray of the t-th scan $Z_t = \{z_{t,1}, z_{t,2}, \ldots, z_{t,n_z}\}$. The measurement history is denoted by $Z_{1:t} = \{Z_1, Z_2, \ldots, Z_t\}$.

The probability density function, namely $p(z_{t,l}|m, X_t)$ with respect to the depth of the l-th measurement ray conditioned on the map m and the pose X_t is commonly referred to as the *forward sensor model*, which characterizes the corresponding depth sensor, such as the maximum range or accuracy. The forward sensor model satisfies (i) the ranges of all depth measurements are positive and finite, and (ii) a measurement cannot pass through occupied regions. Throughout this paper, we assume that the forward sensor model of the selected sensor is given. This can be determined empirically or analytically. For example, the *beam model for range finders* satisfying the above criteria is described in [59].

Occupancy grid mapping provides occupancy probabilities based on robot poses and measurement scans. The goal is to obtain $P(\mathbf{m}_i|X_{1:t}, Z_{1:t})$, commonly referred to as the *inverse sensor model* for a given forward sensor model $p(z_{t,l}|m, X_t)$ and the initial estimate of the map $P(m)$ with $X_{1:t}$ and $Z_{1:t}$.

2.2 Inverse Sensor Model

In this section, we propose an algorithm to compute the exact inverse sensor model efficiently. Suppose the probability of the map conditioned on the past poses and measurements, namely $P(m|X_{1:t-1}, Z_{1:t-1})$, is known. Here we construct a posteriori probability $P(m|z_{t,l}, X_{1:t}, Z_{1:t-1})$, based on the current pose X_t, the measurement from the l-th ray $z_{t,l}$, and the given forward sensor model $p(z_{t_l}|m, X_t)$.

2.2.1 Bayesian Framework

The occupancy probability $P(m|z_{t,l}, X_{1:t}, Z_{1:t-1})$ is based on the forward sensor model $p(z_{t,l}|m, X_{1:t}, Z_{1:t-1})$ that describes the distribution of the measurements for the given robot pose and the map. This specifies the stochastic characteristics of the sensor are a known distribution, specific to a particular depth sensor. One could certainly obtain a stochastic model by fitting a probability density to a controlled set of measurements. Bayes' rule yields

$$P(m|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \frac{p(z_{t,l}|m, X_{1:t}, Z_{1:t-1})P(m|X_{1:t-1}, Z_{1:t-1})}{p(z_{t,l}|X_{1:t}, Z_{1:t-1})}, \tag{2.2}$$

where the second term in the numerator considers that X_t carries no information about m without Z_t. If the current pose X_t and map m are known, then the measurement ray $z_{t,l}$ is independent of past poses $X_{1:t-1}$ and past measurements $Z_{1:t-1}$ to obtain

$$P(m|z_{t,l},X_{1:t}, Z_{1:t-1}) = \eta_{t,l}p(z_{t,l}|m, X_t)P(m|X_{1:t-1}, Z_{1:t-1}), \tag{2.3}$$

where the normalizing constant $\eta_{t,l} \in \mathbb{R}$ absorbs all terms that do not depend on the map m. Next, we compute the occupancy probability of each cell. Let $\mathcal{M}_i$ be the set of maps where the i-th cell is occupied, i.e., $\mathcal{M}_i = \{m \in \{0,1\}^{n_m} \mid \mathbf{m}_i = 1\}$. To compute the probability of occupancy of the i-th cell, all possible combinations of map in $\mathcal{M}_i$ should be considered, i.e.,

$$P(\mathbf{m}_i|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \eta_{t,l} \sum_{m \in \mathcal{M}_i} p(z_{t,l}|m, X_t)P(m|X_{1:t-1}, Z_{1:t-1}). \tag{2.4}$$

Furthermore, to determine the normalizing constant $\eta_{t,l}$, the complement $P(\bar{\mathbf{m}}_i|z_{t,l}, X_{1:t}, Z_{1:t-1})$ must be calculated in the similar manner. Since there are 2^{n_m-1} maps in $\mathcal{M}_i$, these equations require 2^{n_m} terms to compute the summation for the i-th grid cell and normalizer, and this process should be repeated for all other cells along the measurement ray. This is the main reason why the existing results are based on approximations or learned

solutions of the above expression.

2.2.2 Computationally-Efficient Solution

We propose a computational algorithm to evaluate (2.4) efficiently [31, 32]. Since the cells outside of the sensor field of view (FOV) are not affected, we focus on a reduced map r_l in the FOV of the l-th ray. This reduced map is chosen such that each cell of r_l corresponds to a grid cell of map m that the l-th ray intersects, ordered by increasing distance. Let $\mathbf{r}_{l,k}$ be the binary random variable representing the occupancy of the k-th cell of the l-th ray. The number of cells in the reduced map is denoted by $n_{r,l} \leq n_m$.

Next, let $\mathbf{r}_{l,k+}$ correspond to the event that the k-th cell of the l-th ray is occupied, cells with lower index (closer cells) are free, and cells with greater index (farther cells) may or may not be occupied, i.e., event $\mathbf{r}_{l,k+}$ occurs when
$r \in \{r \in \{0,1\}^{n_{r,l}} \,|\mathbf{r}_{l,1} = 0, \mathbf{r}_{l,2} = 0, \ldots, \mathbf{r}_{l,k-1} = 0, \mathbf{r}_{l,k} = 1\}$. Put differently, the k-th cell $\mathbf{r}_{l,k}$ is the closest occupied cell to current pose X_t along the l-th ray. For example, see Figure 2.1 that illustrates the event of $\mathbf{r}_{l,k+}$ when $l = 1$ and $k = 4$ for an one-dimensional cell array. This concept of grouping map outcomes can be easily extended to 2D using ray casting, where a 1D ray is spanning 2D space, which is easily determined from geometry, described in further detail in Section 2.3.1. Then, the forward sensor model is identical for all maps defined by $\mathbf{r}_{l,k+}$, regardless of the occupancy of the cells beyond the k-th cell, and the corresponding forward sensor model $p(z_{t,l}|\mathbf{r}_{l,k+}, X_t)$ depends on the distance from X_t to the k-th cell. Based on this, we present the mathematical expressions for the exact inverse sensor model, as summarized by (2.5) in Proposition 1. Later in Section 2.3.2, these are also rearranged in computational algorithms.

Proposition 1. *For the l-th measurement ray, the a posteriori probability of the occupancy of the k-th cell, namely the ray inverse sensor model, is given by*

$$P(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \eta_{t,l}\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}), \tag{2.5}$$

Figure 2.1: Occupancy Grid Maps with Identical Forward Sensor Models

A robot located to the left of a 1D occupancy grid map r_l composed of $n_{r,l} = 6$ grid cells in four cases. In each case, cells the first three cells are free, the fourth cell is occupied, and the fifth and sixth cells may or may not be occupied. All above outcomes correspond to the event $\mathbf{r}_{l,4+}$, and therefore share the same forward sensor model.

where the unnormalized probability of the inverse sensor model is defined as

$$\begin{aligned}\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) &= P(\mathbf{r}_{l,k}|X_{1:t-1}, Z_{1:t-1}) \\ &\times \left[\sum_{i=1}^{k-1}\left\{\prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1})\right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1})\right] \\ &+ \left\{\prod_{j=0}^{k-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1})\right\} p(z_{t,l}|\mathbf{r}_{l,k+}, X_t) P(\mathbf{r}_{l,k}|X_{1:t-1}, Z_{1:t-1}),\end{aligned} \tag{2.6}$$

where $P(\bar{\mathbf{r}}_{l,0}|X_{1:t-1}, Z_{1:t-1}) = P(\mathbf{r}_{l,n_r+1}|X_{1:t-1}, Z_{1:t-1}) = 1$ *is chosen for convenience and* $p(z_{t,l}|\mathbf{r}_{l,(n_r+1)+}, X_t)$ *represents the probability density of the measurement when all of cells in the field of view of the* l*-th ray are not occupied. The normalizer* $\eta_{t,l}$ *is given by*

$$\eta_{t,l} = \left[\sum_{i=1}^{n_{r,l}+1}\left\{\prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1})\right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1})\right]^{-1}, \tag{2.7}$$

and it is independent of the cell index k*.*

Proof. See Appendix A. □

Since $\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})$ from (2.6) uses several repeated terms from the prior $\tilde{P}(\mathbf{r}_{l,k-1}|z_{t,l}, X_{1:t}, Z_{1:t-1})$, and $\eta_{t,l}$ is easily obtained from these as well, the computational cost of (2.5) is linear with respect to the number of cells along a measurement ray, amortized to $\mathcal{O}(1)$ for each cell. Because of this substantial computational improvement, the exact inverse sensor model can be applied in real-time. The process of updating each cell along a

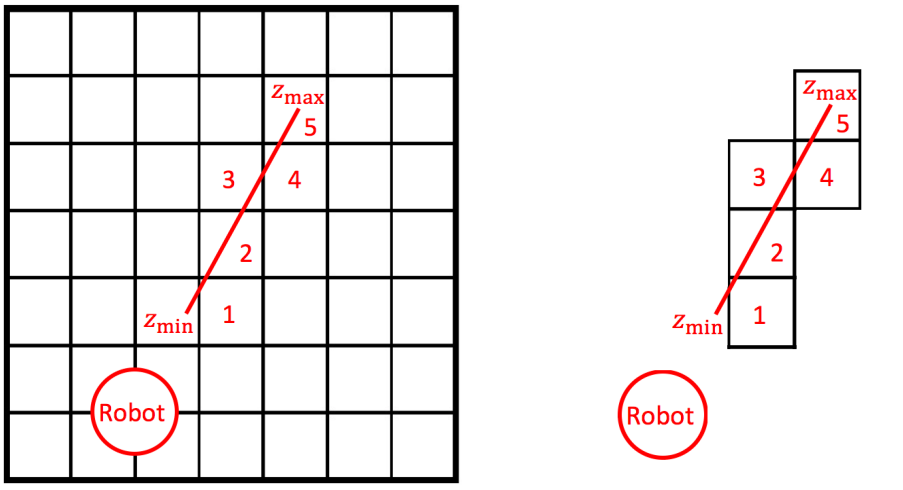
measurement ray is repeated for all rays composing scan Z_t.

2.3 Mapping in 2D Space

The above formulations describe how a single ray updates grid cells along its 1D path. Next we describe how measurement rays can update 2D maps, and how large scans of measurement rays are handled together.

2.3.1 Ray Casting

Ray casting is the process of determining which cells a measurement ray might intersect, and the distances to these cells. The process is straightforward: follow a measurement unit vector from its minimum range z_{min} to maximum range z_{max}, and identify the edges of all grid cells that it intersects. Save the cell index and Cartesian distance from the robot, and re-order the cells by increasing distance. An illustration of a simple example is shown in Figure 2.2.



(a) Ray Casting on a 2D Grid

(b) Reduced Map Only

Figure 2.2: Ray Casting Illustration

This simple 2D example illustrates how ray casting determines the cells that compose the reduced map for the inverse sensor model. The closest edges are used to determine which cells should be considered, and in what order. The complete map cell indices are temporarily saved as well, which are used to associate probabilities to cells of the reduced map.

2.3.2 Combining Measurements from a Single Scan

Typically, many measurement rays are provided from a single scan from modern sensors. Next, we describe two approaches to combine the inverse sensor models of numerous measurement rays.

Ray-By-Ray Approach At the t-th time step, consider the measurement scan $Z_t = \{z_{t,1}, z_{t,2}, ..., z_{t,n_z}\}$. One may apply the results of Proposition 1 repeatedly for each ray to obtain a two-dimensional inverse sensor model because

$$P(\mathbf{m}_i|X_{1:t}, Z_{1:t}) = P(\mathbf{m}_i|X_{1:t}, Z_{1:t-1}|z_{t,1}|z_{t,2}|...|z_{t,n_z}). \tag{2.8}$$

In this approach, each ray is considered individually such that the inverse sensor model updates the probabilities of grid cells from one ray before subsequent rays of Z_t are considered. Therefore, we assume the first measurement ray $z_{t,1}$ is independent of the remaining rays $z_{t,2:n_z}$. For the second ray, namely $z_{t,2}$, this may depend on $z_{t,1}$, but not subsequent rays $z_{t,3:n_z}$. By the time z_{t,n_z} is considered, it may depend on any combination of prior rays $z_{t,1:n_z-1}$. Therefore, the ray-by-ray approach considers rays within a single scan as partially dependent on each other.

In short, the proposed scan inverse sensor model is computed by (2.5)-(2.8). These are summarized in Algorithm 1 as a computationally efficient, recursive algorithm that avoids several repeated calculations of the same quantity. This algorithm utilizes temporary variables, where the required computational resources are minimized by computing the updated occupancy probabilities of all grid cells along a measurement ray together. Note that this function is only considering l-th ray at the t-th time step, where probabilities are subject to conditions on the history of poses $X_{1:t-1}$ and measurement scans $Z_{1:t-1}$, so these

are removed from the algorithm for simplicity.

Algorithm 1: Inverse Sensor Model of a Scan Updated Ray-By-Ray

Function: $P(m|X,Z) = \text{InverseSensorModelRayByRay}(P(m), X, Z)$;

for $l = 1, 2, \ldots, n_l$ **do**

  Find $n_{r,l}$ cells from m corresponding to z_l;

  Initialize $\eta_l^{-1} = 0$ and $P(\bar{\mathbf{r}}_{l,0}|X, z_{1:l-1}) = 1$ ($z_{1:0}$ is no condition);

  for $k = 1, 2, \ldots, n_{r,l}$ **do**

    $P(\mathbf{r}_{l,k+}|X, z_{1:l-1}) = P(\bar{\mathbf{r}}_{l,0:k-1}|X, z_{1:l-1})P(\mathbf{r}_{l,k}|X, z_{1:l-1})$;

    $P(\bar{\mathbf{r}}_{l,0:k}|X, z_{1:l-1}) = P(\bar{\mathbf{r}}_{l,0:k-1}|X, z_{1:l-1})(1 - P(\mathbf{r}_{l,k}|X, z_{1:l-1}))$;

    $a_{\text{temp}} = P(\mathbf{r}_{l,k+}|X, z_{1:l-1})p(z_l|\mathbf{r}_{1:l,k+}, X)$;

    $\tilde{P}(\mathbf{r}_{l,k}|X, z_{1:l}) = P(\mathbf{r}_{l,k}|X, z_{1:l-1})\eta_l^{-1} + a_{\text{temp}}$;

    $\eta_l^{-1} = \eta_l^{-1} + a_{\text{temp}}$;

  $\eta_l^{-1} = \eta_l^{-1} + P(\bar{\mathbf{r}}_{l,0:n_{r,l}}|X, z_{1:l-1})p(z_{\max})$;

  $P(\mathbf{r}_k|z_{1:l}) = \eta_l \tilde{P}(\mathbf{r}_k|X, z_{1:l})$ for all $k \in \{1, 2, \ldots, n_{r,l}\}$;

  Substitute $P(\mathbf{r}_k|X, z_{1:l})$ back into $P(m|X, z_{1:l})$;

Return $P(m|X, z_{1:n_l}) = P(m|X, Z)$;

Synergistic Update Approach In this approach, all rays from a single scan are considered simultaneously for a synergistic update of all grid cells falling inside the scan FOV. However, the subsequent formulations require the assumption that all measurement rays within the scan are mutually independent, i.e.,

$$p(z_{t,1}, z_{t,2}, \ldots, z_{t,n_l}|\mathbf{m}_i, X_{1:t}, Z_{1:t-1}) = \prod_{l=1}^{n_l} p(z_{t,l}|\mathbf{m}_i, X_{1:t}, Z_{1:t-1}).$$

Here, we construct such two-dimensional inverse sensor model, referred to as the synergistic scan inverse sensor model. Let $\mathcal{L}_i \subset \{1, \ldots, n_l\}$ be the set of rays that pass through the i-th cell, $\mathbf{m}_i$. Applying Bayes' rule repeatedly, we obtain

$$\begin{aligned} P(\mathbf{m}_i|X_{1:t}, Z_{1:t}) &= \tilde{\zeta}_i \left\{ \prod_{l\in\mathcal{L}_i} p(z_{t,l}|\mathbf{m}_i, X_{1:t}, Z_{1:t-1}) \right\} P(\mathbf{m}_i|X_{1:t-1}, Z_{1:t-1}) \\ &= \zeta_i P(\mathbf{m}_i|X_{1:t-1}, Z_{1:t-1}) \prod_{l\in\mathcal{L}_i} \frac{P(\mathbf{m}_i|z_{t,l}, X_{1:t}, Z_{1:t-1})}{P(\mathbf{m}_i|X_{1:t-1}, Z_{1:t-1})}, \end{aligned} \tag{2.9}$$

where $\tilde{\zeta}_i, \zeta_i \in \mathbb{R}$ correspond to normalizing constants independent of $\mathbf{m}_i$.

Suppose the l-th measurement ray intersects with the cell $\mathbf{m}_i$, and it is the k-th cell of the corresponding reduced map, i.e., $\mathbf{r}_{l,k}$ corresponds to the cell $\mathbf{m}_i$ in the l-th reduced map, and $\mathbf{r}_{l,k} = \mathbf{m}_i$ since they represent the same cell. Then, we have $P(\mathbf{m}_i|z_{t,l}, X_{1:t}, Z_{1:t-1}) = P(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \eta_{t,l}\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})$.

Using this, (2.9) is rewritten as

$$P(\mathbf{m}_i|X_{1:t}, Z_{1:t}) = \xi_i P(\mathbf{m}_i|X_{1:t-1}, Z_{1:t-1}) \prod_{l\in\mathcal{L}_i} \hat{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}), \tag{2.10}$$

where the normalizer ξ_i is composed of the product of normalizers ζ_i, and η_i. Here, we

introduce a new term $\hat{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})$ for computational efficiency as

$$\begin{aligned}
&\hat{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) \triangleq \frac{\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})}{P(\mathbf{m}_i|X_{1:t-1}, Z_{1:t-1})} \\
&= \sum_{i=1}^{k-1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}) \\
&\quad + \left\{ \prod_{j=0}^{k-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,k+}, X_t),
\end{aligned} \tag{2.11}$$

where we have used (2.6). Similarly, its complement is

$$P(\bar{\mathbf{m}}_i|X_{1:t}, Z_{1:t}) = \xi_i P(\bar{\mathbf{m}}_i|X_{1:t-1}, Z_{1:t-1}) \prod_{l \in \mathcal{L}_i} \hat{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}), \tag{2.12}$$

where $\hat{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})$ is defined as

$$\begin{aligned}
&\hat{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) \triangleq \frac{\tilde{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})}{P(\bar{\mathbf{m}}_i|X_{1:t}, Z_{1:t-1})} \\
&= \sum_{i=1}^{k-1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}) \\
&\quad + \frac{\sum_{i=k+1}^{n_{r,l}+1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1})}{P(\bar{\mathbf{r}}_{l,k}|X_{1:t-1}, Z_{1:t-1})},
\end{aligned} \tag{2.13}$$

which is obtained from (A.9). Since

$$P(\mathbf{m}_i|X_{1:t}, Z_{1:t}) + P(\bar{\mathbf{m}}_i|X_{1:t}, Z_{1:t}) = 1,$$

the normalizer ξ_i is obtained using (2.10) and (2.12) as,

$$\begin{aligned}
\xi_i = \Bigg[&P(\mathbf{m}_i|X_{1:t-1}, Z_{1:t-1}) \prod_{\mathcal{L}_i} \hat{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) \\
&+ P(\bar{\mathbf{m}}_i|X_{1:t-1}, Z_{1:t-1}) \prod_{\mathcal{L}_i} \hat{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) \Bigg]^{-1},
\end{aligned} \tag{2.14}$$

which is substituted into (2.10) to obtain the complete scan inverse sensor model $P(\mathbf{m}_i|X_{1:t}, Z_{1:t})$.

The synergistic scan inverse sensor model algorithm is summarized with the pseudo-code of Algorithm 2 as a computationally efficient, recursive algorithm that avoids repeated calculations of the same quantity. This algorithm utilizes the following temporary variables to develop the algorithm in an efficient recursive form, defined as

$$
\begin{aligned}
a_k &= \sum_{i=1}^{k-1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,i}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}), \\
b_k &= \prod_{j=0}^{k-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}), \\
c_k &= \frac{\sum_{i=k+1}^{n_{r,l}+1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1})}{P(\bar{\mathbf{r}}_{l,k}|X_{1:t-1}, Z_{1:t-1})},
\end{aligned}
$$

as well as d and e for the two terms composing (2.14). Once again, the history of poses and

measurement scans are removed from the pseudo-code for simplicity.

Algorithm 2: Inverse Sensor Model of a Scan Updated Synergistically

Function: $P(m|X,Z) = \text{InverseSensorModelSynergistic}(P(m), X, Z)$;

for $l = 1, 2, \ldots, n_l$ **do**

  Find $n_{r,l}$ cells from m corresponding to z_l;

  Define $P(\mathbf{r}_{l,0}) = 0$, $P(\bar{\mathbf{r}}_{l,0}) = 1$, $P(\mathbf{r}_{l,n_{r,l}+1}) = 1$, and $c_{n_{r,l}+1} = 0$;

  for $k = 1, 2, \ldots, n_{r,l}$ **do**

    if $k = 1$ **then**

      $a_1 = 0$, $b_1 = 1$;

    else

      $a_k = a_{k-1} + b_{k-1} p(z_l|\mathbf{r}_{l,k-1}, X) P(\mathbf{r}_{l,k-1})$, $b_k = b_{k-1} P(\bar{\mathbf{r}}_{l,k-1})$;

  for $k = n_{r,l}, n_{r,l} - 1, ..., 1$ **do**

    $c_k = \frac{P(\mathbf{r}_{l,k+1})}{P(\bar{\mathbf{r}}_{l,k})} c_{k+1} + b_k p(z_{t,l}|\mathbf{r}_{l,k+1}, X) P(\mathbf{r}_{l,k+1})$;

  for $k = 1, 2, ..., n_{r,l}$ **do**

    $\hat{P}(\mathbf{r}_{l,k}|z_{t,l}, X) = a_k + b_k p(z_{t,l}|\mathbf{r}_{l,k}, X)$, $\hat{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X,) = a_k + c_k$;

for $i \in m_{FOV}$ *(map cells inside FOV)* **do**

  Obtain the set $\mathcal{L}_i$ of l_i measurement rays intersecting this cell;

  if $l_i > 0$ **then**

    $d = P(\mathbf{m}_i) \prod_{\mathcal{L}_i} \hat{P}(\mathbf{r}_{l,k}|z_{t,l}, X)$;

    $e = P(\bar{\mathbf{m}}_i) \prod_{\mathcal{L}_i} \hat{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X)$;

    $P(\mathbf{m}_i|X,Z) = \frac{d}{d+e}$;

  else

    $P(\mathbf{m}_i|X,Z) = P(\mathbf{m}_i)$;

Return $P(m|X,Z)$;

In short, we propose two alternatives for combining inverse sensor models for multiple

rays within a scan. The ray-by-ray approach updates the entire map one ray at a time with some dependencies among the rays, while the synergistic approach considers all rays together where rays are assumed mutually independent.

2.3.3 Numerical Examples

In contrast to the current approximate inverse sensor models, the proposed ray-by-ray and synergistic algorithms evaluate the exact inverse sensor model efficiently without relying on approximations, learned solutions, or log-odds ratio assumptions. The following simulations show two examples comparing an approximate inverse sensor model with the proposed exact inverse sensor model, either with integrating multiple measurements ray-by-ray or synergistically. The proposed algorithms yield substantially more accurate maps for the same set of measurements.

Approximate Inverse Sensor Model We compare the proposed exact solution to the inverse sensor model with an approximate algorithm developed for a Microsoft Kinect sensor [48, 34], summarized as follows. The probability that the i-th grid cell $\mathbf{m}_i$ is occupied conditioned on the measurement ray $z_{t,l}$ (the l-th ray at the t-th time step) at the pose X_t is the continuous function

$$P(\mathbf{m}_i|z_{t,l}, X_t) = \begin{cases} 0.3 + (\frac{k}{\sigma\sqrt{2\pi}} + 0.2)e^{-\frac{1}{2}\left(\frac{\hat{z}_{l,i}-z_{t,l}}{\sigma}\right)^2} & \text{if } z_{t,l} \leq \hat{z}_{l,i}, \\ 0.5 + \frac{k}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{\hat{z}_{l,i}-z_{t,l}}{\sigma}\right)^2} & \text{if } z_{t,l} > \hat{z}_{l,i}, \end{cases} \tag{2.15}$$

which is based on the expected distance to the cell $\hat{z}_{l,i}$ with parameters $k = \sigma = 0.6$. This follows the structure of the approximate inverse sensor model proposed in [7]. The main idea of this approach is that the probability of a cell being occupied (i) near a measurement is high (measurement likely hits this cell), (ii) between the robot and the measurement is low (measurement passes through these cells), and (iii) beyond the measurement is unchanged (the robot cannot measure through a wall/object).

Then, these probabilities are combined in a weighted fashion such that all measurements rays of scan Z_t simultaneously update the same grid cell in a log-odds format,

$$\log\left(\frac{P(\mathbf{m}_i|Z_t, X_t)}{1-P(\mathbf{m}_i|Z_t, X_t)}\right) = \frac{1}{\sum_{z_{t,l}\in\mathbf{m}_i}\hat{z}_{l,i}}\sum_{z_{t,l}\in\mathbf{m}_i}\log\left(\frac{P(\mathbf{m}_i|z_{t,l}, X_t)}{1-P(\mathbf{m}_i|z_{t,l}, X_t)}\hat{z}_{l,i}\right). \quad (2.16)$$

Mapping a Hallway with the Ray-By-Ray Approach First, we compare the proposed exact solution to the ray-by-ray inverse sensor model summarized in Algorithm 1 with an approximations of (2.15)–(2.16). Table 2.1 shows the mapping parameters and data specifications for the actual odometry and lidar measurements from University of Pennsylvania through the Coursera open course [25]. Among these parameters, cell resolution and the number of rays have the greatest impact on computation. In general, the finer the grid, the greater the memory and computational requirements are because the computational order of (2.5) grows linearly with the number of grid cells inside the sensor range limits. Since the inverse sensor models are combined sequentially as shown in (2.8), the number of rays considered are proportional to the computation order as well. Figure 2.3 shows the direct comparison of the resulting map based on the approximate and proposed inverse sensor model. As shown in the figure, both approaches capture the structure of the environment. However, significant differences can be seen, particularly with an enlarged mapping section in Figure 2.3b.

To quantify the degree of map uncertainty, we define the entropy of the map as

$$\begin{aligned} H(P(m|X_{1:t}, Z_{1:t})) = -\sum_{i=1}^{n}\Big\{&P(\mathbf{m}_i|X_{1:t}, Z_{1:t})\log P(\mathbf{m}_i|X_{1:t}, Z_{1:t}) \\ &+ (1-P(\mathbf{m}_i|X_{1:t}, Z_{1:t}))\log(1-P(\mathbf{m}_i|X_{1:t}, Z_{1:t}))\Big\}, \end{aligned}$$

which is maximized when the probability of occupancy is 0.5 for all cells (more uncertain), and is minimized as probabilities approach either 0 or 1 (less uncertain). The entropy differences among the algorithms can be clearly depicted in Figure 2.4 at frame 2280. The

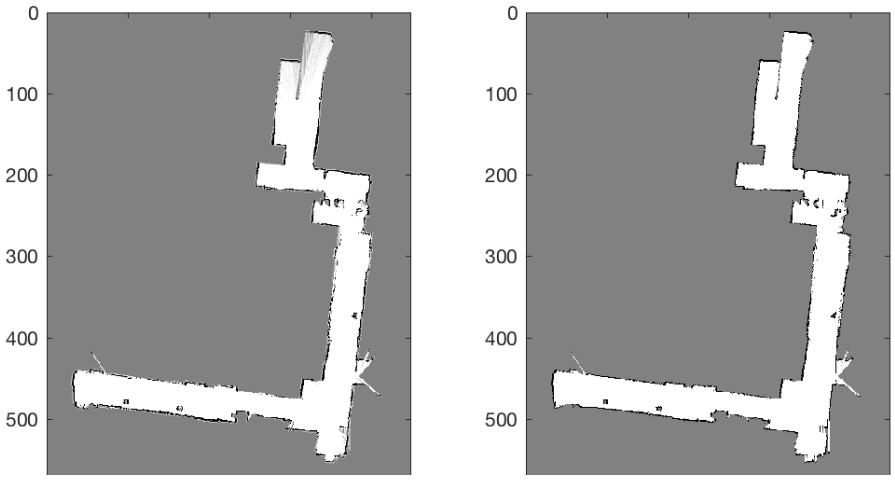
red part of the map shows the maximum entropy whereas blue shows lowest entropy for the cell. Finally, overall improvement was demonstrated in Figure 2.5 resulting in an improved entropy value per grid cell throughout the mapping process. In short, the proposed exact occupancy grid mapping produced a cleaner map with lower entropy from the same set of measurements.

Table 2.1: Experimental Parameters Provided from the Dataset

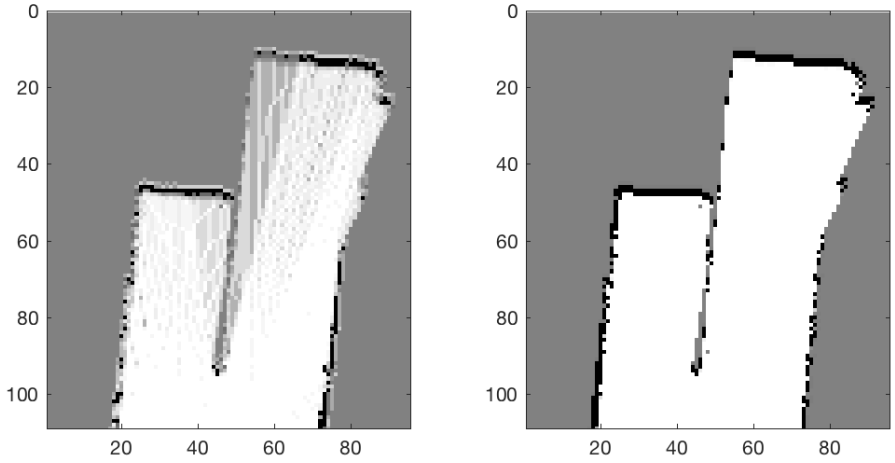
Parameters	Value
Map dimension	576x448 [pixel]
Resolution	1/16 [m]
Scan angle	[-2.36, 2.36]
Ray number	1081
Total frame	Every 120 scans out of 3701

Mapping a Room with the Synergistic Approach Next, we compare the same approximate inverse sensor model against the synergistic exact inverse sensor model. Here, the robot maps in a two-dimensional environment composed of ten wall edges, and the robot follows a figure-eight curve, then turns around and completes the same curve in the reverse direction. A pseudo-random measurement scan is sampled at each time step via the inverse transform sampling [14], and like the prior example, the same set of measurements are used with both occupancy grid mapping algorithms to construct the map. The resulting maps are illustrated in Figure 2.6 for both algorithms, where it is shown that the proposed algorithm yields a substantially more accurate and clear map with less uncertainty.

The change of the map entropy over time, and the entropy of the completed maps, for both methods are depicted in Figure 2.7. The subfigure 2.7a illustrates that the proposed exact inverse sensor model exhibits rapid decreases of entropies, and smaller entropies always. The resulting terminal map obtained from the proposed approach, shown in the subfigure 2.7b, has less uncertainty than 2.7c constructed by the approximate model. In short,



(a) Approximate (left) and proposed (right) inverse sensor model of the map

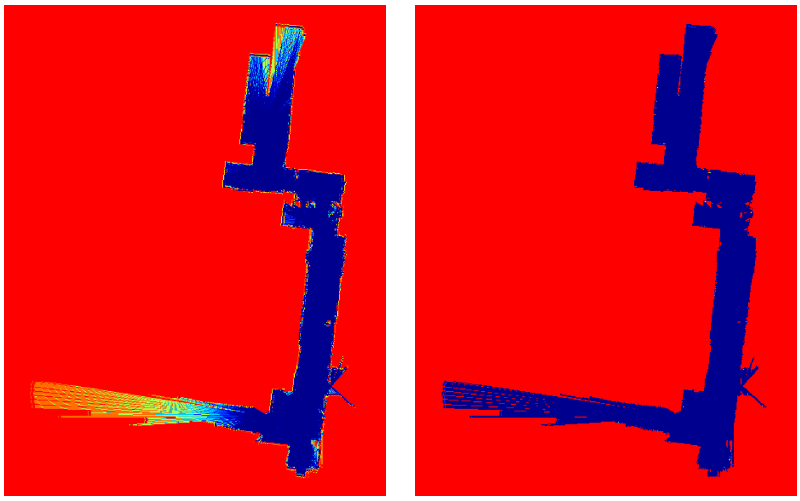


(b) Approximate (left) and proposed (right) inverse sensor model of an area

Figure 2.3: Comparison of Proposed Ray-By-Ray Approach with Approximate Solution

Using experimental sensor data, the map was generated using an approximate and the proposed exact inverse sensor models. The laser sensor and odometry data used for the experiment was provided by University of Pennsylvania open course robotics estimation and learning on Coursera [25].

the proposed approach is more efficient at extracting information about the environment from the same set of set measurements.



(a) Approx. inverse sensor model (b) Exact inverse sensor model

Figure 2.4: Map Uncertainty Comparison Between Proposed Ray-By-Ray Approach and Approximate Solution

The entropy is compared between the approximate and proposed exact inverse sensor models at step 2280. The blue areas show the lowest entropy while red shows highest.

2.4 Conclusions

In this chapter, we present the exact solution to the inverse sensor model for a single measurement ray, then extend this result to 2D occupancy grids. Measurement scans are analyzed with two techniques, namely ray-by-ray and synergistic combinations. Then, numerical results show how both proposed solutions outperform existing approximate solutions. Upon further study, the ray-by-ray approach has proven more robust against localization errors and changing environments because the rays are not assumed completely independent. Therefore, the ray-by-ray approach is applied to the subsequent proposals, numerical simulations, and experimental results.

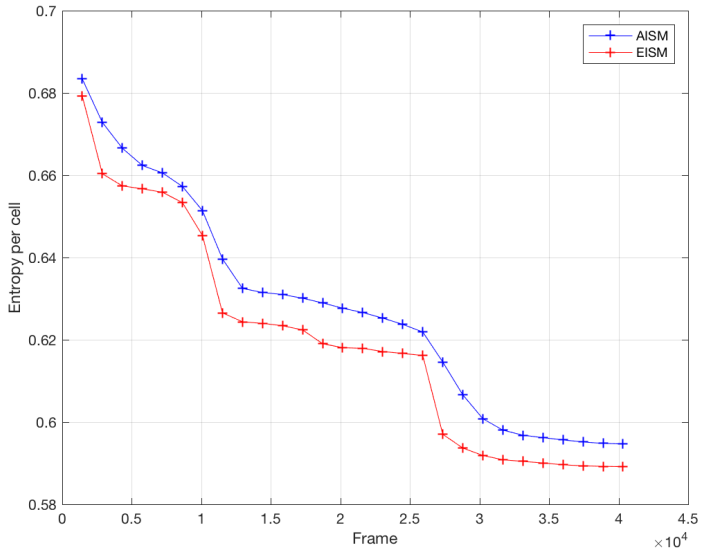
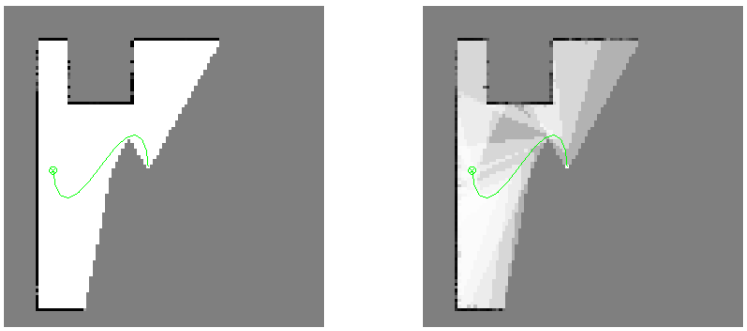


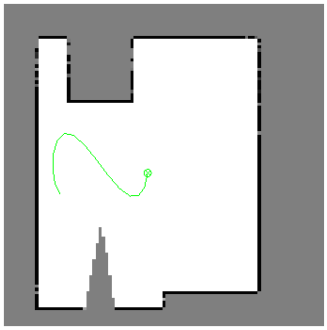
Figure 2.5: Entropy Histories of the Proposed Ray-By-Ray Approach and Approximate Solution

The map entropies of the approximate inverse sensor model (blue) and the proposed exact inverse sensor model (red) show that the exact version yields a more certain occupancy grid map.

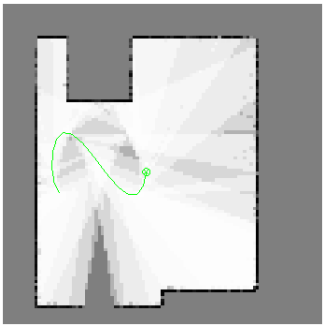


(a) Map at $t = 13$ sec (exact)

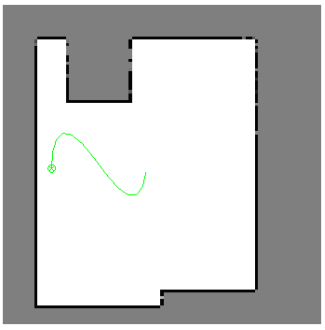
(b) Map at $t = 13$ sec (approx.)



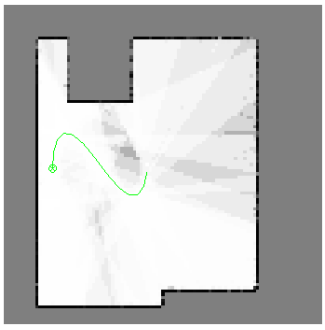
(c) Map at $t = 25$ sec (exact)



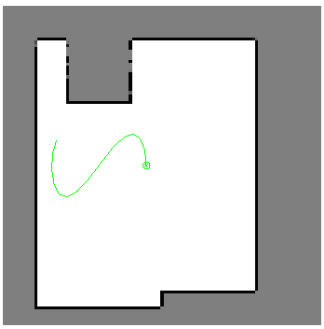
(d) Map at $t = 25$ sec (approx.)



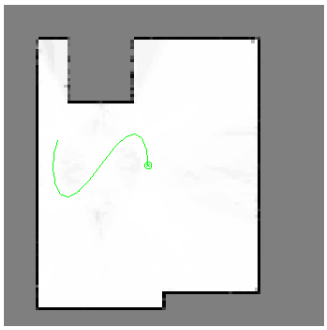
(e) Map at $t = 38$ sec (exact)



(f) Map at $t = 38$ sec (approx.)



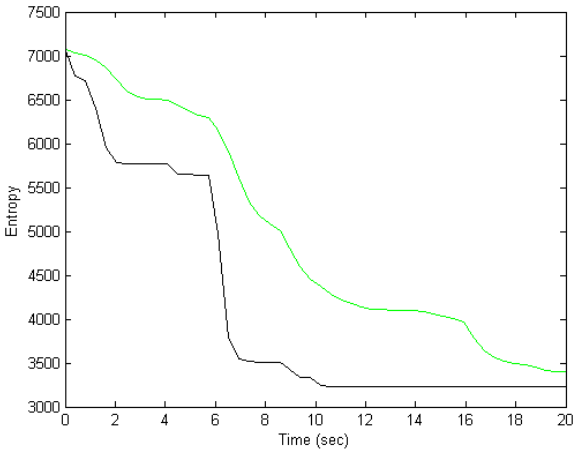
(g) Map at $t = 50$ sec (exact)



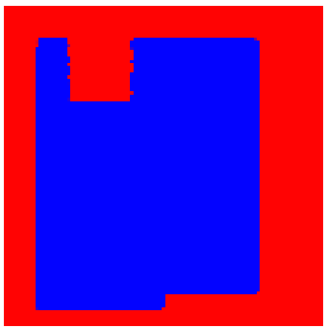
(h) Map at $t = 50$ sec (approx.)

Figure 2.6: Comparison of Proposed Synergistic Approach with Approximate Solution

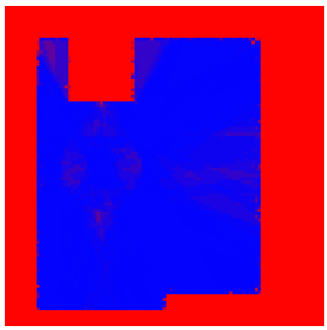
A robot (green crossed circle, green curve of its previous 13 seconds) measures a room with a Kinect depth sensor. Grid cells are either known as free (white) or occupied (black), or uncertain (gray).



(a) Entropy (black: exact model, green: approx.)



(b) Entropy map at $t = 50$ (exact)



(c) Entropy map at $t = 50$ (approx.)

Figure 2.7: Map Uncertainty Comparison Between Proposed Synergistic Approach and Approximate Solution

Entropy serves as a measure of uncertainty of the occupancy grid, where more blue regions are more certain, and red regions are more uncertain. The uncertainty is always less when the exact solution is applied.

Chapter 3: Autonomous Exploration in 2D Space

The goal of autonomous exploration is to select robotic actions designed to minimize map uncertainty, thereby maximizing map information gain. We formulate autonomous exploration as an optimization problem to determine a policy that accounts for map uncertainty and travel cost.

3.1 Entropy-Based Exploration

In this section, we define Shannon's entropy as a measure of map uncertainty, and present a novel approach to predict future entropy from an arbitrary ray [28]. Then we discuss computational modifications for real-time implementation.

3.1.1 Shannon's Entropy

Here we present how the probabilistic properties of an occupancy grid map provide key information about the uncertainty of the space for motion planning. Shannon's entropy commonly serves as an uncertainty measure [55]. Provided grid cell probabilities of map m, Shannon's entropy is defined as

$$H(P(\mathbf{m}_i)) = -P(\mathbf{m}_i)\log P(\mathbf{m}_i) - P(\bar{\mathbf{m}}_i)\log P(\bar{\mathbf{m}}_i), \tag{3.1}$$

$$H(P(m)) = \sum_{i=1}^{n_m} H(P(\mathbf{m}_i)), \tag{3.2}$$

for an individual cell and the entire map, respectively. Thus, entropy is maximized when $P(\mathbf{m}_i) = 0.5$, which corresponds to the largest uncertainty; similarly, entropy is minimized as $P(\mathbf{m}_i)$ approaches 0 or 1, which corresponds to the smallest uncertainty.

3.1.2 Expected Information Gain

Suppose a future pose candidate and its associated future measurement scan is $X_c = \{x_c, R_c\}$ and Z_c, respectively, where $c \in \mathcal{C}$ such that $\mathcal{C} = \{1, 2, ..., n_c\}$ accounts for all n_c candidates under consideration for the next robot pose. The current pose of the robot is *not* X_c in general, so any change to the probabilistic map from X_c must be predicted. Much like the probabilistic mapping, this is achieved ray-by-ray [28, 32]. Considering that all grid cell probabilities are conditioned on the history of poses $X_{1:t}$ and measurement scans $Z_{1:t}$, these terms are removed from the remaining equations of this dissertation for simplicity (excluding Appendices).

Proposition 2. *For candidate ray z_c, the expected entropy is*

$$E[H(P(m|x_c, z_c))] = \sum_{k=1}^{n_r+1} \left\{ H(P(m|x_c, z_{c,k}))P(z_{c,k}|x_c) \right\}, \tag{3.3}$$

where $z_{c,k}$ refers to the distance from x_c to the k-th grid cell along the measurement ray. The first term of the summation of (3.3), namely $H(P(m|x_c, z_{c,k}))$, is obtained with entropy definitions (3.1), (3.2) and the inverse sensor model (2.5)–(2.6). The second term is derived from (2.7) with

$$P(z_{c,k}|x_c) = \frac{p(z_{c,k}|x_c)}{\sum_{i=1}^{n_r+1} p(z_{c,i}|x_c)} = \frac{\eta_{c,k}^{-1}}{\sum_{i=1}^{n_r+1} \eta_{c,i}^{-1}}, \tag{3.4}$$

where $\eta_{c,k}$ refers to the normalizer based on the measurement $z_{c,k}$. The expected negative entropy change for candidate pose X_c is equivalently the expected information gain from ray z_c,

$$\mathcal{I}(X_c, z_c) = H(P(m)) - E\left[H(P(m|X_c, z_c))\right]. \tag{3.5}$$

Proof. See Appendix B. □

The proposed approach discretizes the measurement according to assumptions of occupancy grid mapping, and exploits the probabilistic properties uncovered by the proposed inverse sensor model. These are used to directly calculate the expected value of map entropy for a single measurement ray.

3.1.3 Computational Limitations and Approximations

The computational order for each measurement ray is $\mathcal{O}(n_r^2)$ since the summations of (3.4) are embedded in (3.3). However, several of those intersections provide negligible information since the probability of the measurement ray capturing certain cell depths is close to zero.

The approximation of expected ray entropy provides a method to reduce the computation of (3.5) substantially. This goal is achieved by systematically selecting a smaller set of grid cells to consider over the summations of (3.3) and (3.4). The smaller set is determined by the probability that each cell is captured by the measurement ray, known as the detection probability. This can be found recursively as

$$P(\mathbf{r}_{k+}) = \left\{ \prod_{j=0}^{k-1} P(\bar{\mathbf{r}}_j) \right\} P(\mathbf{r}_k), \tag{3.6}$$

which is the probability that $\mathbf{r}_k$ is the closest occupied grid cell based on past poses and measurement scans, independent of cells beyond the k-th cell from x_c. Let $\hat{n} > 0$ be a fixed number of grid cells such that $\hat{n} \leq n_r + 1$. Let $\hat{r}$ correspond to the grid cells that yield the $\hat{n}$ maximum values of (3.6) (the $\hat{n}$ most likely ray detections), indexed by increasing distance from candidate location x_c. By replacing the reduced map r with $\hat{r}$ and changing the summation limits to $\{1, 2, ..., \hat{n}\}$ in (3.3) and (3.4), the order of computation is reduced to $\mathcal{O}(\hat{n}^2)$. Even though the value of n_r is different among various measurement rays in general, $\hat{n}$ is fixed for all rays, so the computational order is fixed as well. In short, this method reduces the required computation substantially by systematically neglecting those

grid cells with little effect. It can be noted that if $\hat{n} = n_r + 1$, the ray objective function is computed without approximation.

Algorithm We present an algorithm pseudo-code providing the necessary steps to obtain the objective function for a single measurement ray (Algorithm 3). Much like the algorithm pseudo-code for the ray-by-ray inverse sensor model (Algorithm 1), the variable a_{temp} serves as an intermediate variable designed to avoid repeated calculations. Since this algorithm

operates as a function, fixed indices and condition variables are removed for simplification.

Algorithm 3: Expected Information Gain from a Measurement Ray

Function: $\mathcal{I}_{\text{ray}} = \text{RayExpInfoGain}(x, P(r), z_{1:n_r})$;

Initialize $P(\bar{\mathbf{r}}_0) = P(\hat{\bar{\mathbf{r}}}_0) = P(\bar{\mathbf{r}}_{n_r+1}) = 1$;

for $k = 1, 2, \ldots, n_r + 1$ **do**

  $P(\mathbf{r}_{k+}) = P(\bar{\mathbf{r}}_{0:k-1})P(\mathbf{r}_k)$;

  $P(\bar{\mathbf{r}}_{0:k}) = P(\bar{\mathbf{r}}_{0:k-1})(1 - P(\mathbf{r}_k))$;

Find $\hat{r} \subset r$ of the $\hat{n}$ greatest values of $\left\{P(\mathbf{r}_{1+}), P(\mathbf{r}_{2+}), \ldots, P(\mathbf{r}_{(n_r+1)+})\right\}$;

for $k = 1, 2, \ldots, \hat{n}$ **do**

  $P(\hat{\mathbf{r}}_{k+}) = P(\hat{\bar{\mathbf{r}}}_{0:k-1})P(\hat{\mathbf{r}}_k)$;

  $P(\hat{\bar{\mathbf{r}}}_{0:k}) = P(\hat{\bar{\mathbf{r}}}_{0:k-1})P(\hat{\bar{\mathbf{r}}}_k)$;

for $k_m = 1, 2, \ldots, \hat{n}$ **do**

  Initialize $\eta_{k_\text{m}}^{-1} = 0$;

  for $k_c = 1, 2, \ldots, \hat{n}$ **do**

    $a_{\text{temp}} = P(\hat{\mathbf{r}}_{k_\text{c}+})p(z_{k_\text{m}}|\hat{\mathbf{r}}_{k_\text{c}+}, x)$;

    $\tilde{P}(\hat{\mathbf{r}}_{k_\text{c}}|x, z_{k_\text{m}}) = P(\hat{\mathbf{r}}_{k_\text{c}})\eta_{k_\text{m}}^{-1} + a_{\text{temp}}$;

    $\eta_{k_\text{m}}^{-1} = \eta_{k_\text{m}}^{-1} + a_{\text{temp}}$;

  $P(\hat{\mathbf{r}}_{k_\text{c}}|x, z_{k_\text{m}}) = \eta_{k_\text{m}}\tilde{P}(\hat{\mathbf{r}}_{k_\text{c}}|x, z_{k_\text{m}})$ for all $k_\text{c} = 1, 2, \ldots, \hat{n}$;

$P(z_{k_\text{m}}|x) = \frac{\eta_{k_\text{m}}^{-1}}{\sum_{i=1}^{\hat{n}} \eta_i^{-1}}$ for $k_\text{m} = 1, 2, \ldots, \hat{n}$;

Initialize $\mathcal{I}_{\text{ray}} = 0$;

for $k_c = 1, 2, \ldots, \hat{n}$ **do**

  $\mathcal{I}_{\text{ray}} = \mathcal{I}_{\text{ray}} + H(P(\hat{\mathbf{r}}_{k_\text{c}}))$;

  for $k_m = 1, 2, \ldots, \hat{n}$ **do**

    $\mathcal{I}_{\text{ray}} = \mathcal{I}_{\text{ray}} - H(P(\hat{\mathbf{r}}_{k_\text{c}}|x_c, z_{k_\text{m}}))P(z_{k_\text{m}}|x_c)$;

Return: $\mathcal{I}_{\text{ray}}$

Numerical Justification for the Approximation The purpose of this numerical example is to provide evidence that the approximations are reasonable and increase the algorithm speed substantially. Since a measurement ray returns a range in a single direction, we only consider a 1D map where the grid cells have spacing $\alpha = 0.2$ m, and the properties of the range sensor are based on the Microsoft Kinect [48, 34] with maximum reading depth $z_{\text{max}} = 4$ m (20 grid cells inside the sensor FOV). The goal is to compare the expected entropy $\text{E}[H(P(m|x_c, z_c))]$ from (3.3) and with an approximation $\text{E}[H_{\text{approx}}(P(m|x_c, z_c))]$, which only considers $\hat{n}$ grid cells with highest detection probability (3.6).

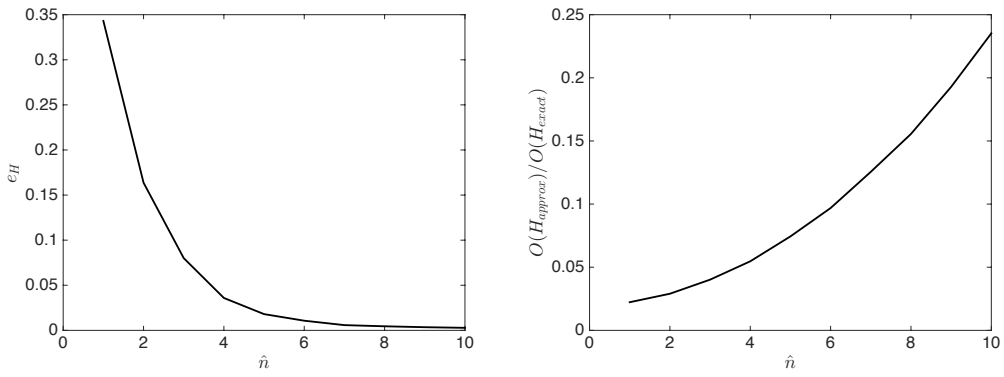
We consider 100 probabilistic maps to obtain Monte Carlo results to evaluate the approximate entropy. In every Monte Carlo trial, each grid cell has an 80% chance of being free and a 20% chance of receiving an a priori probability uniformly distributed between 0 and 1. Several metrics serve to evaluate $\text{E}[H(P(m|x_c, z_c))]$ with $\text{E}[H_{\text{approx}}(P(m|x_c, z_c))]$. The median expected entropy change is $\text{E}[H(P(m|x_c, z_c))] - H(P(m|X_{1:t}, Z_{1:t})) = -0.83792$. The error for the 100 Monte Carlo cases is defined simply as

$$e_H = \frac{1}{100} \sum_{i=1}^{100} \text{abs}\Big(\text{E}[H(P(m|x_c, z_c))] - \text{E}[H_{\text{approx}}(P(m|x_c, z_c))] \Big). \tag{3.7}$$

The Monte Carlo trials are repeated for $\hat{n} = \{1, 2, \ldots, 10\}$ and the results are plotted in Figure 3.1. This example shows a typical case when the summation limits generated from (3.6) have only small effects on (3.3), while providing very large improvements in reducing computation.

3.2 Future Pose Optimization

In this section, we show how expected entropy changes from single measurement rays can be integrated into predicting the expected information gain of measurement scans. This metric, and a cost associated with travel time, are combined to determine an optimal future pose and collision-free path in a 2D environment.



(a) Entropy Error (b) Computation Ratio

Figure 3.1: Expected Entropy Computational Improvement of Reduced Ray Cells Considered

The entropy error is decreased at the cost of increasing computation time in this Monte Carlo 1D measurement ray expected entropy case.

3.2.1 Optimal Pose from Sample Rays

The objective is to choose a future pose that minimizes the map uncertainty, such that the robot can autonomously move toward this optimal goal. The robot pose is selected among various positions and attitudes. Since searching over all locations and attitudes would require infinite computational resources, these search spaces are discretized into a limited number of robot attitudes and positions. The procedure involves three steps. First, the attitude that maximizes the information gain objective function is selected at each candidate future pose location. Second, the location with the maximum objective function of all candidates is selected. Thus, the optimal attitude and the optimal location compose the optimal future pose. Finally, the robot determines and follows a collision-free path to the optimal future pose. This process is repeated, resulting in autonomous exploration.

Attitude Optimization Consider a collision-free pose candidate at an arbitrary location x_c. At this pose, consider n_d evenly-spaced measurement rays oriented radially outward from the robot in a circular pattern (see red lines in Figure 3.2). These rays are denoted $\{z_{c,1}, z_{c,2}, \ldots, z_{c,n_d}\}$. Attitudes correspond to the rays such that the robot sensor is aligned

with the ray direction, and these are denoted $\{R_{c,1}, R_{c,2}, \ldots, R_{c,n_d}\}$, where the scan with attitude $R_{c,d}$ might cover several ray directions depending on the sensor FOV. We choose optimal attitude R_c^* as the summation of the expected entropy changes covered by the scan,

$$R_c^* = \underset{R_{c,d}}{\operatorname{argmax}} \sum_{z_{c,i} \in R_{c,d} \text{ FOV}} \Big(H(P(m)) - \mathrm{E}[H(P(m|x_c, z_{c,i}))] \Big). \tag{3.8}$$

This method provides the attitude that maximizes the information gain at an arbitrary location in 2D space. The set of candidate positions that warrant consideration is determined with one of two methods, namely *expanding ring* and *complete Cartesian*, described next.

Expanding Ring The expanding ring technique is advantageous for searching local solutions quickly, only considering distant future poses when necessary. The key idea is that future candidate locations lie on a circular "ring" centered around the robot, evenly spaced around the ring (see red circles in Figure 3.2), and the ring is expanded if certain criteria are not met. More explicitly, consider n_c candidate locations denoted by $x_c \in \{x_1, x_2, \ldots, x_{n_c}\}$ located with the distance δ away from the robot. All candidates must satisfy the inequality constraint,

$$P_{\text{collision}}(X) = 1 - \prod_{i \in \mathcal{C}_X} P(\bar{\mathbf{m}}_i) \leq \beta, \tag{3.9}$$

where $\mathcal{C}_X$ is the set of grid cells falling inside a volume of preselected size around X that may cause collision and $\beta > 0$ is a small acceptable probability of collision. Any location that violates an inequality constraint (3.9) is excluded to avoid collisions. The set of optimal attitudes at each candidate location is $\left\{R_1^*, R_2^*, \ldots, R_{n_c}^*\right\}$, obtained from (3.8). The information gain objective function for a scan Z_c captured from pose X_c is

$$\mathcal{I}(X_c) = H(P(m)) - \mathrm{E}\left[H(P(m|X_c, Z_c))\right], \tag{3.10}$$

and this is computed by a summation about the n_d rays as

$$\mathcal{I}(x_c, R_c^*) \approx \sum_{z_{c,i} \in R_c^* \text{ FOV}} \Big(H(P(m)) - \mathrm{E}\left[H(P(m|x_c, z_{c,i}))\right] \Big), \tag{3.11}$$

$$x_c^* = \underset{x_c}{\operatorname{argmax}}\, \mathcal{I}(x_c, R_c^*). \tag{3.12}$$

At the optimal pose $X_c^* = (x_c^*, R_c^*)$, the resulting information gain must satisfy $\mathcal{I}(X_c^*) \geq \mathcal{I}_{\text{min}}$, where $\mathcal{I}_{\text{min}}$ is a minimum threshold for expected information gain; robot motion is only justified when the expected information gain is significantly large. If the minimum threshold is not met, the ring of candidate pose locations is increased such that n_c and δ are multiplied by a scale function $\lambda > 1$. This process is repeated until the expected information gain of the optimal pose is at least $\mathcal{I}_{\text{min}}$, or all candidates lie in collision zones or outside map limits.

The primary advantage of the expanding ring approach to search the 2D space is that the number of pose candidates need not be proportional to the map area, and that local maxima tend to fall within short distances of the current robot pose. Additionally, the distance to these poses need not be explicitly considered in the pose determination optimization because the distance costs are identical to each candidate, assuming objects do not occlude the trajectories. The main drawback is that poses outside a small local region of the robot are frequently neglected, causing the robot to repeatedly execute short motions, often without large information gains.

Complete Cartesian The complete Cartesian method to search the 2D space provides candidates located fixed distance d apart in each Cartesian direction. This approach is advantageous for capturing expected information gains in regions outside of the immediate vicinity of the robot, but must be applied carefully to avoid issues with computational bottlenecks due to the potentially-large search space.

Similar to the expanding ring approach, n_c candidate pose locations are considered, where those violating (3.9) are neglected from further consideration. However, unlike the

ring expanding approach, the candidates have different distances from the robot in general, so the objective function is modified to enforce a cost on the squared distance from the current pose location x_t to the candidate location x_c, i.e.,

$$\mathcal{I}(x_c, R_c^*, x_t) \approx \sum_{z_{c,i} \in R_c^* \text{ FOV}} \Big(H(P(m)) - \mathrm{E}\left[H(P(m|x_c, z_{c,i}))\right] \Big) - k_{\text{dist}} \left\| x_t - x_c \right\|^2, \tag{3.13}$$

where weighting function k_{dist} represents the sensitivity to avoiding large motions across the map. Furthermore, the same candidate locations are considered throughout the repeated processes of exploration, and the map cell occupancies are assumed static. Therefore, if candidate location x_c fails to satisfy $\mathcal{I}(x_c, R_c^*, x_t) + k_{\text{dist}} \left\| x_t - x_c \right\|^2 \geq \mathcal{I}_{\text{min}}$, then x_c need *never* be considered again. Avoiding these unnecessary calculations yields a greatly lowered computational burden, particularly late in exploration where several regions are collision-free but disadvantageous to revisit.

3.2.2 Collision-Free Motion Planning

There are two important steps to moving a robot on a collision-free trajectory to its optimal pose. First, Dijkstra's algorithm [16, 19] provides the cost map and optimal collision-free waypoints. Then, a constrained polynomial least squares trajectory serves as the desired path, which is followed by a controller.

Dijkstra's Search Dijkstra's search is naturally applied to this problem because the occupancy grid map serves nicely as a graph, and Dijkstra's algorithm provides an optimal collision-free trajectory along that graph. There are two steps to Dijkstra's algorithm: first, generate a cost map from the robot location to each grid cell on the 2D occupancy grid map. Only cells that satisfy (3.9) are considered to avoid possible collisions. Traveling to a neighboring cells that shares an edge costs the edge length distance α, and the cost to travel to cells that share a corner is $\sqrt{2}\alpha$. This provides the collision-free distances to all reachable

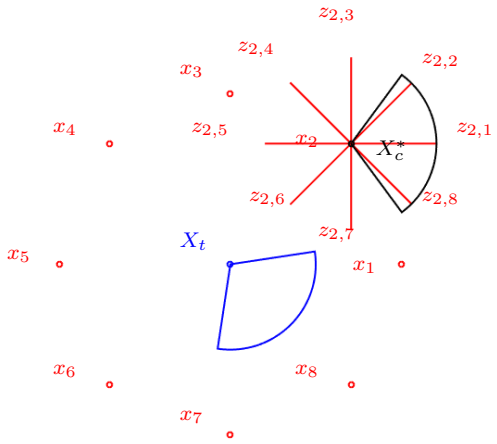


Figure 3.2: Optimal Pose Selection Illustration

This figure illustrates the expanding ring method. Initially, a robot (blue circle) views down and left (blue sector). Then, the robot considers $n_c = 8$ poses (red circles) as possible candidates for where to move next. For each candidate, $n_d = 8$ directions are considered (red lines, only displayed on candidate location $c = 2$). Then, the expected entropy of each ray is calculated. The scan (red sector) covering those rays with the largest expected entropy decrease is chosen for each candidate, and the best candidate X_c^* (black circle: location, black sector: scan) is chosen to maximize information gain. Finally, Dijkstra's algorithm provides the collision-free motion between X_t and X_c^*, and the process is repeated.

locations on the map. Second, the waypoints from a candidate pose to the current robot pose are easily obtained along the cost map using steepest descent.

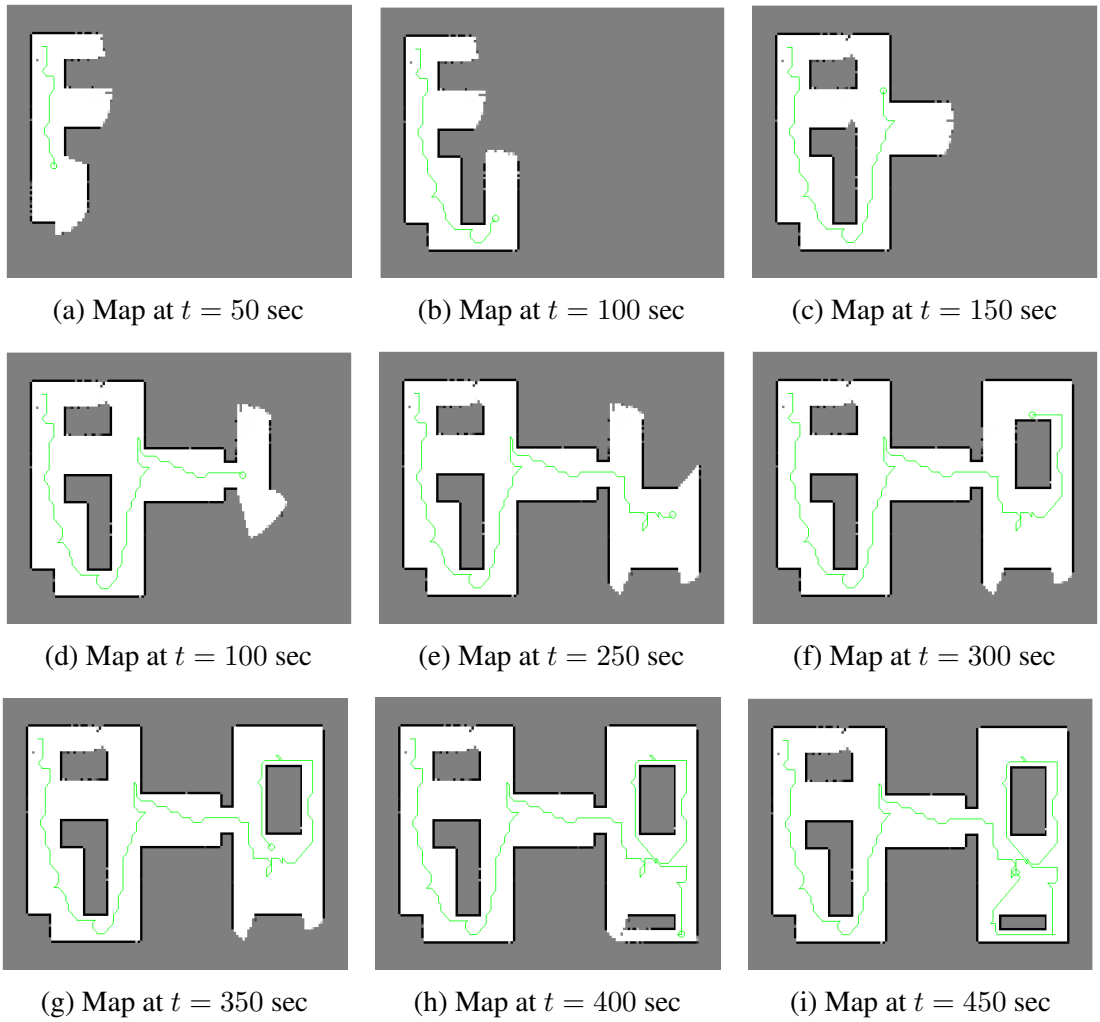
Constrained Polynomial Least Squares Trajectory The trajectory to follow the path outlined by Dijkstra's algorithm is determined with a linear constrained least squares optimization, assuming the robot follows a polynomial trajectory with fixed speed. The starting and ending positions and attitudes may be constrained, and polynomials patched together for long trajectories share a common position and velocity with respect to time. Then, a controller on the robot tracks this trajectory until the robot falls within acceptable thresholds of the final optimal pose. Any control scheme for the motion of the robot can be integrated with the proposed exploration algorithm. Once the robot completes this motion, the entire process is repeated.

3.3 Numerical Examples

Here we present two numerical examples. The first uses the expanding-ring technique in a small environment and the second considers a larger benchmark environment, so the complete Cartesian method is applied, as described in Section 3.2.1.

3.3.1 Exploring a Simple Environment

Here, a robot autonomously maps and explores a 2D environment composed of two rooms and one hallway. The robot models its surroundings with an occupancy grid of $15,000$ cells where grid cell edges have length $\alpha = 0.2$ m, composing a map with dimensions 30 m $\times$ 20 m. The initial probability $P(\mathbf{m}_i) = 1 \times 10^{-10} \approx 0$ (minimum value for free space) for grid cells covered by the circular robot of radius 0.1 m and $P(\mathbf{m}_i) = 0.5$ for all other cells. At each time step, the robot receives a measurement scan, where the probabilistic properties of the sensor are taken from [48, 34]. Then, the $n_c = 8$ evenly-spaced candidate locations about a circle of radius $\delta = 0.5$ m around the current pose location are considered,



(a) Map at $t = 50$ sec (b) Map at $t = 100$ sec (c) Map at $t = 150$ sec

(d) Map at $t = 100$ sec (e) Map at $t = 250$ sec (f) Map at $t = 300$ sec

(g) Map at $t = 350$ sec (h) Map at $t = 400$ sec (i) Map at $t = 450$ sec

Figure 3.3: Autonomous Exploration of a 2D Space Using the Expanding Ring Approach

A robot (green circle, green marker of previous path) measures a room with a Kinect depth sensor to maximize map information gain.

where $n_d = 32$ measurement rays are evenly-spaced about the candidate location. When no current candidates yield expected information gains above $\mathcal{I}_{\text{min}} = 2$, $\lambda = 1.25$ is multiplied to n_c and δ. The motion of the robot is restricted to movement along cells satisfying (3.9) with $\beta = 0.01$. Dijkstra's algorithm provides collision-free motion planning. The results are illustrated in Figure 3.3.

Knowing only that the robot is inside free space at the beginning, the robot carefully navigates the environment while avoiding collisions. The robot motion is governed by a

policy that maximizes the map information gain within its set of pose candidates, where the $\hat{n} = 6$ is chosen to approximate the expected entropy of each ray. When running the exploration algorithm within the framework of the Robot Operating System (ROS) [2], the mean computation time is 0.0194 seconds to determine the optimal future pose and complete Dijkstra's algorithm on the occupancy grid. The computation times for this map reached roughly 1 second at maximum, corresponding to rare cases when the robot is locally surrounded by a highly-certain environment; this task requires evaluating many more pose candidates and motion planning over a larger terrain.

Throughout the numerical example, the robot chooses numerous actions, based directly on their expected information gains, not frontiers or predicted measurement scans. If the obstacles were known a priori, the motion planning problem would yield a simpler path; however, the autonomous exploration is based on only the information of the map that the robot generates, so the motion planning must be reevaluated repeatedly. Even with these limitations, the robot explores the vast majority of reachable space in the 450 second period.

3.3.2 Exploring a Complicated Benchmark Environment

Next, the proposed autonomous exploration approach is applied to the floor plan of the Intel Research Lab, illustrated in Figure 3.4. The robot explores its surroundings with an occupancy grid with $90,000$ cells where grid cell edges are $\alpha = 0.1$ m, composing a map with dimensions 30 m $\times$ 30 m. Similar to the prior example, the initial probability $P(\mathbf{m}_i) = 1 \times 10^{-10} \approx 0$ (minimum value for free space) is chosen for grid cells covered by the circular robot of radius of 0.3 m and $P(\mathbf{m}_i) = 0.5$ for all other cells. For added safety, cells falling within 0.6 m of the robot are considered inside the possible collision zone $\mathcal{C}_X$, from (3.9), where $\beta = 0.1$ is selected.

The robot follows the complete Cartesian candidate search method to determine future poses, with $k_{\text{dist}} = 5$ to avoid large motions with little added expected information gains. Due to the large size of the map, candidates below the expected information gain of 1.25 are

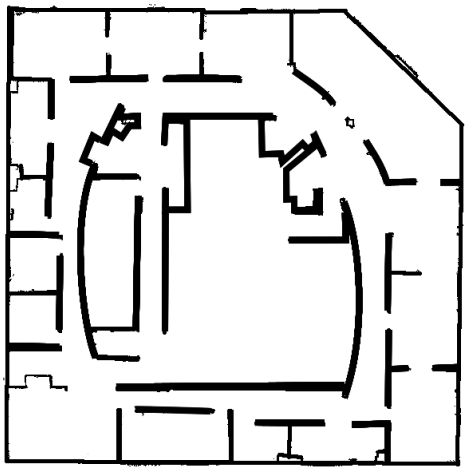


Figure 3.4: Intel Research Lab Floor Plan Benchmark

The 2D floor plan from the Intel Research Lab [39] is simplified to eliminate small objects. The remaining black pixels are used to produce walls for 2D simulations.

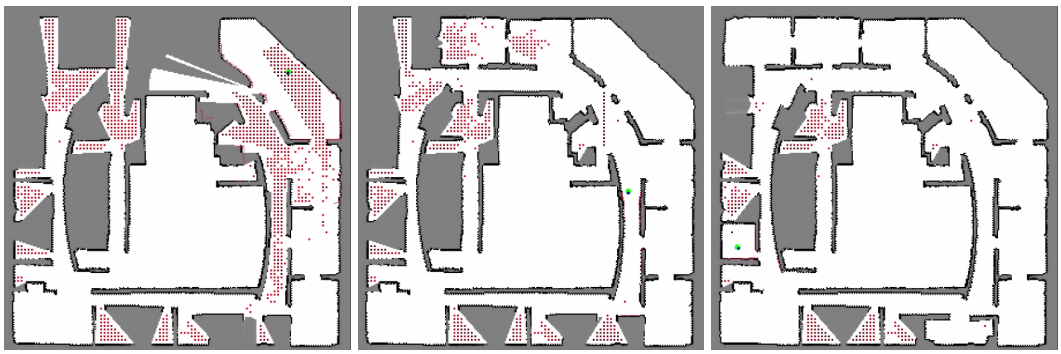
neglected from future consideration.

The large 2D exploration is simulated in the ROS Stage 2D environment. The corresponding simulation results are illustrated in Figure 3.5 and a video is available at https://www.youtube.com/watch?v=5VdzKHreB_s. Lighter red dots and darker blue dots in the explored area correlate to the level of information gain at those positions with low and high value, respectively. It is shown that the proposed autonomous exploration algorithm successfully mapped the complicated environment composed of a number of rooms, narrow hallways, and open spaces that are irregularly shaped.

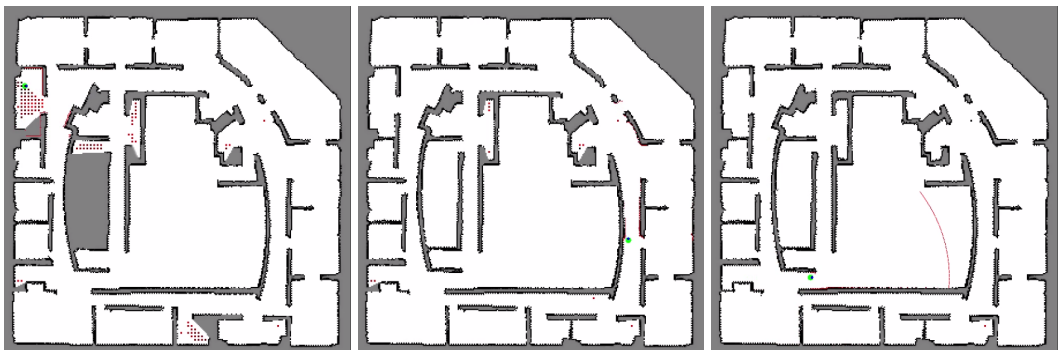
3.4 Conclusions

The proposed autonomous exploration scheme determines the expected value of entropy for various potential robotic actions. The optimal policy maximizes an objective function that includes expected information gain and travel distance. Collision-free paths to optimal poses are determined with Dijkstra's search and a constrained polynomial least squares trajectory. These are demonstrated with two numerical examples, showing the expanding ring and complete Cartesian approaches. Due to superior scalability, the complete Cartesian

(a) $t = 0$ min (b) $t = 5$ min (c) $t = 10$ min



(d) $t = 15$ min (e) $t = 20$ min (f) $t = 25$ min



(g) $t = 30$ min (h) $t = 35$ min (i) $t = 40$ min

Figure 3.5: Autonomous Exploration of the Intel Research Lab Using the Complete Cartesian Approach

A robot (green) measures a room in the ROS Stage simulator using the Intel Research Lab floor plan. The exact inverse sensor model is used for the autonomous exploration algorithm.

approach is applied to subsequent exploration algorithms in this dissertation.

Chapter 4: Autonomous Exploration in Vertically-Uniform 3D Space

In this chapter, we introduce mapping and autonomous exploration in 3D environments. Here, we consider common spaces like rooms and hallways where objects are largely vertically-uniform. A complete 3D probabilistic map is projected onto a simpler 2D map at a fixed exploration height, where the projected map can be used for collision-avoidance and entropy predictions.

4.1 Occupancy Grid Mapping in 3D

The inverse sensor model from Chapter 2 is based on an arbitrary vector spanning 2D space. The distances from the robot to grid cells are obtained geometrically through ray casting. This method is easily extended to 3D space, and the inverse sensor model is applied identically. Since the map is updated ray-by-ray (each measurement ray is considered once by itself), this approach is extended for measurement fusion of any number of sensors; the map is updated by each ray of a scan sequentially, then the map is updated by the rays of the next scan in queue, which might have different characteristics. There are several reasons why a robot might be equipped with multiple sensors, e.g., different fields-of-view or depth sensor properties. For example, a robot may be equipped with a 2D LIDAR with highly accurate measurements, but a limited single-plane field-of-view. Then, the robot can compensate for the LIDAR limitations with an IR depth sensor that provides many more measurement rays with a 3D scan, where each ray has lower accuracy than the LIDAR rays. Since each sensor model is explicitly considered in the cell probability calculations, measurements from both sensors may serve to update the same probabilistic map. However, the impact from a LIDAR measurement with high accuracy on the probabilistic map will be greater than that of the IR depth sensor with lower accuracy. Since sensor properties are considered explicitly, the proposed 3D mapping approach can fuse the impact of measurements on the

probabilistic map from different sources with different properties.

However, implementation in 3D requires several careful considerations. First, the number of cell occupancy probabilities tracked with computer memory increases by a factor of the number of cells in the added dimension. In its current form, the proposed method assumes fixed map limits, so these should be carefully selected with grid cell resolution based on available memory. Second, measurement rays spanning 3D space typically involve more intersections with grid cells. The computational orders of ray casting and the inverse sensor model are proportional to the number of grid cells along the ray n_r, so careful consideration must be placed on sensor limits and rays per scan that can be considered. Typically, computers small enough to be carried onboard flying robots lack the processing capabilities to update massive maps quickly, particularly because the ray-by-ray updating method requires many computations in series with large measurement scans. Therefore, an onboard computer can simply stream the sensor data, and a more powerful computer can run the mapping.

Additionally, mobile robots, particularly aerial vehicles, are frequently subject to fast movements, and the time stamps for pose estimates and depth sensor readings are not guaranteed to align. For example, the most recent pose estimate and measurement scan may have significantly different time stamps while the robot is rapidly rotating. Using these together violates the assumption that the robot pose is well-known for every measurement scan, i.e., X_t and Z_t are given. Therefore, pose estimates and sensor readings must be paired according to their time stamps. A useful tool that easily satisfies this goal is the Message Filters package with the Robot Operating System (ROS), where stamped poses and measurement scans are synchronized using recently-received data.

4.2 Exploration in Vertically-Uniform Environments

Next, we describe two approaches to projecting the 3D probabilistic map onto a 2D map at the exploration height [33]. Assuming projections are sufficient is consistent with rooms

and hallways with non-complex geometries.

4.2.1 Collision and Entropy Maps

Occupancy grid map cell volume is frequently smaller than the size of the robot taking measurements, particularly with highly-detailed maps. This motivates a method to simply represent locations that risk collision, referred to as a *collision map* C. Let $k_C \geq 1$ be an integer representing the minimum number of 3D grid cells that form a cube that can completely encompass the robot, i.e., the collision map cell edge length is

$$\alpha_C = k_C \alpha. \tag{4.1}$$

Consider the set $m_{C,k} \subset m$ consisting of the 3D grid cells falling inside the limits of the k-th cell of collision map C, namely $\mathbf{C}_k$. Then the collision probability is

$$P(\mathbf{C}_k) = \max\left(P(\mathbf{m}_i)\ \forall\ i \in m_{C,k}\right). \tag{4.2}$$

Then, the robot is only allowed to consider moving into k-th cell of the collision map if

$$P(\mathbf{C}_k) \leq C_{\text{thresh}}, \tag{4.3}$$

where $C_{\text{thresh}} > 0$ is a maximum acceptable collision threshold, typically chosen slightly greater than 0. In short, this conservative approach limits robotic motion to regions with a low risk of collision while neglecting objects far above or below the exploration plane.

In a similar method, *entropy map* E determines cells based on the entropy metric of (3.1) over a different subset of cells. Let the E be a 2D map composed of cells with edge length α located at the exploration height. Let the set $m_{E,k} \subset m$ be the cells directly above and below the k-th cell of E, neglecting any 3D cells located lower than the floor height. Selecting this set is advantageous because measurements are assumed not to penetrate occupied cells

(edge length α) and measurement rays are typically not closely-aligned with the third axis of the inertial frame (vertically long set). Furthermore, any cells located below the floor, which cannot be reached, should not impact the uncertainty of the space. Then, the occupancy probability of the k-th entropy map cell is selected as

$$P(\mathbf{E}_k) = \underset{P(\mathbf{m}_i)}{\operatorname{argmax}}(H(P(\mathbf{m}_i)) \; \forall \; i \in m_{E,k}). \tag{4.4}$$

After several measurements of the cells belonging to $m_{E,k}$, it is common for $H(P(\mathbf{E}_k)) \approx 0$, which implies that these cells are well-known. However, this does not imply that the captured space is *entirely* free or occupied (e.g. an object on the ground that does not breach the upper map boundary), so applying (4.4) alone could arbitrarily set $P(\mathbf{E}_k)$ close to 0 or 1. This can have damaging effects on the expected information of cells beyond the k-th cell. Let $P_{\text{min}} > 0$ be a lower bound such that $H(P_{\text{min}}) = H(1 - P_{\text{min}}) \approx 0$. If $H(P(\mathbf{E}_k)) \leq H(P_{\text{min}})$, then this probability is corrected to

$$P(\mathbf{E}_k) = \begin{cases} P_{\text{min}}, & \text{if mean}(P(\mathbf{m}_i) \; \forall \; i \in m_{E,k}) \leq 0.5 \\ 1 - P_{\text{min}}, & \text{otherwise} \end{cases}, \tag{4.5}$$

where the first case corresponds to mostly open space and the second case corresponds to mostly occupied space (e.g. walls and objects). In short, the candidates are only considered if they can be reached over collision map C via Dijkstra's search, and the information gains of the reachable candidates are predicted using entropy map E.

4.2.2 Collision and Entropy Combination Map

The second proposed technique combines the advantages of both the collision and entropy maps into a 2D projected occupancy grid map, which can be used for both tasks. Let the *collision and entropy combination map* m_{2D} be a 2D projected map located at the exploration height with grid cell edges the same as the collision map, namely α_C from (4.1), using the

same subset of grid cells from the collision map, namely m_C. Considering that the cells begin with uniform probability P_{init} where $0 < P_{\text{init}} < 1$, define a threshold P_{thresh} such that $P_{\text{init}} < P_{\text{thresh}} < 1$. Then, the probability assigned to the k-th grid cell of m_{2D}, namely $\mathbf{m}_{\text{2D},k}$ is

$$P(\mathbf{m}_{\text{2D},k}) = \begin{cases} P(\mathbf{C}_k), & \text{if } P(\mathbf{C}_k) \geq P_{\text{thresh}} \\ \min\left(P(\mathbf{m}_i)\ \forall\ i \in m_{C,k}\right), & \text{otherwise} \end{cases}. \tag{4.6}$$

This approach has several advantages. First, measurements must enter the vicinity of the k-th cell for $P(\mathbf{m}_{\text{2D},k})$ to change from P_{init}. Without prior measurements, capturing $P(\mathbf{m}_{\text{2D},k})$ produces a large reward for the expected information gain in (3.10). Second, the value of P_{thresh} can be modified based on how conservative the collision risk assessment for a robot should be. Additionally, when $P(\mathbf{m}_{\text{2D},k}) < P_{\text{thresh}}$, this event tends to favor free cells such that expected information gains of cells beyond the k-th cell have greater impact on exploration. Furthermore, a single map is simpler, and since cell sizes satisfy $\alpha_C \geq \alpha$, it follows that n_r can only be decreased with α_C, thereby increasing computational speed.

4.3 Numerical Example

The following numerical example involves a simulated quadrotor taking measurements of a large room and generating a 3D occupancy grid map. Separate collision and entropy map projections are used for autonomous exploration.

4.3.1 Software Structure

The Robot Operating System (ROS) provides an excellent framework for several programs, referred to as *nodes*, to communicate easily. These are mostly coded in C++ for speed, except for a few Python scripts for a GUI and trivial message conversions, and launch/yaml scripts for running nodes and setting parameters.

Gazebo is a 3D dynamic simulator [35], with plugins to simulate an Asus Xtion color depth sensor and a Hokuyo LIDAR. Gazebo also provides position and attitude transformations from the world to the quadrotor body, and the ability to set robot model states directly.

Using fixed transformations between the quadrotor body and its onboard sensors, an original mapping node uses a variation of ROS message filters, known as transform message filters, to synchronize the sensor poses with sensor readings. These serve as inputs for ray casting in 3D, which provides the cell probabilities and depths along the ray through geometry. Then, the inverse sensor model updates the map ray-by-ray. The node is designed to work for any number of sensors, specified by sensor parameters in a yaml configuration file.

There are two processes for exploration. The first process is projecting the 3D map onto 2D maps; the same code under different conditional parameters produces two nodes, for collision and entropy maps. The second process subscribes to these two 2D maps; the collision map serves as input for candidate pose consideration and Dijkstra's search, and the entropy map serves to predict information gain. This node provides path messages, composed of desired poses 0.01 sec apart, displayed with a ROS visualization package, Rviz.

Finally, a path-interpreter node subscribes to the path messages and interpolates the pose at the current ROS time step. This pose is converted into a message that sets the Gazebo quadrotor state.

4.3.2 Simulation Results

The 3D mapping and exploration algorithms are simulated with several parameters, described next.

Maps A cell size of $\alpha = 0.075$ m is selected and the map limits are -4 m to 4 m in the x-direction, -6 m to 6 m in the y-direction, and -0.15 m to 1.5 m in the z-direction. The initial probability of cells is selected at $P_{\text{init}} = 0.1$. For the collision map, the cell edge factor is $k_C = 3$. Additionally, Dijkstra's algorithm produces a cost map composed of distance costs for each collision map cell, denoted d_{cell}, from the current robot position to the candidate pose locations over a collision-free path about the occupancy grid.

Sensors Next we describe the key parameters used for mapping and exploration. The inverse sensor model (2.5)–(2.6) relies on the stochastic properties of two sensors, namely an Asus Xtion color IR depth sensor and a Hokuyo LIDAR. The Xtion and Hokuyo sensors are modeled using a modified version of the beam model for range finders [59]. This includes uniform distribution for a phantom reading with probability $P_{\text{rand}} = 0.1$ and a Gaussian distribution with probability $P_{\text{hit}} = 0.9$. The Gaussian distribution corresponds to the desired case when the measurement z correctly captures a particular cell at expected distance $\hat{z}$ with standard deviation σ. The total forward sensor model for the Xtion and Hokuyo is

$$p_{\text{total}} = P_{\text{rand}} p_{\text{rand}} + P_{\text{hit}} p_{\text{hit}}, \tag{4.7}$$

$$p_{\text{rand}}(z, z_{\text{min}}, z_{\text{max}}) = \frac{1}{z_{\text{max}} - z_{\text{min}}}, \tag{4.8}$$

$$p_{\text{hit}}(z, \hat{z}, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ \frac{(\hat{z} - z)^2}{2\sigma^2} \right\}, \tag{4.9}$$

such that $z_{\text{min}} = 0.5$ m and $z_{\text{max}} = 4.0$ m for both the Xtion and Hokuyo, and the standard deviations are $\sigma_{\text{Xtion}} = 0.25$ m and $\sigma_{\text{Hokuyo}} = 0.1$ m, respectively, which exceed their specified values to account for localization uncertainty and sensor distortion of the Xtion. These sensors update the probabilities of cubic 3D cells with edge length $\alpha = 0.075$ m and initial probability $P_{\text{init}} = 0.1$.

Bump Function Dijkstra's algorithm produces a cost map composed of distance costs for each collision map cell, denoted d_{cell}, from the current robot position to the candidate pose

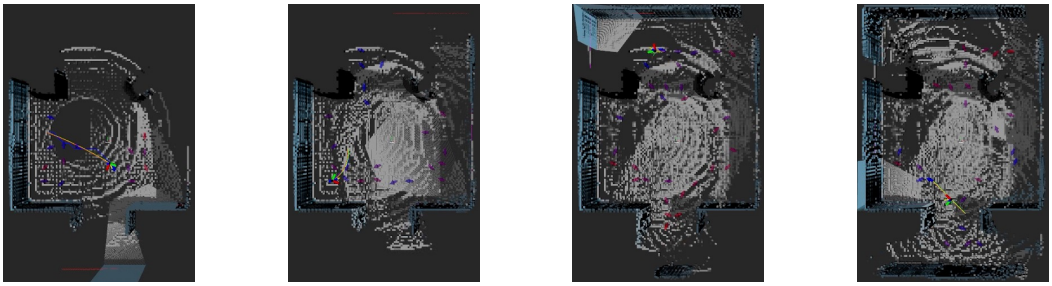
locations over a collision-free path about the occupancy grid. This cost map provides a more accurate travel distance cost than the Euclidean distance used in (3.13) because the cost map can account for obstacles within the environment, which is particularly important to properly analyze long trajectories. Therefore, we replace $k_{\text{dist}} \|x_t - x_c\|^2$ from (3.13) with a normalized variation on the *bump function* [26] with maximum distance $d_{\text{max}} > 0$,

$$\text{bump}(d) = \begin{cases} \exp\left\{1 - \frac{1}{1-(d/d_{\text{max}})^2}\right\}, & \text{if } 0 < d < d_{\text{max}} \\ 0, & \text{otherwise} \end{cases}, \tag{4.10}$$

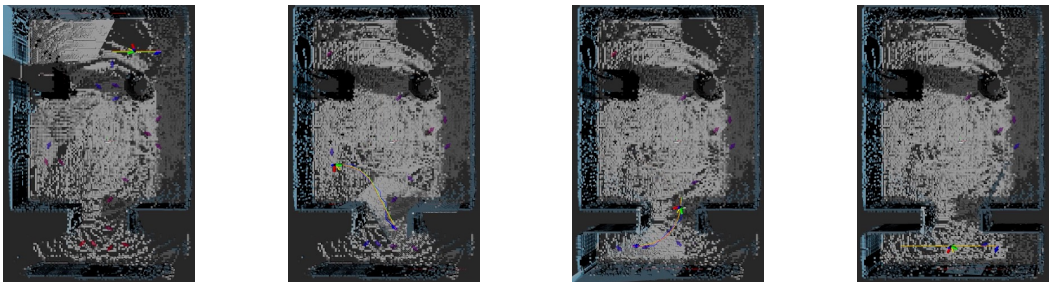
such that d_{max} is selected as the maximum cost map value. Then, bump(d) is multiplied to $\sum_{z_{c,i} \in R_c^* \text{ FOV}} \Big(H(P(m)) - \text{E}\left[H(P(m|x_c, z_{c,i}))\right] \Big)$ from (3.13) when finding optimal poses. The bump function prioritizes actions that improve the knowledge of the local surrounding space before distant traversals across the map.

Simulation In the simulation, the robot explores the Gazebo environment for 8 min while taking measurements. The 3D occupancy grid map and true environment are shown in Figure 5.3. The Gazebo simulated environment and two 2D projected maps for collision and entropy are shown in Figure 5.4. The total map entropy as a function of time is shown in Figure 5.5.

The two-map approach both exhibits advantages and disadvantages. The exploration algorithm successfully avoided collisions and gave way to a fairly complete 3D occupancy grid. The floor, which is not always visible, is properly estimated in most places. Conversely, this attention to detail has negative consequences that encourage the robot to repeatedly look at the same regions from different vantage points. Since an entropy map cell is composed from many 3D grid cells single-stacked vertically, if just one of these 3D map cells is not captured or hardly captured, the entropy map cell will be recorded as quite uncertain, shown covering numerous locations in Figure 4.2c. The robot may attempt to measure this space again, even when other regions might more strongly warrant observation. These periods



(a) $t = 1$ min (b) $t = 2$ min (c) $t = 3$ min (d) $t = 4$ min



(e) $t = 5$ min (f) $t = 6$ min (g) $t = 7$ min (h) $t = 8$ min

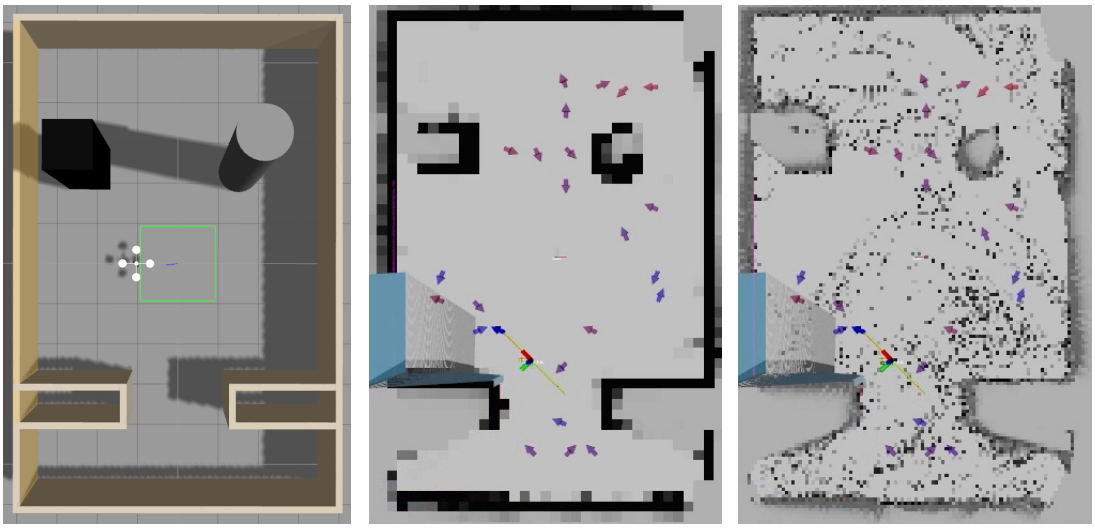
Figure 4.1: Simulated 3D Occupancy Grid Map

A robot (RGB axes) explores a simulated 3D Gazebo environment using separate collision and entropy projected maps. The robot considers numerous candidates (arrows, blue: greater objective function, red: smaller objective function) and selects the optimal candidate (blue arrow).

of sluggish entropy decrease are shown with flatter sections of Figure 5.5. In short, this algorithm successfully explores the space collision-free while producing a complete map, but over-attention to just a few uncertain cells has negative consequences on the exploration speed.

4.4 Conclusions

In this chapter, we proposed an extension of exact probabilistic occupancy grid mapping from 2D to 3D, then proposed a simplified autonomous exploration scheme for vertically-uniform environments. Two versions of the projected map, namely one for collision and one for entropy, are used in a numerical example. The collision and entropy combination map is applied to an experimental result in Chapter 7.



(a) Environment (b) Collision Map (c) Entropy Map

Figure 4.2: Simulated Environment and 2D Projected Maps at 4 min

The robot hovering near the beginning of the simulation is shown in (a). The projected maps for collision (b) and entropy (c) are shown 4 minutes into the simulation. The robot and candidates are shown the same as Figure 5.3, where the space is considered mostly collision-free, though still largely uncertain.

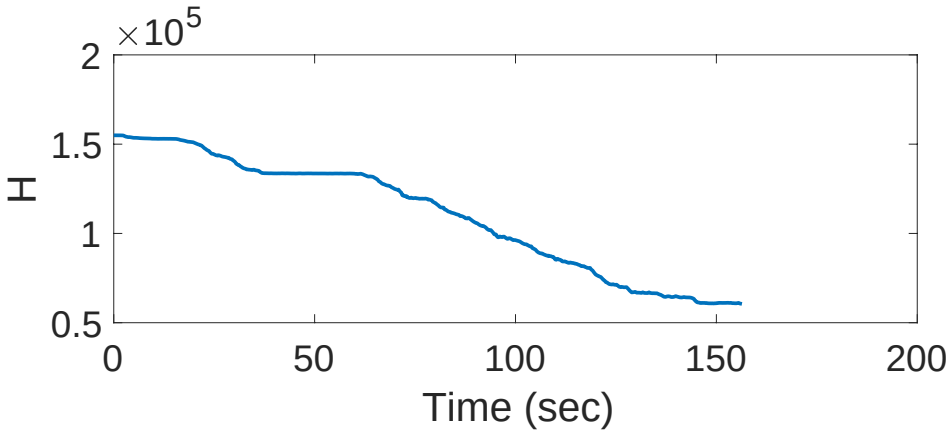


Figure 4.3: Simulated Environment Map Entropy

The entropy of the full 3D map decreases while the robot follows an autonomous exploration scheme based on projected map cells that consider maximum entropy in the vertical direction.

Chapter 5: Multi-Vehicle Autonomous Exploration and Patrol

This chapter covers a bidding-based formulation for multiple vehicles cooperating together in the context of autonomous exploration and patrol. We focus on a series of auctions to assign tasks, and consider dynamic obstacles, such as people, in large office-like environments using a receding horizon framework.

5.1 Bidding-Based Autonomous Exploration

When exploring uncertain environments, robots can generate large maps much faster if they coordinate their efforts. However, the problem of determining a multi-vehicle exploration strategy is complicated and computationally expensive, but must be performed in real-time.

In this section, we formulate a cooperative and autonomous exploration scheme based on sequential auctions for maximizing map information gain. The first auction is based on the expected information gain presented in the prior section, scaled by travel distance. Once the winner of the first auction is determined, the map for bidding is updated based on the expected measurements of the first robot. The second auction for the remaining robots is determined by the information gain from the revised map that is further scaled by the travel distance and a penalty for collision-avoidance. This process is repeated until all robots are assigned tasks.

During this chapter, we assume a 3D vertically-uniform environment at a fixed exploration height as described in Chapter 4 using the 2D projected combination map m_{2D}. Then information gain is based on (3.10) and (3.11), and depends on this reduced map only, i.e.,

$$\mathcal{I}(X_c, m_{\text{2D}}) = H(P(m_{\text{2D}})) - \mathrm{E}\left[H(P(m|X_c, Z_c))\right], \tag{5.1}$$

where the expected scan Z_c is given from (3.5) and (3.8).

5.1.1 Objective Function for the First Auction

Here, we describe how robots participate in the first auction for maximizing map information gain while accounting for travel time. The information gain (3.13) is computed for each candidate pose. Expected information gains may vary among the robots only if they have different sensor configurations. All expected information gains must be computed prior to the first auction.

Next, we describe how travel time is integrated into exploration. Considering that distances from current robot poses to candidate poses may differ greatly, accounting for these varying travel times properly is essential for exploration time efficiency. Travel distances are computed using Dijkstra's search to provide collision-free waypoints for each robot. There are two steps: first, generate a cost map from the robot location to each location on the 2D projected map. This provides the collision-free distances to all reachable locations on the map, which are different for each robot in general. Second, the waypoints from a candidate pose to the current robot pose are easily obtained along the cost map using steepest descent.

The travel time is integrated into the autonomous exploration optimization using a bump function similar to (4.10). Let the distance along the collision-free path from the k-th robot pose, namely X_k, to the c-th candidate pose, namely X_c, be denoted $d(X_k, X_c, m_{\text{2D}}) \geq 0$, which is taken from the cost map belonging to X_k. A continuous bump function is defined to account for traveling costs, and is composed of two parts.

The first part of the bump function corresponds to short trajectories. Consider d_{opt}, the time-optimal distance that a robot may travel at full speed during the time between exploration updates. Here, we consider the case when $d(X_k, X_c, m_{\text{2D}}) \leq d_{\text{opt}}$, i.e., the robot can completely traverse this distance during the allotted travel time. If $d(X_k, X_c, m_{\text{2D}}) \ll d_{\text{opt}}$, then the robot is not moving at its full potential. This can be wasteful, because the robot tends to capture larger regions while moving. Therefore, it is desirable for $d(X_k, X_c, m_{\text{2D}}) \to d_{\text{opt}}$, which corresponds to maximizing map coverage without time cost.

The first part of the bump function is sinusoidal and is optimized at d_{opt} to maximize robotic movement as

$$\mathcal{B}_1(d) = \frac{1}{2} f_{\max} \left(1 - \cos \frac{d\pi}{d_{\text{opt}}} \right), \tag{5.2}$$

where $f_{\max} > 0$ is the maximum value when $d(X_k, X_c, m_{\text{2D}}) = d_{\text{opt}}$.

The second part of the bump function is defined over the domain where $d(X_k, X_c, m_{\text{2D}}) > d_{\text{opt}}$ to minimize time required for X_k to arrive at X_c, putting a penalty on traversing across the environment. This choice is beneficial to generating accurate local maps before exploring new regions. The second half is defined to be nonzero everywhere and is strictly decreasing as

$$\mathcal{B}_2(d) = (f_{\max} - f_{\text{far}}) \exp\left\{ -\beta (d_{\text{opt}} - d)^2 \right\} + f_{\text{far}}, \tag{5.3}$$

where $f_{\max} > f_{\text{far}} > 0$ guarantees $\mathcal{B}_2 > 0$ such that $\mathcal{B}_2 \to f_{\text{far}}$ as $d(X_k, X_c, m_{\text{2D}}) \to \infty$ and $\beta > 0$ assigns the rate of functional decrease relative to $f_{\max}$ and f_{far}. Then, the complete bump function is defined as

$$\mathcal{B}(d) = \begin{cases} \mathcal{B}_1(d), & \text{if } d \leq d_{\text{opt}}, \\ \mathcal{B}_2(d), & \text{otherwise}, \end{cases} \tag{5.4}$$

which is illustrated in Fig. 5.1.

Then, using the information gain from (5.1) and considering the impact from travel distance from (5.4), the objective function for a single agent with respect to X_c is

$$\text{Obj}_{\text{single}}(X_k, X_c, m_{\text{2D}}) = \mathcal{B}(d(X_k, X_c, m_{\text{2D}}))\mathcal{I}(X_c, m_{\text{2D}}). \tag{5.5}$$

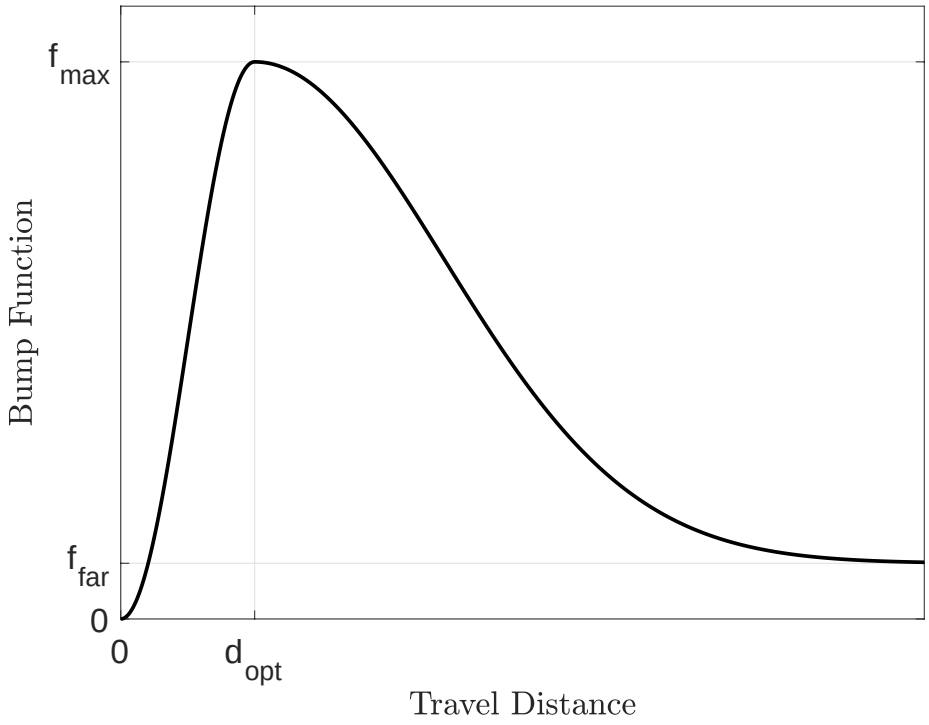


Figure 5.1: Bump Function for Multi-Vehicle Autonomous Exploration

The bump function is designed to promote travel distances close to d_{opt} by multiplying this function by the expected information gain objective. Travel distances less than d_{opt} are below the robot capabilities. Travel distances beyond d_{opt} are beyond what the robot can reach within the minimum exploration computation time.

Then, the optimal pose selection for the k-th agent is

$$X_k^* = \underset{X_c}{\operatorname{argmax}} \, \text{Obj}_{\text{single}}(X_k, X_c, m_{\text{2D}}). \tag{5.6}$$

Like (4.10), the bump function from (5.4) prioritizes robotic movement within the local surrounding space before the robot moves across the map to distant regions, while still considering faraway pose candidates. Unlike (4.10), (5.4) discourages the robot from staying motionless or near-motionless.

The first auction includes bids from all robots, and proceeds as follows. Let the set of all n_R robots be denoted $\mathcal{R} = \{R_1, R_2, \ldots, R_{n_R}\}$ such that the k-th robot, namely R_k, bids $\text{Obj}_{\text{single}}(X_k^*, X_c, m_{\text{2D}})$ from (5.5) and (5.6). This is repeated for all k such that $R_k \in \mathcal{R}$. The robot with the largest bid wins the first auction at index k^*, and is assigned to travel to $X_{k^*}^*$. Then, the remaining robots must account for the planned trajectory of the robot that won the first auction in subsequent auctions, described next.

5.1.2 Objective Function for Subsequent Auctions

The robots unable to win the first auction are updated to reflect the expected impact from the first robot traveling to optimal pose $X_{k^*}^*$. This robot is expected to modify the probabilistic occupancy grid map, which may significantly change the expected information gains and collision properties of candidate poses nearby. The process of auctioning and modifying bids is repeated, removing the winning robot from consideration after each auction, until no robots remain. Between auctions, updating the map with expected measurements from auction-winning robots discourages the remaining robots from entering the same regions and capturing the same cells. As such, there is no need to consider the coverage overlap explicitly as in [53]; instead coverage overlap is avoided in a systematic way as increased coverage overlap would reduce the overall information gain.

More specifically, we coordinate robotic efforts by modifying bids to prevent robots

from updating the same grid cells and to avoid collisions among robots. The goal is to find the winning bids $\mathcal{X}^* = \left\{X_1^*, X_2^*, \ldots, X_{n_R}^*\right\}$ corresponding to each robot from $\mathcal{R}$. During the auctioning process, let $\mathcal{W} \subset \mathcal{R}$ be the set of robots that have already won auctions by producing the largest objective function bid. After the first auction described above, $\mathcal{W} = \{R_{k^*}\}$; after all auctions are complete, $\mathcal{W} = \mathcal{R}$. Between auctions, the candidate poses are modified for information gain and collision-avoidance, coordinating the multi-vehicle exploration.

The first step between auctions is to modify the expected map information gain for efficient map coverage. Once a robot wins a bid, the measurement ray expected values serve to update a temporary copy of m_{2D}, namely $m_{\text{2D,copy}}$, and those candidates located in a local neighborhood (twice the radius of a maximum sensor reading) of the winning candidate from the prior auction are recalculated, thereby coordinating total map information gain with auction-winning robots. Using the same local map notation from (2.7) and (2.6) along a measurement ray, the expected measurement value is

$$\mathrm{E}[z] = \sum_{k=1}^{n_r} \left\{ \prod_{j=0}^{k-1} P(\bar{\mathbf{m}}_j) \right\} P(\mathbf{m}_k) z_k, \tag{5.7}$$

where z_k denotes the distance from the robot sensor to the k-th cell along the measurement ray. Then $\mathrm{E}[z]$ is substituted into (2.5)–(2.7) to modify $m_{\text{2D,copy}}$, and expected map information gains are recomputed with (3.4) and (3.5), while the bump function (5.4) remains the same. In short, the expected changes of m_{2D} are integrated into $m_{\text{2D,copy}}$ for expected information gain estimation between multiple vehicles.

The second step focusses on collision-avoidance between robots based on their proximity. Let $\rho_{\text{max}} > 0$ be a fixed maximum radius to consider collision-avoidance and $\rho_{i,c} \geq 0$ be the Euclidean distance from the i-th already-assigned pose X_i^* to the c-th candidate pose X_c,

i.e.,

$$\rho_{i,c} = \|x_i^* - x_c\|, \tag{5.8}$$

where x_i^* and x_c are the pose locations of X_i^* and X_c, respectively. Then the collision-avoidance factor for the k-th robot such that $R_k \notin \mathcal{W}$ is,

$$\mathbf{C}(X_c) = \begin{cases} \prod_{i|\mathcal{R}_i \in \mathcal{W}} \left(\frac{\rho_{i,c}}{\rho_{\max}}\right)^2, & \text{if } \rho_{i,c} < \rho_{\max}, \\ 1, & \text{otherwise}, \end{cases} \tag{5.9}$$

where this product serves to decrease the value of bids for candidate poses in close proximity with already-assigned poses to avoid collisions between robots. The multi-vehicle objective function for the k-th robot accounting for collision-avoidance is

$$\text{Obj}_{\text{multi}}(X_k, X_c, m_{\text{2D,copy}}) = \mathbf{C}(X_c)\mathcal{B}(d(X_k, X_c, m_{\text{2D}}))\mathcal{I}(X_c, m_{\text{2D,copy}}), \tag{5.10}$$

where its optimal pose during this auction is

$$X_k^* = \underset{X_c}{\text{argmax}}\, \text{Obj}_{\text{multi}}(X_k, X_c, m_{\text{2D,copy}}), \tag{5.11}$$

such that $\text{Obj}_{\text{multi}}(X_k^*, X_c, m_{\text{2D,copy}})$ is the bid for the k-th robot. Optimal pose selection and bidding is repeated for all robots not belonging to $\mathcal{W}$. The largest bid among these wins the auction and is tasked with moving to the associated pose candidate, and then this robot is included with $\mathcal{W}$ to avoid further consideration. Completing n_R auctions produces the set of coordinated poses $\mathcal{X}^*$.

The proposed approach uses auctions and bid modifications based on total map expected information gain and collision-avoidance, promoting exploration of different spaces without explicit consideration of coverage overlaps. The pseudo-code for the bidding process is shown with Algorithm 4.

Algorithm 4: Robot Task Bidding

Function: $Bidding(\mathcal{X}_r, \mathcal{I}(X_c) \forall c \in \mathcal{C}, \rho_{\max})$;
Initialize $k = 0$, $\mathcal{W} = 0_{n_R \times 1}$, $B = 0_{n_R \times 1}$;
Update candidate expected information gains using (3.3);
for $i = n_R, n_R - 1, \ldots, 1$ **do**
    for $j = 1, 2, \ldots, n_R$ **do**
        if $\mathcal{W}(j) == 0$ **then**
            if $i == n_R$ **then**
                Maximize bid (5.5) for X_j^* from (5.6);
            else
                Update $\mathcal{I}(X_c)$ close to $\mathcal{X}^*(k)$;
                for $c = 1, 2, \ldots, n_c$ **do**
                    Find Euclidean distance $\rho_{k,c}$;
                    Get $\mathbf{C}(X_c)$ with (5.9);
                Maximize bid (5.10) for X_j^* from (5.11);
            Insert the j-th maximum bid into $B(j)$;
    k: index maximizing B with corresponding X_k^*;
    $\mathcal{X}^*(k) = X_k^*$;
    $\mathcal{W}(k) = 1$;
    $B(k) = 0$;
Return $\mathcal{X}^*$;

5.1.3 Receding Horizon Framework

This algorithm is further aided by following a receding horizon framework for improved information gain maximization and collision-avoidance with dynamic obstacles and other robots. A receding horizon simply repeats the autonomous exploration steps as quickly as possible over a finite time period, frequently before a robot reaches its desired pose. Since exploration optimizations can only occur during exploration updates, a receding horizon maximizes the rate at which these updates occur. These updates require time for computation, over which the robot can move d_{opt}, used to optimize the robot travel distances with (5.4). Hence, the receding horizon framework serves to repeat optimizations as quickly as possible with maximal robotic movement while exploring a changing occupancy grid.

Furthermore, a receding horizon framework enhances autonomous exploration in two other ways. First, the occupancy grid map is constantly updated while a robot traverses a trajectory, so the expected information gains change as well. Since the bids depend on the occupancy grid, $\mathcal{X}^*$ is updated accordingly. This prevents robots from completing trajectories that have become unnecessary as new terrain becomes captured. Second, the receding horizon prevents robots from colliding with each other when they become too close together (e.g., crossing paths, traversing a tight passage) or other moving obstacles (e.g., a human). The cost map for each robot is updated, and the location of other robots are considered as collision-prone space. Hence, the robot trajectories become better separated due to rapid updates of the cost maps with the receding horizon.

5.1.4 Multi-Vehicle Exploration Numerical Simulation

The algorithms for mapping, exploration, and visualization are designed for the ROS framework. This framework allows the node that computes 3D mapping probabilities to easily transfer these to an exploration node and a visualization node. However, unlike the simulation in Section 4.3, scalability to larger environment is necessary. For this task, the exploration node is separated into three threads running in parallel. The first thread

constantly updates the 2D projected map according to (4.6). The second thread computes the information gains, runs auctions, and generates trajectories. The third thread broadcasts a message for visualizing the 2D projected map. This structure is chosen because the 2D map cannot afford to miss messages, and the last two may produce time bottlenecks and are independent of each other. The final node is for 3D map visualization, and it creates a message to the ROS visualization package, Rviz, to produce 3D cells with varying opacities corresponding to cell occupancy probabilities.

The multi-vehicle exploration problem is typically applied to large environments, so several additional scalability precautions are taken to avoid large communications and computations. Most importantly, only the map *changes* are communicated from the mapping node to the exploration and visualization nodes using a custom ROS message type. This change drastically decreases the time needed to generate or interpret the message. This is accomplished by finding a rectangular prism of the minimum and maximum limits of cells that may have changed during a particular measurement scan update. Publishing and subscribing to map changes provides an efficient means to transfer mapping information, independent of map size.

Another computational improvement key to autonomous exploration success is that only important and changing map information gains are updated. The expected information gain for a single candidate is updated ray-by-ray, where only the top $\hat{n} = 5$ cells with the largest detection probabilities are considered using (3.6). Since (3.4) is embedded in (3.3), the computational complexity for a ray with n_r cells is $\mathcal{O}(n_r^2)$. Reducing the considered cells to $\hat{n}$ limits the computation to $\mathcal{O}(n_p^2)$ where $n_p \ll n_r$ in general. Furthermore, finding these $\hat{n}$ cells is computationally inexpensive; using a selection algorithm of the n_r cells and a sorting algorithm of the n_p cells, the computational complexity is amortized to $\mathcal{O}(n_r + \hat{n}\log(\hat{n}))$, which can be further approximated to $\mathcal{O}(n_r)$ assuming that $\hat{n} \ll n_r$. Thus, this entropy updating scheme scales well to varying sensor ranges and cell sizes. Furthermore, the information gains of poses far from any robot trajectory need not be updated between

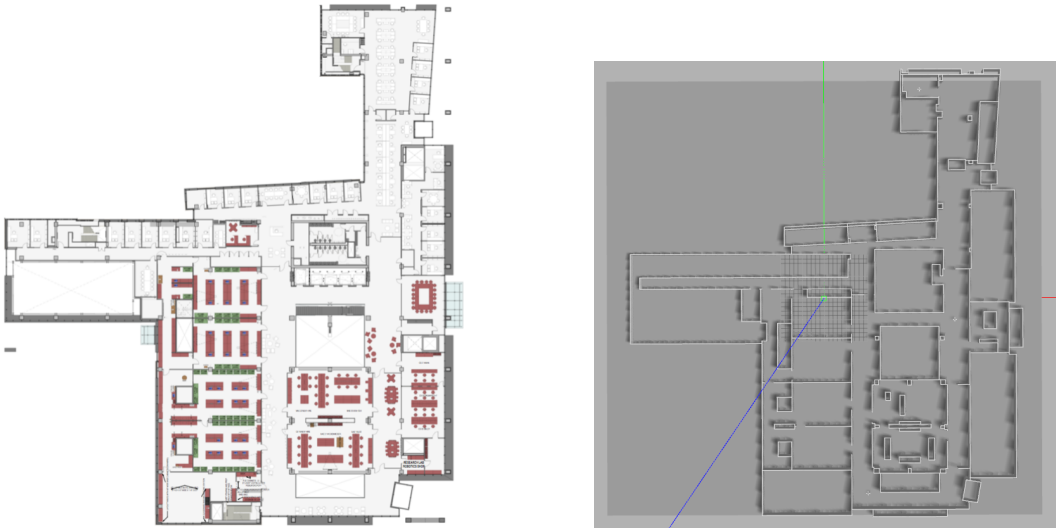
receding horizons.

Parameters and Resulting Maps The 3D mapping and exploration algorithms for three robots are simulated with several parameters, which are the same as those in Section 4.3 except the map limits are extended to -55.0 m to 55.0 m in the x- and y-directions, and 0.0 m to 1.5 m in the z-direction, producing a total of $45,255,504$ grid cells with edge length 0.075 m (dimensioned $1468 \times 1468 \times 21$). For the bump function, $\mathcal{B}_{\text{max}} = 1$, $\mathcal{B}_{\text{far}} = 0.1$, and $\beta = 0.01$ are chosen. For bidding $\rho_{\text{max}} = 3.0$ m is chosen.

In the simulation, three quadrotors explore the simulated environment of The George Washington University's (GWU) Science and Engineering Hall (SEH) for 10 minutes while taking measurements. Each quadrotor is given an Asus Xtion depth sensor and Hokuyo LIDAR, where the sensor properties for the Asus Xtion are used with exploration expectations because they provide the vast majority of measurements. The building floor plan and Gazebo simulated environment are shown in Fig. 5.2. The 3D occupancy grid maps are shown in Fig. 5.3, and the corresponding 2D projected maps for exploration are shown in Fig. 5.4 using the ROS visualization package Rviz. The full video of both maps is available at https://youtu.be/OMn2c453oik.

The resulting 3D occupancy grid shows a nearly-completed map after three quadrotors cooperatively explored over 10 minutes, which is quantified with total map entropy from (3.2), shown in Fig. 5.5. An important observation is that the majority of map space was unreachable by any robot, so about 1.2×10^7 of the entropy metric could not be improved, regardless of mapping or exploration strategy.

Special attention had to be placed on parameter selection for the bump function and collision-avoidance. For the bump function, increasing the parameter β narrows the bump, placing a stronger influence on nearby candidates. However, this choice corresponds to the bump function decreasing at a negligible rate beyond a close neighborhood of the robot, i.e., a large range of distances yield $\mathcal{B} \approx \mathcal{B}_{\text{far}}$. Hence, there is a tradeoff between prioritization of



(a) Floor Plan (b) Gazebo Model

Figure 5.2: SEH Floor Plan

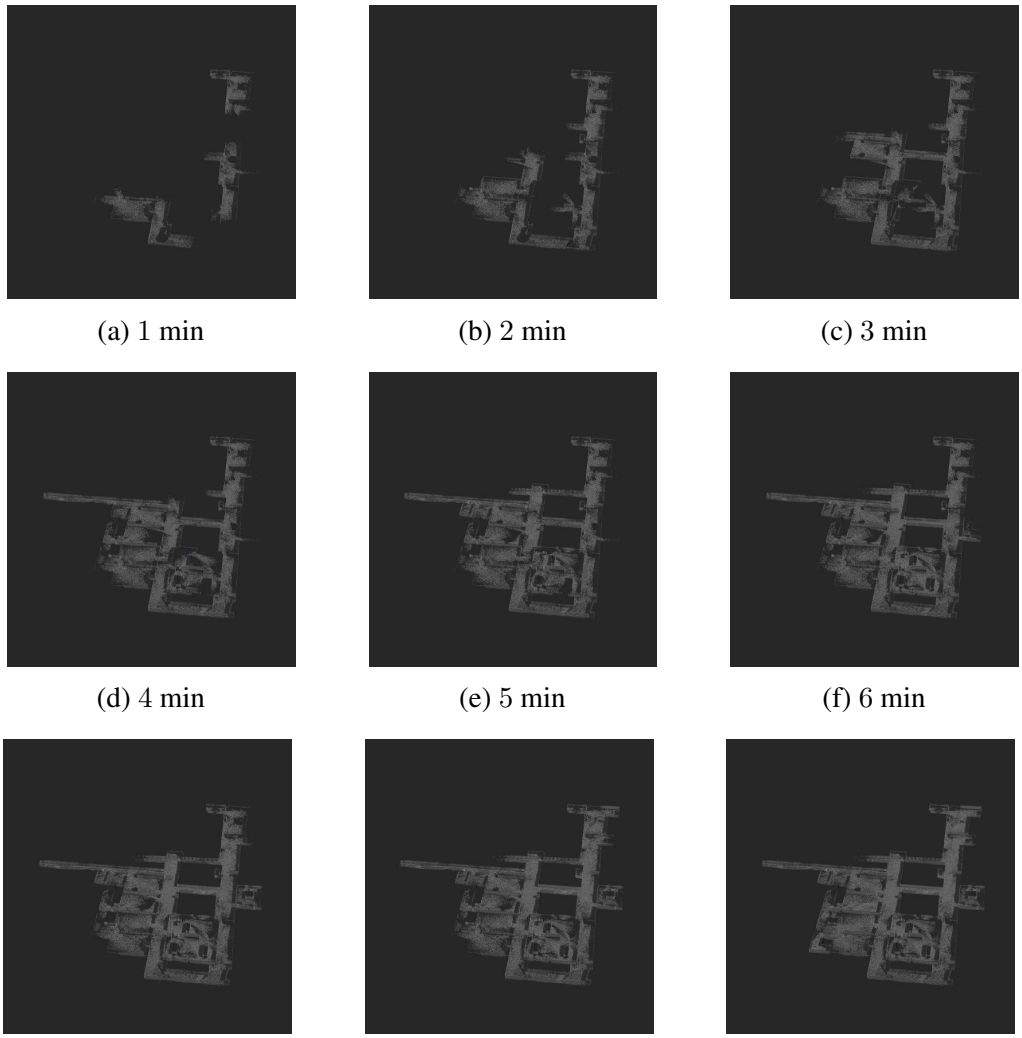
The SEH second floor plan (about $15,000$ sqft) is reconstructed with a Gazebo simulation environment, excluding some small details and features of closed rooms.

local movements and being able to differentiate the effects of distance on large trajectories.

When robots competed for future poses in a centralized auction framework, the repeated optimizations provided an effective multi-vehicle exploration policy. Between auctions, only a small set of candidates within the neighborhood of the last auction-winning candidate require consideration when updating expected information gains with (5.7) and avoiding collisions between robots using (5.9), both of which modify the optimization of (5.11). Since bidding is applied efficiently, its computation time is negligible compared with other parts of the exploration update such as computing the initial candidate information gains and robot cost maps.

5.2 Multi-Vehicle Cooperative Patrol

Patrol is the continuous and repetitive observation of different regions within a larger environment. An effective patrol algorithm is particularly valuable for surveillance operations, especially when the robot members are incapable of measuring all regions of an environment



(a) 1 min (b) 2 min (c) 3 min

(d) 4 min (e) 5 min (f) 6 min

(g) 7 min (h) 8 min (i) 9 min

Figure 5.3: 3D Occupancy Grid Map of SEH Second Floor During Multi-Vehicle Exploration

The 3D occupancy grid map of SEH is generated from three robots coordinating their exploration strategy with auctions.

from fixed poses. Here, we formulate cooperative patrol as an optimization problem, where expected map information gains are maximized cooperatively according to the methods outlined in Section 5.1. However, the grid cells experience slow degradation over time, thereby modifying the expected map information gains that govern exploration. This way, different regions are revisited after some time passes. This method of cell degradation for cooperative robotic patrol is described next.

(a) 1 min (b) 2 min (c) 3 min

(d) 4 min (e) 5 min (f) 6 min

(g) 7 min (h) 8 min (i) 9 min

Figure 5.4: 2D Projected Occupancy Grid Map During Multi-Vehicle Exploration

The 2D projected map serves for both entropy and collision information at the exploration height for all three vehicles as they coordinate their mapping efforts with bidding-based multi-vehicle autonomous exploration.

5.2.1 Continuous-Time Markov Process for Cell Degradation

The key idea to cell degradation is that as time progresses forward, cells become more uncertain, which causes the exploration algorithm to revisit these cells. All cells are degraded through a continuous-time Markov process at an equal rate, thereby creating a fair policy for revisiting regions that exploits the benefits of multi-vehicle exploration, such as following a receding horizon and optimizing information gain and travel time. Cell degradation is

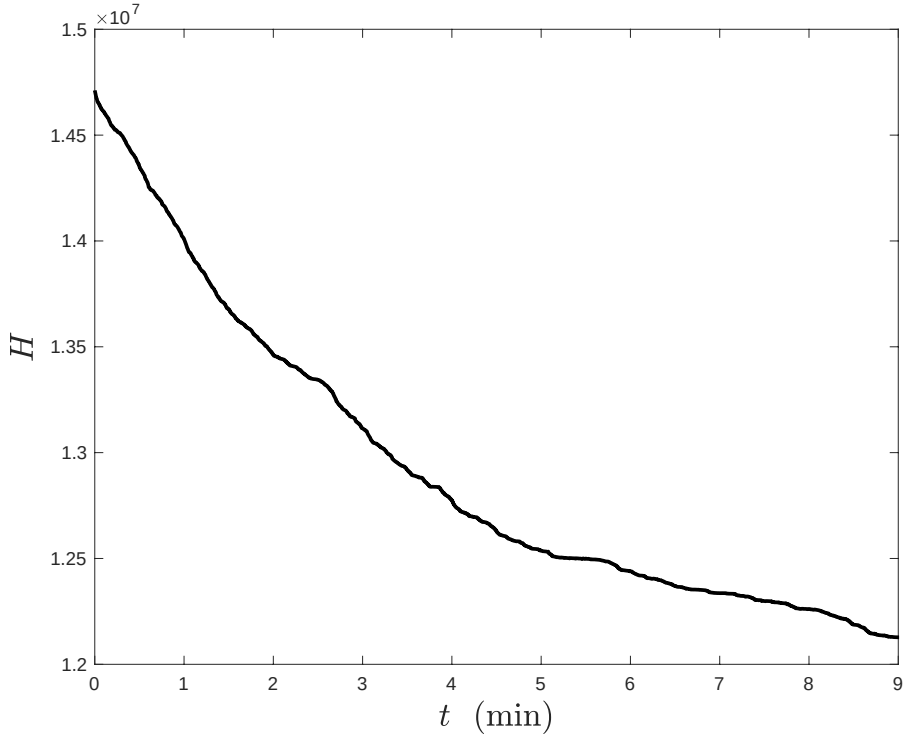


Figure 5.5: Total 3D Map Cell Entropy for Multi-Vehicle Exploration

The total map cell entropy is calculated using the difference in map entropy before and after each measurement scan from each of the three coordinating vehicles.

applied simply and uniformly to the entire 3D map for autonomous cooperative patrol. Since this analysis is valid for all grid cells, cell index notation is neglected for this analysis.

Consider that a single grid cell begins with probability $P(\mathbf{m}_{t_0})$ at time t_0 and degrades to $P(\mathbf{m}_{t_f})$ at time $t_f > t_0$. Suppose that as time progresses forward after t_0, the cell probability approaches a terminal value $P_\infty \in (0, 1)$, i.e.,

$$\lim_{t\to\infty} \begin{bmatrix} P(\mathbf{m}_t) \\ P(\bar{\mathbf{m}}_t) \end{bmatrix} = \begin{bmatrix} P_\infty \\ 1 - P_\infty \end{bmatrix}. \tag{5.12}$$

Let the degradation rate be $\lambda > 0$ and matrix $A \in \mathbb{R}^{2\times 2}$ be defined as

$$A = \lambda \begin{bmatrix} -(1 - P_\infty) & P_\infty \\ (1 - P_\infty) & -P_\infty \end{bmatrix}. \tag{5.13}$$

Then the first-order ordinary differential equation is expressed as

$$\begin{bmatrix} \dot{P}(\mathbf{m}_t) \\ \dot{P}(\bar{\mathbf{m}}_t) \end{bmatrix} = A \begin{bmatrix} P(\mathbf{m}_t) \\ P(\bar{\mathbf{m}}_t) \end{bmatrix}. \tag{5.14}$$

Hence, the state transition from t_0 to t_f is

$$\begin{bmatrix} P(\mathbf{m}_{t_f}) \\ P(\bar{\mathbf{m}}_{t_f}) \end{bmatrix} = \exp\{A(t_f - t_0)\} \begin{bmatrix} P(\mathbf{m}_{t_0}) \\ P(\bar{\mathbf{m}}_{t_0}) \end{bmatrix}, \tag{5.15}$$

which is illustrated with Fig. 5.6. Most importantly, (5.15) satisfies (5.12) as $t_f \to \infty$. Solving the top row of (5.15) yields

$$P(\mathbf{m}_{t_f}) = P(\mathbf{m}_{t_0}) \exp\{-\lambda(t_f - t_0)\} + P_\infty(1 - \exp\{-\lambda(t_f - t_0)\}). \tag{5.16}$$

Applying degradation is computationally-inexpensive due to the simplicity of (5.16). For an arbitrary positive fixed time step, $\exp\{-\lambda(t_f - t_0)\}$ needs only be calculated once

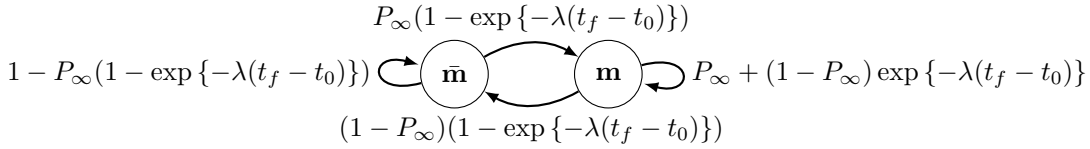


Figure 5.6: Continuous-Time Markov Process Illustration

The continuous-time Markov process for a single binary cell occupancy is illustrated. As time progresses, this process degrades the cell slowly toward P_∞.

and is multiplied to every map cell. Similarly, the second term of (5.16), namely $P_\infty(1-\exp\{-\lambda(t_f-t_0)\})$, is simply added to every cell. For grid cells with probabilities far from P_∞, the times to converge to this final degraded value increase, shown in Fig. 5.10. Slow convergence of cells with probabilities close to 0 or 1 allows the robots to stop viewing these cells, giving the robots opportunities to visit different regions before returning.

In short, each grid cell is continuously degrading according to (5.16) toward P_∞. Only in the presence of measurements does the cell occupancy become well-known according to (2.5)–(2.6). Cell degradation encourages the robot to revisit cells from the distant past using bidding-based multi-vehicle autonomous exploration to patrol the environment continuously. Adopting the receding horizon framework allows patrol to account for dynamic obstacles by quickly updating patrol strategies as map information changes.

5.2.2 Multi-Vehicle Patrol Numerical Simulation

In this section, we present a multi-vehicle exploration simulation that is identical to the scenario in Section 5.1.4, except for two key differences. The first is that patrol is included in the exploration scheme to incentivize the robots to revisit areas after time passes. The second is that the robots must handle a human walking around the space, unknown to the cooperating robots, requiring the receding horizon framework to avoid collisions with this dynamic obstacle.

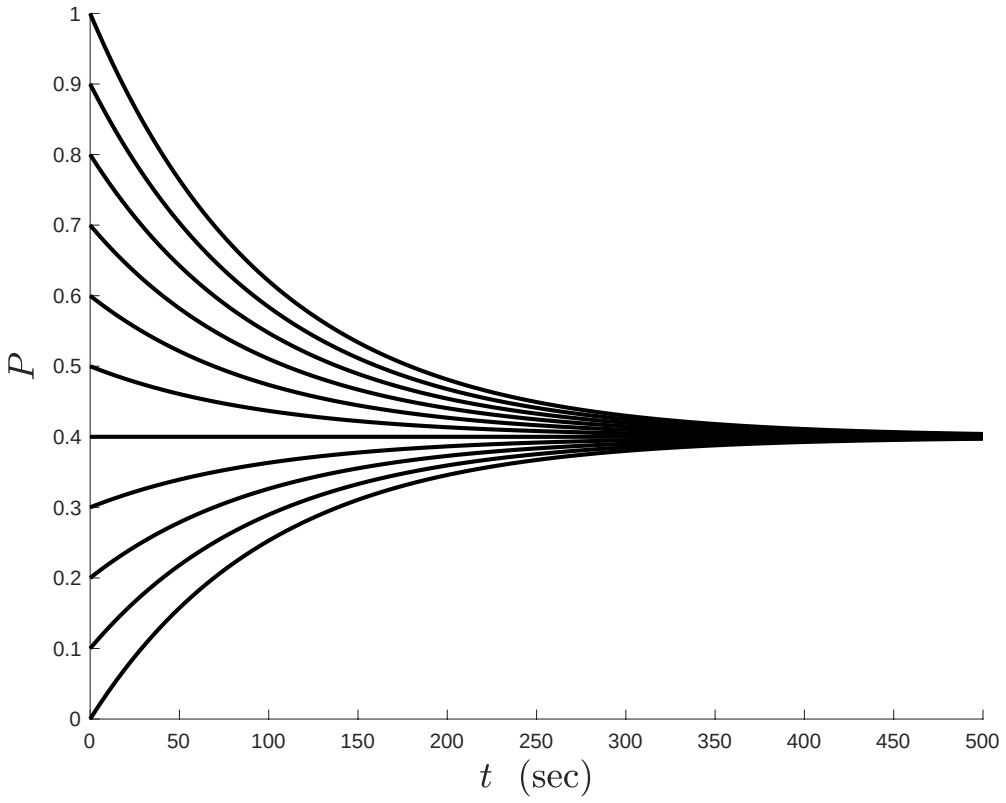


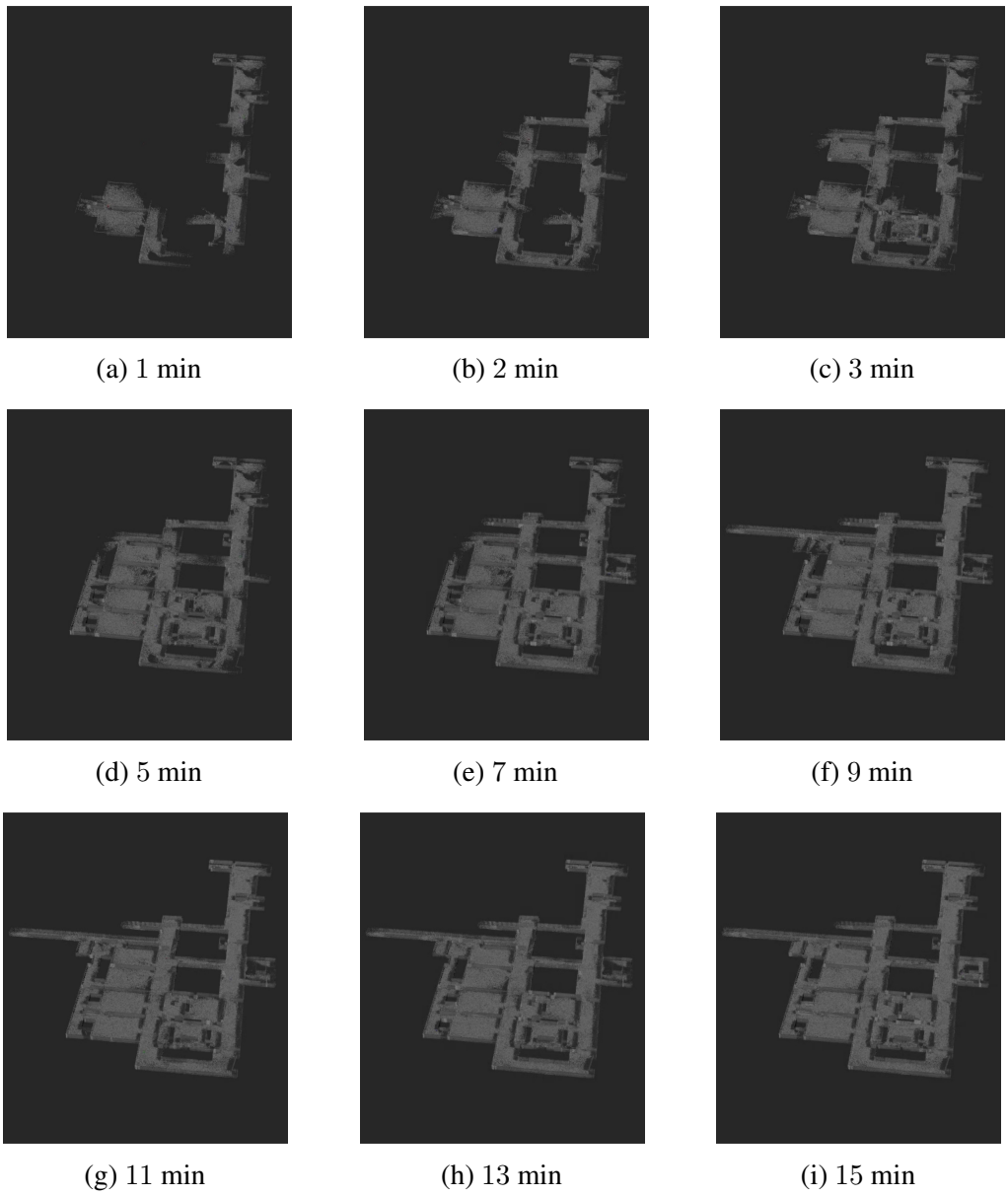
Figure 5.7: Grid Cell Degradation Illustration

The rate at which probabilities converge toward $P_\infty = 0.4$ increases based on the difference magnitude of their initial values.

Degradation Thread An additional thread within the mapping node handles cell degradation with $\lambda = 1 \times 10^{-3}$, which corresponds to a cell half life of roughly 11.5 minutes; if this time passes since a measurement captures a cell, the occupancy probability will be halfway between its value 11.5 minutes ago and $P_\infty = P_{\text{init}} = 0.1$. While degrading cells with (5.16) is inexpensive, degradation must cycle through all cells, which can be costly with a map containing over 45 million cells. Since all cells are affected by degradation, simply sending the map *changes* from measurement scan updates alone is insufficient for the exploration node to track all changing probabilities. Thus, the complete map is sent from this mapping thread, and an additional thread in the exploration node subscribes to this message to update the projected map $m_{2\text{D}}$ slowly. The sluggish speed of this thread is acceptable, as cell degradation is a slow process, so fast updates are uncritical. Both the mapping and exploration nodes require an additional thread for degradation.

Simulated Results In the simulation, three quadrotors explore and patrol the same simulated environment from Section 5.1.4 of The George Washington University's (GWU) Science and Engineering Hall (SEH) for 15 minutes while taking measurements. Initially, the robots are given no information about the environment except that their immediate surroundings are free. Unknown to the robot team, a simulated human walks in a rectangular repeated pattern at 0.5 m/s. The 3D occupancy grid maps are shown in Fig. 5.8, and the corresponding 2D projected maps for exploration are shown in Fig. 5.9. The full video of both maps is available at https://youtu.be/bsLG2romP_8.

The robot generates a fairly-complete occupancy grid within 9 minutes, and proceeds to patrol the space during the remaining time. After 9 minutes, the total map entropy no longer increases or decreases significantly; the rate of cell degradation is roughly equal and opposite to improvements to the probabilistic map from the robot team. In contrast, if the robots simply hover instead of patrol after 9 minutes, the map uncertainty quickly increases, shown in Fig. 5.10. Furthermore, the receding horizon framework minimizes time between



(a) 1 min (b) 2 min (c) 3 min

(d) 5 min (e) 7 min (f) 9 min

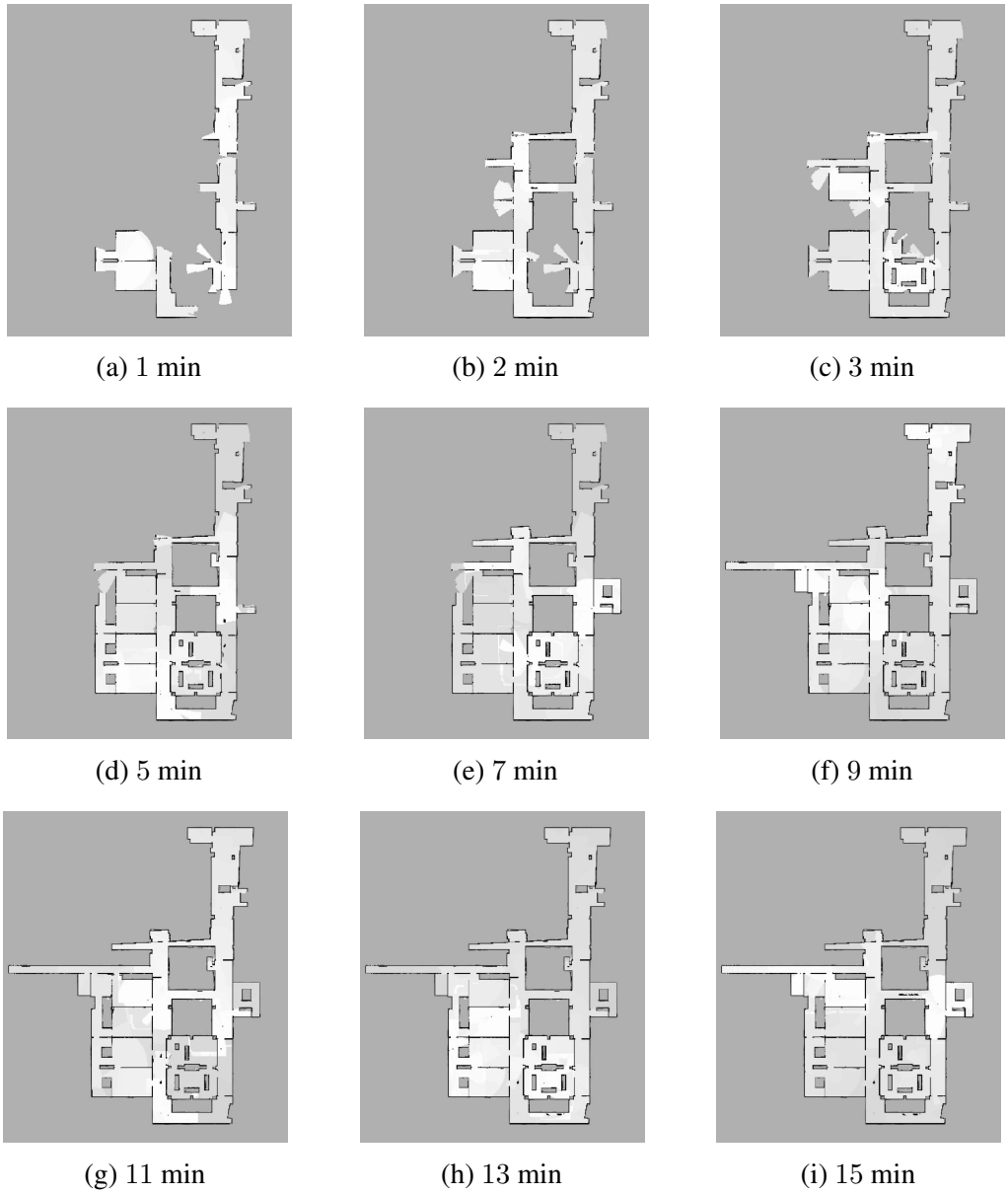
(g) 11 min (h) 13 min (i) 15 min

Figure 5.8: 3D Occupancy Grid Map of SEH Second Floor During Multi-Vehicle Patrol

The 3D occupancy grid map representation of the SEH second floor is shown during patrol. The robots generate these maps while avoiding collisions with each other, the environment, and a non-cooperative human.

exploration updates, which allows the robots quickly react to dynamic obstacles like the non-cooperative human, shown in Fig. 5.11.

The results also demonstrate an interesting tradeoff between exploring new terrain and patrolling previously-visited spaces. When the degradation rate λ is increased, grid cells are



(a) 1 min (b) 2 min (c) 3 min

(d) 5 min (e) 7 min (f) 9 min

(g) 11 min (h) 13 min (i) 15 min

Figure 5.9: 2D Projected Occupancy Grid Map During Multi-Vehicle Patrol

The 2D projected map from of the SEH second floor shows degradation while robots patrol the space. The robots evade the non-cooperative human walking in a rectangular pattern, occasionally capturing the temporarily-occupied space and identifying these cells as occupied on the grid.

degrading faster, incentivizing the exploration strategy to revisit spaces faster. This behavior is beneficial for frequent patrol, but certain areas, particularly those difficult to reach, are not visited as quickly or as frequently. Should an area receive extra attention, a greater value of λ may be applied to this region.

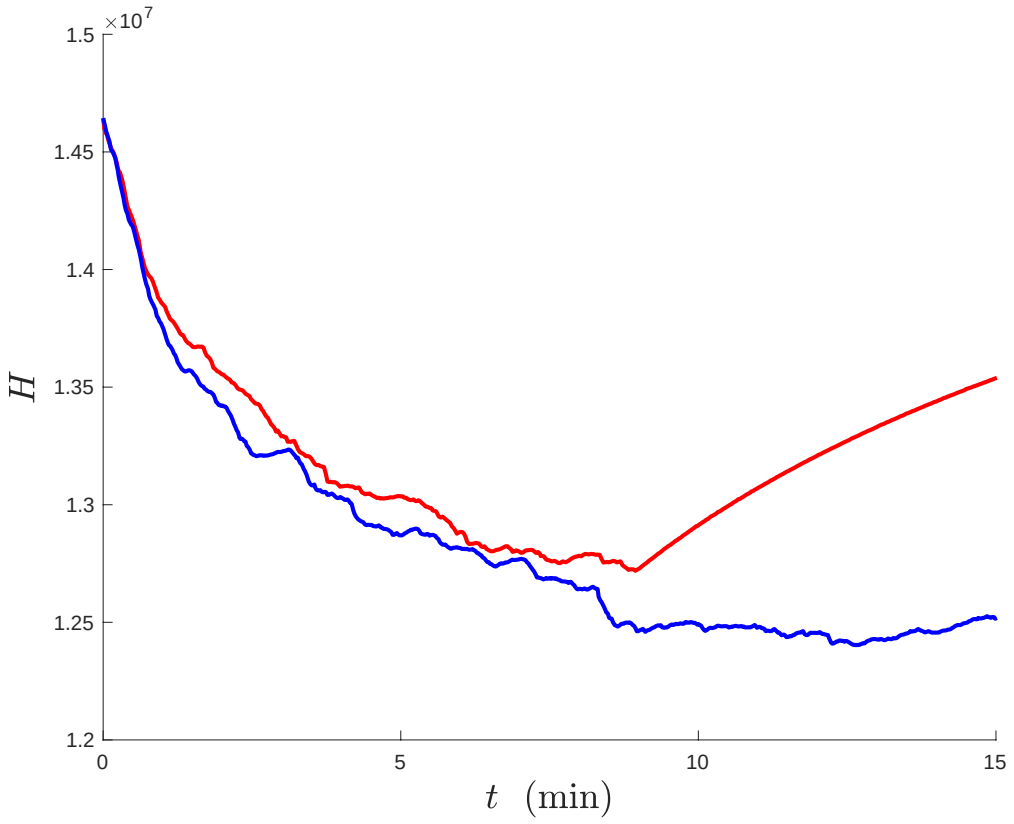


Figure 5.10: Total 3D Map Cell Entropy for Multi-Vehicle Patrol

Total map entropies are tracked for separate trials: patrol of the building for 15 minutes (blue) and patrol of the building for 9 minutes, then hover (red). Even though the map is fairly well-known after 9 minutes in either case, the continuous effort of patrolling over the complete time period shows lower map uncertainty beyond this time as cells are constantly degrading.

(a) 15 sec (b) 16 sec (c) 17 sec

Figure 5.11: Human Collision-Avoidance Using a Receding Horizon

A cooperating robot must evade a non-cooperative human. The receding horizon framework allows a quickly-updating map to modify the exploration commands to avoid collisions with a dynamic obstacle.

5.3 Conclusions

This chapter introduced a bidding-based framework for cooperative autonomous exploration and patrol. These contributions exploit the values of the objective functions used in autonomous exploration optimizations for bids over a series of auctions. Between auctions, a copy of the map and collision criteria are updated, thereby coordinating the exploration efforts. Finally, a patrol scheme is developed based on simple cell degradation, promoting periodic visitations to the same regions over time.

Chapter 6: Autonomous Exploration in Complex 3D Space

This chapter extends the autonomous exploration scheme to complex 3D environments. Unlike Chapter 3 where exploration was applied in 2D, or Chapters 4 and 5 where exploration was confined to a single height in vertically-uniform environments, this chapter proposes a completely 3D exploration approach for complex environments. This proposed approach is designed for rooms with complicated features and places with changing and uncertain terrain such as Mars.

6.1 Exploration in 3D

In this section, we propose how multiple measurement rays in 3D can determine map information gain from a pose candidate [30]. Then, we show how a collision-free trajectory is determined with a 3D Dijkstra's search. Finally, we combine 3D information gain and travel costs into a single optimization.

6.1.1 Map Information Gain in 3D

The expected entropy for an arbitrary 1D ray spanning 3D space can be calculated with (3.3) and (3.4) in a computationally tractable manner. However, computing the expected entropy from multiple rays simultaneously has exponential complexity, and is therefore computationally intractable. Additionally, numerous rays of a single scan commonly intersect the same grid cells, making consideration of *every* measurement ray unnecessary.

Instead of considering expected entropy from a complete scan, we propose a real-time solution that selects sample measurements to determine an optimal attitude at each candidate pose and the associated map information gain, similar to how the scan information gain is found in Section 3.2.1. We assume the vehicle is capable of level flight, so the third axes of

the world and body are aligned (Figure 6.1), so R_c can be expressed as

$$R_c = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad 0 \leq \psi < 2\pi,$$

where ψ represents the angle about the third body-fixed axis. The direction of a 3D measurement may also have a nonzero component in the vertical direction. This is achieved by rotating the sensor frame angle θ about the second sensor-fixed axis,

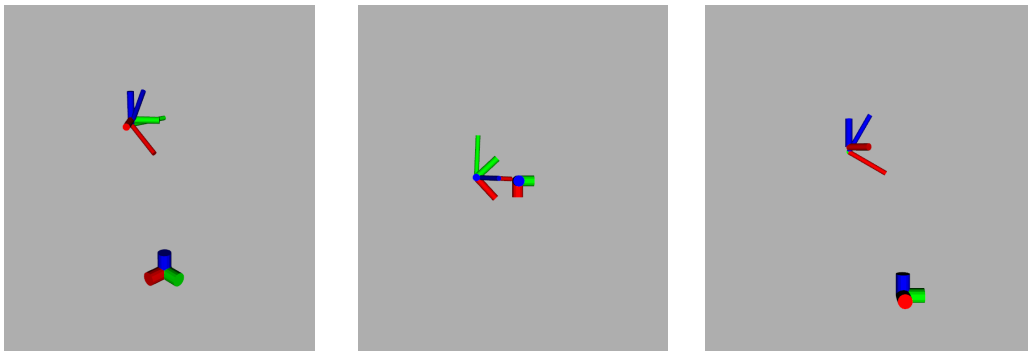
$$R_{c,\text{sensor}} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}, \quad -\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}.$$

Combining these two rotation matrices, the measurement z with depth $\|z\|$ is expressed with respect to a frame fixed to the world as

$$z(\psi,\theta) = \begin{bmatrix} \cos\theta\cos\psi & -\sin\psi & \cos\psi\sin\theta \\ \cos\theta\sin\psi & \cos\psi & \sin\psi\sin\theta \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \|z\| \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi \\ \cos\theta\sin\psi \\ -\sin\theta \end{bmatrix} \|z\|. \tag{6.1}$$

In short, unit vectors are acquired via Euler angles within certain sensor limits.

Next, we show how multiple 3D measurements provide an estimate for information gain. Let Ψ and Θ be the angular ranges for the sensor FOV in the horizontal and vertical directions, respectively. Within the sensor FOV, we select sample measurements for evaluating their expected information gains. Let the number of sample measurements be n_ψ and n_θ for horizontal and vertical rotations, respectively. Since R_c and the FOV are known, the set of sample measurement rays, namely $\mathcal{Z}(R_c)$, is known as well. Then, the expected information gain for a single measurement ray z from candidate position x_c and attitude R_c is the



(a) General View (b) ψ Rotation (c) θ Rotation

Figure 6.1: Illustration of 3D Exploration Reference Frames

The three frames are shown with three axes (red: first axis, green: second axis, blue: third axis). The world frame (short and thick) is fixed to the ground. The body-fixed frame (medium thickness and length) is fixed to the camera frame (long and narrow). The rotation of ψ (b) represents the fixed camera yaw rotation and θ (c) represents the downward angle of the camera to capture the ground below the flying robot.

negative expected entropy change,

$$\begin{aligned}&\mathcal{I}_{\text{ray}}(x_c, R_c, \psi, \theta) \\ &= \begin{cases} H(P(r)) - \mathrm{E}[H(P(r|x_c, z(\psi+\tilde{\psi}, \theta+\tilde{\theta})))], & \text{if } z(\psi+\tilde{\psi}, \theta+\tilde{\theta}) \in \mathcal{Z}(R_c), \\ 0, & \text{otherwise}, \end{cases}\end{aligned} \tag{6.2}$$

where the mounting yaw $\tilde{\psi}$ and pitch $\tilde{\theta}$ of the sensor is a fixed rotation in the horizontal and vertical directions, respectively in that order, such that $-\pi < \tilde{\psi} \leq \pi$ and $-\frac{\pi}{2} + \frac{\Theta}{2} \leq \tilde{\theta} \leq \frac{\pi}{2} - \frac{\Theta}{2}$. A positive $\tilde{\theta}$ corresponds to rotating the depth sensor from a forward direction to a downward direction toward the ground below. The expected information gain for the full scan is the summation of expected information gains for individual rays

$$\mathcal{I}_{\text{scan}}(x_c, R_c) = \sum_{i_\psi=1}^{n_\psi} \sum_{i_\theta=1}^{n_\theta} \mathcal{I}_{\text{ray}} \left(x_c, R_c, \frac{(2i_\psi - n_\psi - 1)\Psi}{2n_\psi}, \frac{(2i_\theta - n_\theta - 1)\Theta}{2n_\theta} \right). \tag{6.3}$$

Finally, (6.3) is computed for n_ψ possible attitudes when the robot is at location x_c, one

corresponding to each yaw angle ψ. Then, the optimal attitude at x_c is

$$R_c^* = \underset{R_c}{\operatorname{argmax}}\, \mathcal{I}_{\text{scan}}(x_c, R_c). \tag{6.4}$$

In short, the expected information gain from each possible measurement ray is obtained using predicted entropy from (3.3) and (3.4). Then the expected information gain from all possible scans at a candidate location is calculated from (6.2) and (6.3). Finally, the optimal attitude at this candidate location is found using (6.4).

6.1.2 Collision-Free Trajectory in 3D

Next, we present a fairly straightforward application of Dijkstra's search to 3D occupancy grid mapping. First we reduced the number of grid cells to cubic blocks large enough for a robot to fit completely inside. Based on the occupancy probability of these blocks and their neighbors, we determine which cells are reachable, and what are their travel costs by building a cost map based on Dijkstra's algorithm.

Here we describe how a reduced map is generated for collision avoidance and motion planning. If the largest edge-to-edge distance of the robot, namely ρ, is not larger than the 3D grid cell size α, this step is unnecessary. Let $k \geq 1$ be an integer such that $k\alpha \geq \rho$ and $(k-1)\alpha < \rho$, i.e., k is the minimum number of grid cells in each dimension capable of fully-enclosing the robot inside this cube. Then, we decompose the complete probabilistic 3D into map m_{reduced}, which is composed of larger cubic cells that encompass k^3 cells from map m. Let $\mathbf{m}_{\text{reduced},k}$ denote the k-th cell of m_{reduced} being occupied, and let $a_k \subset m$ be those k^3 cells from m composing this larger cell. Then, we apply the same criteria as (4.6) to obtain the probability of $\mathbf{m}_{\text{reduced},k}$ in 3D,

$$P(\mathbf{m}_{\text{reduced},k}) = \begin{cases} \max_{i \in a_k} P(\mathbf{m}_i), & \text{if } \max_{i \in a_k} P(\mathbf{m}_i) \geq P_{\text{thresh}}, \\ \min_{i \in a_k} P(\mathbf{m}_i), & \text{otherwise}, \end{cases} \tag{6.5}$$

where initial probability is $0 < P_{\text{init}} < 1$ and the threshold probability is constrained to $P_{\text{init}} < P_{\text{thresh}} < 1$. This approach uses probabilities that have changed from P_{init}, and favors occupied cells to avoid risking collisions.

Next, we show how $\mathbf{m}_{\text{reduced},k}$ is used in Dijkstra's search for collision avoidance and motion planning. Defining $0 < P_{\text{coll}} < 1$ as the acceptable probability of collision, every grid cell of m_{reduced} is considered a safe and unvisited node if its occupancy probability, and the occupancy probabilities of its neighbors (sharing a face, edge, or corner), are below P_{coll}. A cost map is built from the starting robot location by neighboring nodes, where the cost to travel to another node is based on Euclidean distance: a cell sharing a face is α away, a cell sharing an edge is $\sqrt{2}\alpha$ away, and a cell sharing a corner is $\sqrt{3}\alpha$ away. Generating a cost map for all reachable cells in $\mathbf{m}_{\text{reduced},k}$ provides a collision-free travel cost for all reachable candidate pose locations. Once an optimal pose is selected, the path is easily found from the cost map using steepest descent.

6.1.3 Optimal 3D Pose

Here we present how an optimal pose is selected based on information gains at optimal attitudes and known collision-free travel costs. This process is similar to prior the autonomous exploration proposed in earlier chapters, with a few minor differences.

The proposed 3D autonomous exploration follows a receding-horizon framework, and uses a bump function (5.2)–(5.4) to account for travel distance. The only difference is that the start of the bump function $B_1(d) = f_{\text{max}}$, illustrated in Figure 6.2). This change, though minor, avoids the possible scenario where a robot repeatedly alternates between neighboring candidates with similar expected information gains. Then, we incorporate

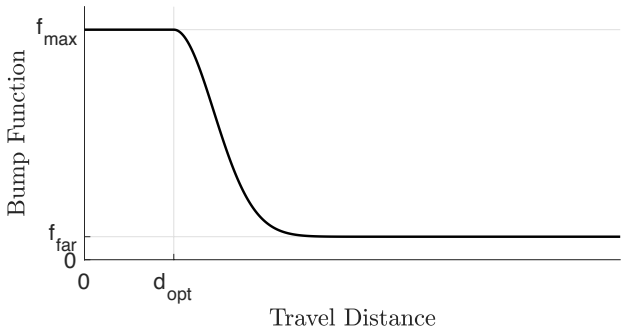


Figure 6.2: Bump Function for Full 3D Exploration

The bump function is multiplied to the objective function of (6.6) to prioritize local trajectories to avoid traversals across the map, where travel costs are determined using Dijkstra's algorithm. However, there is no time wasted up to d_{opt} because this distance may be achieved in the minimum exploration computation time.

expected information gain and travel costs into a unified optimization. Given $d(x_c)$ from Dijkstra's cost map, the optimal candidate location is

$$x_c^* = \underset{x_c}{\operatorname{argmax}} \mathcal{I}_{\text{scan}}(x_c, R_c^*)\mathcal{B}(d(x_c)), \tag{6.6}$$

and therefore the optimal pose is $X_c^* = \{x_c^*, R_c^*\}$.

In conclusion, every reachable candidate pose is considered for its optimal attitude, which is found using sample rays spanning 3D space. The travel costs, acquired from Dijkstra's search, serve as input for the bump function, which prioritizes local trajectories to avoid costly map traversals. Then, the optimal candidate location is selected to maximize a combination of expected information gain and travel cost.

6.2 Numerical Simulation

In this section, we demonstrate the efficacy of the proposed 3D mapping and autonomous exploration with a simulated Mars environment. First we cover the parameters, then compare results using two different maps for entropy prediction.

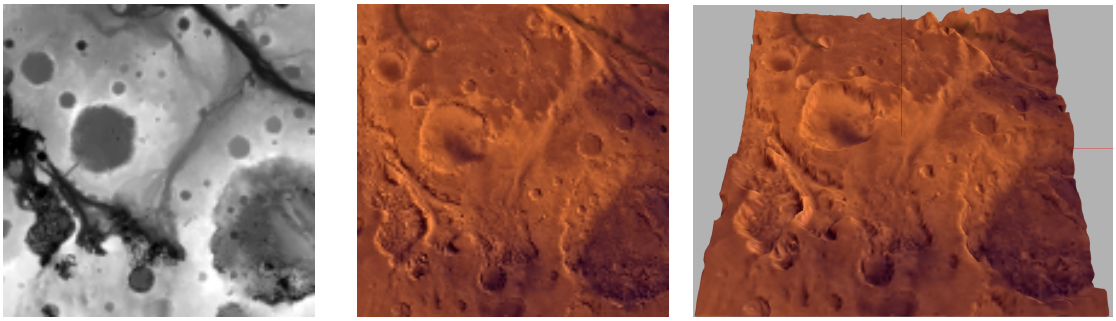
6.2.1 Mars Exploration Motivation

The topography of Mars has several benefits in future mission planning for landing spacecraft and image distortion correction, among other research objectives. The Mars Orbiting Laser Altimeter (MOLA) mission greatly improved the topography map of the surface, but was limited to an uncertainty of roughly eight meters [54]. The only precise topography knowledge about the surface comes from four rovers, two of which are inactive, and have covered only a small portion of the surface area of Mars. Approaches for exploring environments similar to offices, such as [43], are incompatible with the complex terrain presented in this numerical example.

6.2.2 Mars Parameters

A 3D environment of Mars is simulated in Gazebo. A height map [1] is used to generate a contoured surface, and the corresponding picture of Mars is draped over this contour, shown in Figure 6.3. A 3D laser scanner is also simulated in Gazebo. In the horizontal direction, the sensor has limits $\Psi = 120^\circ$ with a total of 1000 measurement rays inside, the sensor is fixed at angle $\tilde{\psi} = 45^\circ$, and $n_\psi = 16$ sample measurements are used. In the vertical direction, the sensor also has limits $\Theta = 120^\circ$ with a total of 1000 measurement rays inside, but the sensor is fixed at angle $\tilde{\theta} = 30^\circ$, and $n_\psi = 7$ sample measurements are used. These ray samples are taken from each candidate location, which are separated 1 m apart in each of the 3 dimensions.

The map parameters are also important to the success of the exploration. The full map has dimensions 20 m $\times$ 20 m $\times$ 5 m in the Mars-fixed frame, with cell edge length $\alpha = 0.075$ m. The reduced map for collision avoidance and motion planning has cells $k = 3$ times the size (0.225 m), so $k^3 = 27$ cells are considered in (6.5). The bump functions use $f_{\text{max}} = 1.0$, $f_{\text{far}} = 0.1$, and $\beta = 0.1$ to account for travel costs with (5.4). The receding horizon optimal time d_{opt} is based on a fixed robot velocity of 0.25 m/sec and the computation times vary from 1.8 sec to 2.5 sec.



(a) Height Map (b) Color Map (c) Gazebo Environment

Figure 6.3: Simulated Mars Surface

The height map (a) and color map (b) are combined in the Gazebo simulator (c) to model a Mars environment. The robot captures the simulated environment to generate an occupancy grid map.

6.2.3 Mars Results

The simulation is run twice. Case 1 is as described in this paper. Case 2 is identical to Case 1, except the reduced map m_{reduced} is used when computing (6.3). Case 2 is largely inspired by the promising experimental results covered in Section 7.3, where a projected map based on the same criteria as (6.5) proved effective for level flight. The resulting occupancy grid maps for Cases 1 and 2 are shown in Figures 6.4 and 6.5, respectively. A video of Case 1 can be found at https://youtu.be/FrBcL2UMW9w, and close-up pictures from Case 2 are shown in Figure 6.6. The complete map entropies for both cases are illustrated in Figure 6.7.

In both cases, the robot built the 3D probabilistic map of the 2000 cubic meter space composed of 4.74×10^6 grid cells. The maps were mostly complete within 10 minutes. The $\tilde{\theta} = 30^\circ$ downward viewing angle captured the ground nicely, as candidate attitudes tended to direct the onboard sensor toward uncertain regions on the surface of Mars. The proposed approach provided collision-free mapping and autonomous exploration in real-time.

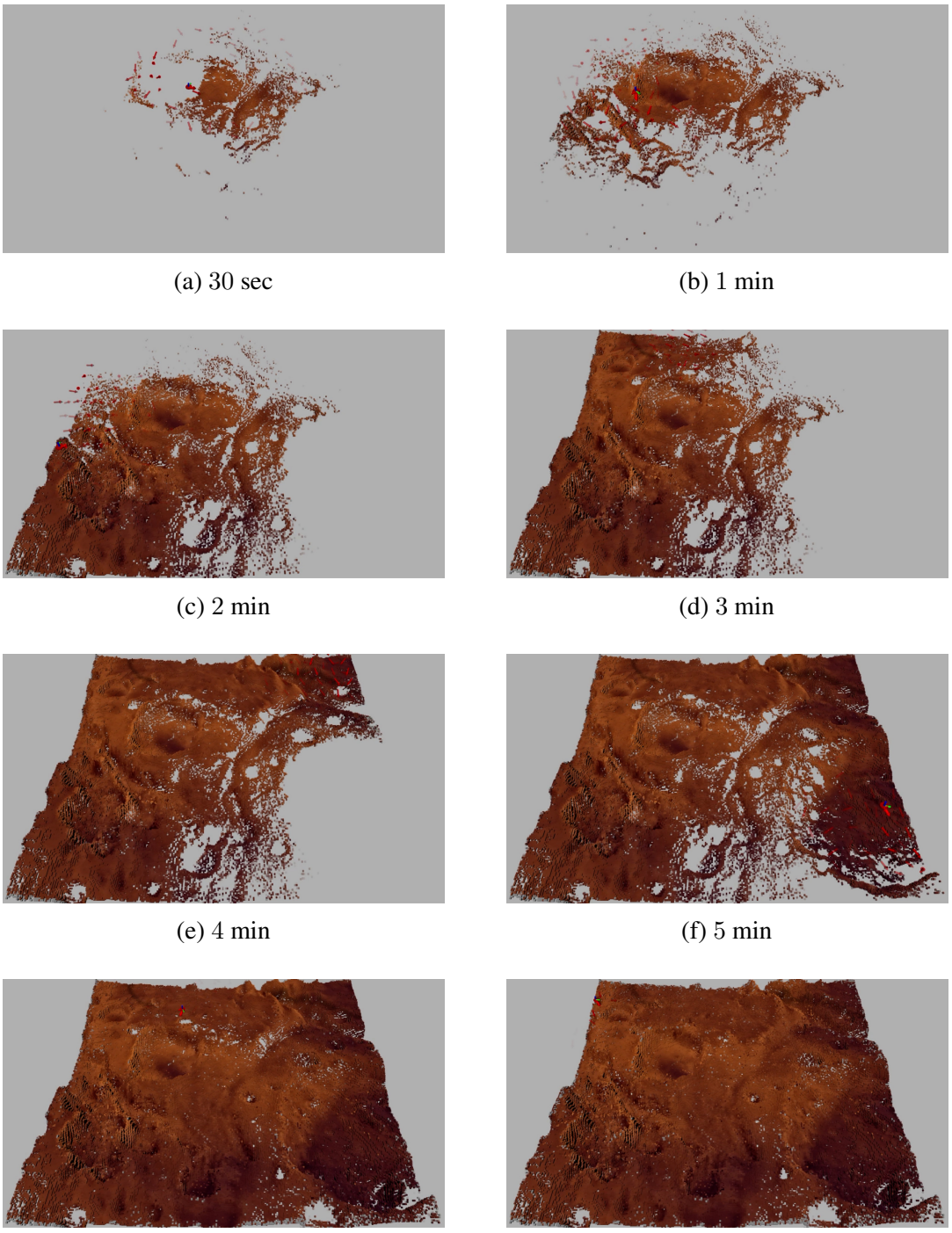
However, the choice of occupancy grid map used for entropy predictions introduces an interesting tradeoff. In Case 1, when the full probabilistic map m is used in (6.3), regions where the grid cells are partially-known are frequently reconsidered until the space is well-known. Conversely in Case 2, when m_{reduced} is used instead, the robot repeatedly leaves

spaces that are missing a few grid cells to visit new terrain. This is because m_{reduced} contains some cells with large occupancy probabilities based on (6.5), which allows some grid cells of m enclosed within a cell of m_{reduced} to be uncertain. The exploration policy of Case 2 incorrectly assumes these regions are well-known. Ironically, this false assumption can actually lead to greater information gains when the vehicle moves on to unvisited terrain. Conversely with Case 1, which uses m for computing expected entropy, the total map entropy decreases more steadily and generates a more-complete 3D occupancy grid of the surface of Mars.

In short, the proposed 3D probabilistic occupancy grid mapping and autonomous exploration are simulated successfully in real-time over the surface of Mars using ROS and Gazebo. Choosing m for entropy predictions produces a more-complete map, but choosing m_{reduced} is sometimes beneficial for exploring new terrain faster, while forgoing map completeness.

6.3 Conclusions

This chapter covers autonomous exploration through a 3D environment, such that the robot can account for complex geometries in all dimension and explore with upward and downward motions. The expected information gains are calculated with predicting the entropy changes from sample rays in 3D space, and the bump function is applied using the cost from Dijkstra's search. An example where a robot maps and explores the surface of Mars demonstrates the efficacy of the approach.



(a) 30 sec (b) 1 min

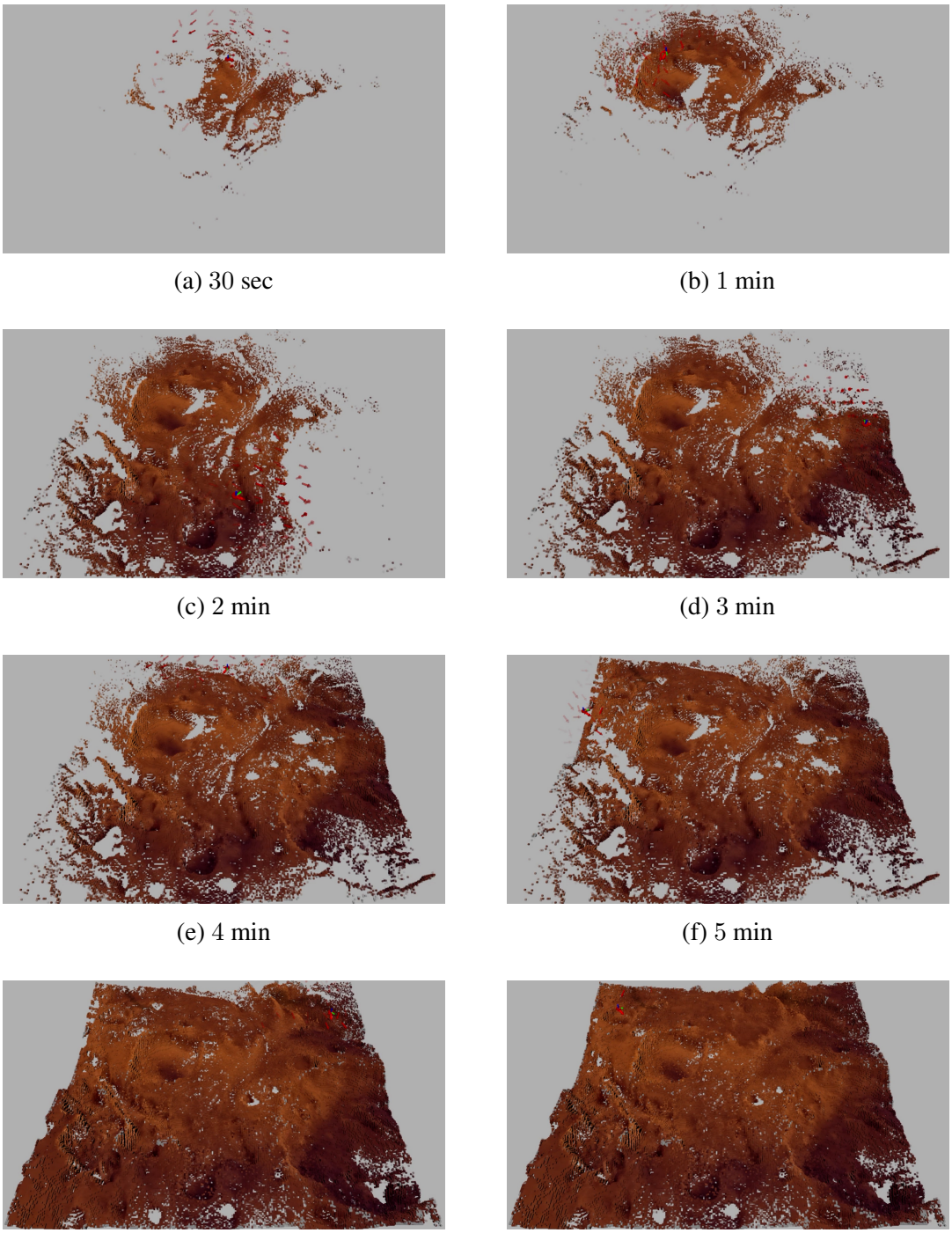
(c) 2 min (d) 3 min

(e) 4 min (f) 5 min

(g) 10 min (h) 15 min

Figure 6.4: Mars 3D Occupancy Grid Map for Case 1

In Case 1, the robot (red disk with arrow indicating laser direction) moves toward candidate poses (red arrows, more opaque for greater reward) based on expected entropy change of the 3D probabilistic occupancy grid map m (cubes: greater opacity for greater occupancy probability) of the surface of Mars.



(a) 30 sec

(b) 1 min

(c) 2 min

(d) 3 min

(e) 4 min

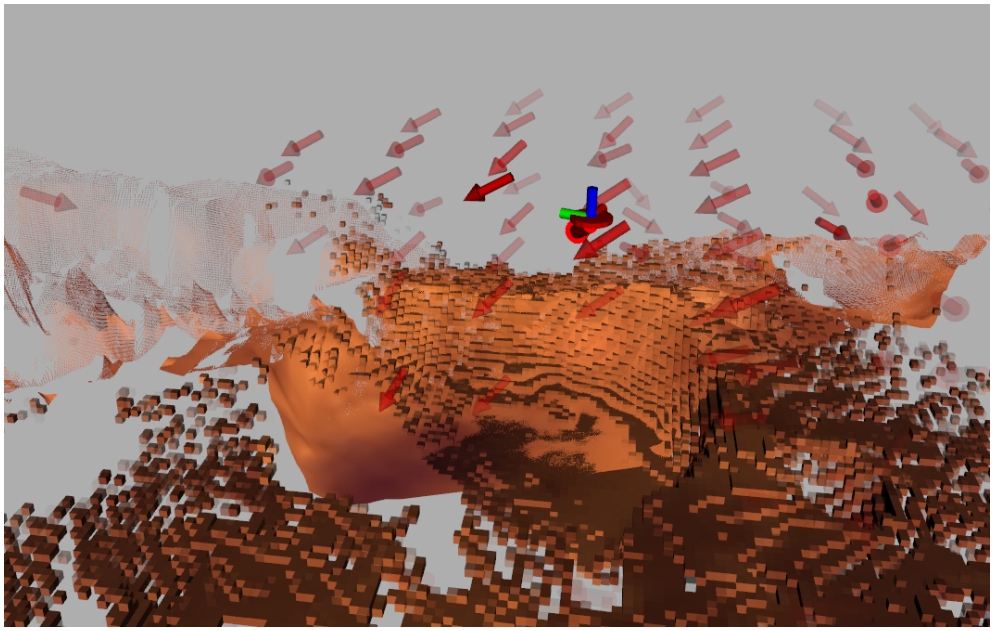
(f) 5 min

(g) 10 min

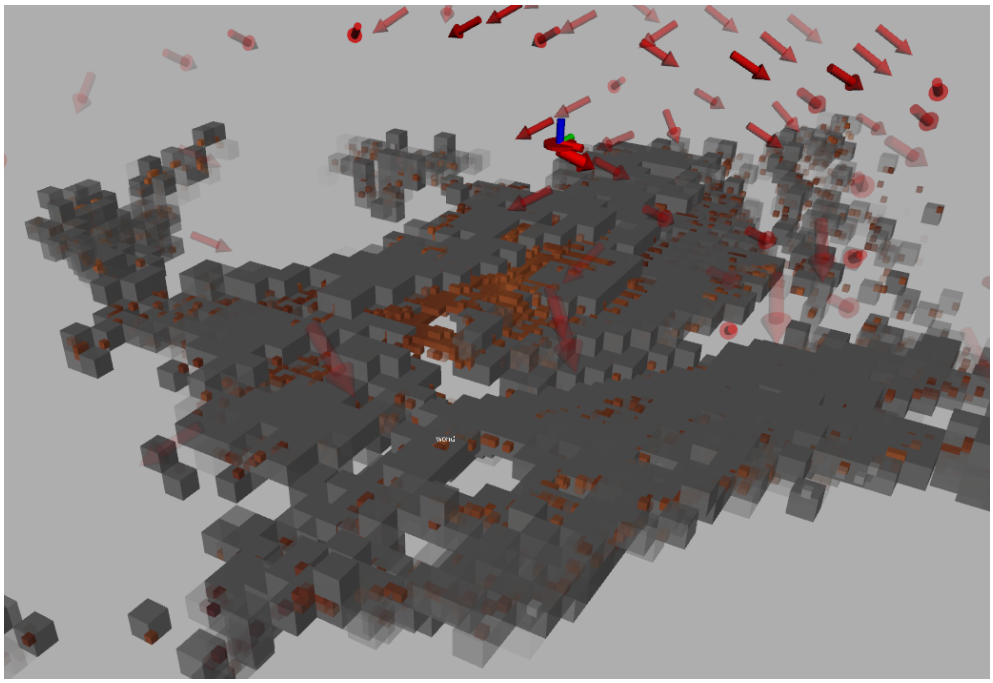
(h) 15 min

Figure 6.5: Mars 3D Occupancy Grid Map for Case 2

In Case 2, the robot (red disk with arrow indicating laser direction) generates a 3D probabilistic occupancy grid map m (cubes: greater opacity for greater occupancy probability), which is used to generate a reduced map m_{reduced} based on (6.5). The robot moves toward candidate poses (red arrows, more opaque for greater reward) based on expected entropy change of m_{reduced}. This approach leads to faster exploration of new terrain, but this leaves some grid cells missing.



(a) Full Map Cells and Sensor Scan



(b) Full Map and Reduced Map Cells

Figure 6.6: Zoomed-In Occupancy Grid Cells of Mars

The close-up images from the Case 2 trial show the candidate future poses (red arrows), where greater opacity represents a larger objective function of (6.6). In (a), we show the 3D scan with color corresponding to the surface of Mars. In (b), we overlay the full map m (colored) with the reduced map m_{reduced} (gray) from (6.5).

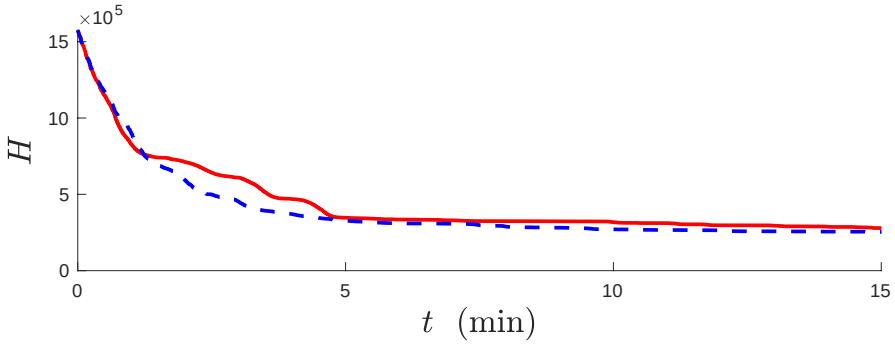


Figure 6.7: Entropy Histories During Mars Exploration

The complete map entropy for Case 1 (red solid line) decreases at roughly the same rate as the reduced map entropy for Case 2 (dashed blue line) toward the beginning. However, the expected entropy in Case 2 promotes actions toward unexplored territory, while the policy of Case 1 yields a more complete map before moving on to unexplored spaces.

Chapter 7: Experimental Results

In this chapter, we present three experiments that demonstrate the efficacy of the proposed exact occupancy grid mapping and autonomous exploration from earlier chapters. The first example involves a ground vehicle mapping and exploring a 2D space. The second example is a quadrotor unmanned aerial vehicle (UAV) flying around a vertically-uniform space while generating a 3D map from level flight. The final example shows autonomous exploration completely in 3D, where a quadrotor must map and explore around complicated objects.

7.1 2D Exploration With an Unmanned Ground Vehicle

The experiment involves a Pioneer ground vehicle producing a probabilistic occupancy grid map in real time. The robot motion is governed by the information gain-maximizing policy described in Chapter 3.

7.1.1 Hardware Configuration

The Pioneer 3 ground robot (Figure 7.1) is chosen for this experiment because of its reliability and simple operation. The vehicle accepts two inputs, namely linear velocity (aligned with the robot wheels) and angular velocity (about a central axis passing vertically through the robot). The robot pose is estimated via Vicon Tracker, which provides the location and attitude of a rigid body. The depth readings are obtained by a Kinect depth sensor. Since the experiment is to map and explore a 2D environment, only a single central row of the 3D Kinect depth scan provides the necessary measurements. The robot captures an experimental environment with walls constructed from Styrofoam.

(a) Pioneer 3 Vehicle

(b) Side View

(c) Front View with Kinect

Figure 7.1: Pioneer Robot

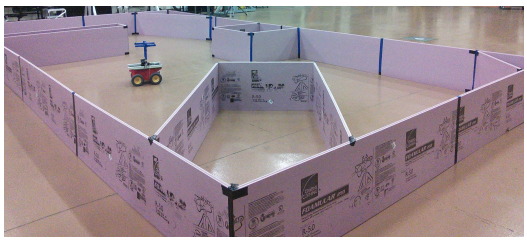
The Pioneer 3 robot serves as a reliable experimental platform for 2D mapping and autonomous exploration using a Microsoft Kinect.

7.1.2 Software Configuration

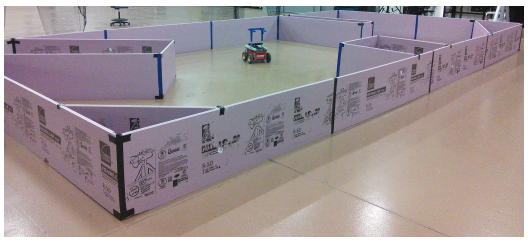
The software structure is similar to ROS nodes Section 3.3, which simply involves a mapping node, an exploration node, and a visualization node using OpenGL libraries [3]. The only differences are eliminating the simulator (Stage) and adding synchronization among the sensors, which is of high importance to the proposed occupancy grid mapping approach. Any time delay between the robot pose estimation and the sensor depth readings can cause conflicting information, which may be harmful to the probabilistic map. Thus, ROS Message Filters are applied such that the pose from Vicon Tracker (100 Hz) and the Microsoft Kinect (30 Hz) are nearly synchronized, which provides validity of the assumption that the measurement ray positions and directions are known deterministically.

7.1.3 Exploring and Mapping a 2D Environment

The environment consists of Styrofoam walls on a flat floor. The perimeter is rectangular, with several obstacles and angled walls (see Figure 7.2). The robot autonomously explored the space using the complete Cartesian searching approach, and Dijkstra's algorithm provides collision-free waypoints to the optimal future poses. Based on these waypoints, a constrained least squares polynomial fitting generates a smooth trajectory to follow the waypoints with fixed velocity. Finally, a simple controller is implemented such that the Pioneer follows the



(a) Bottom-Left View



(b) Bottom-Right View

Figure 7.2: 2D Experimental Environment

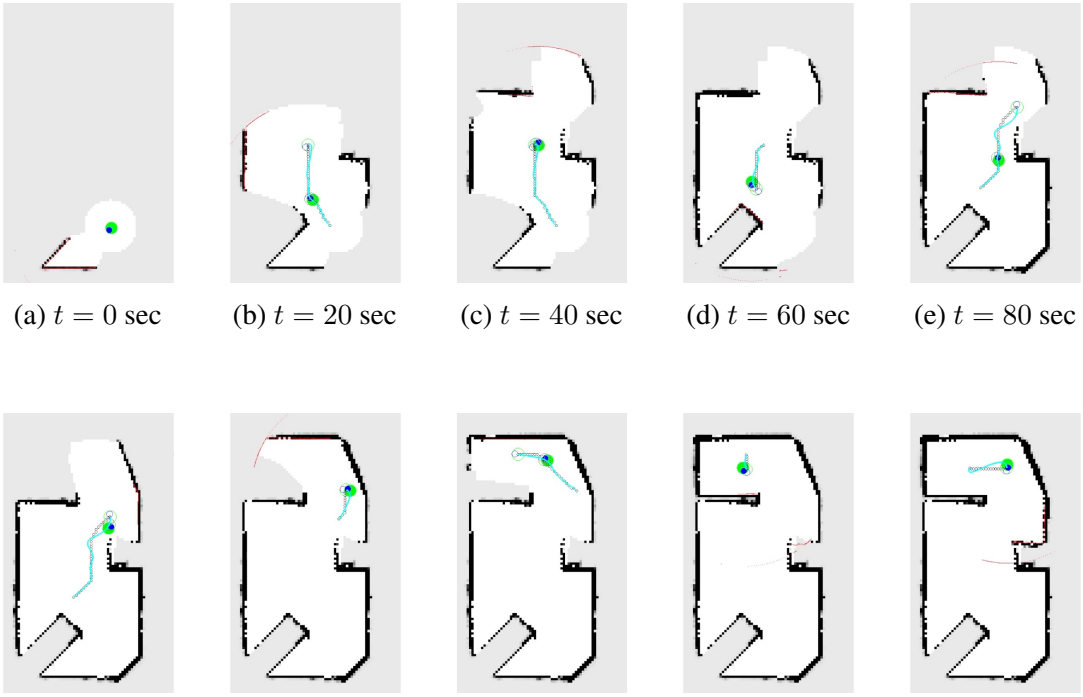
Images from two perspectives show the walls and obstacles of the experimental environment.

trajectory by moving and orienting toward the desired robot location 1 sec in the future.

The resulting occupancy grid maps and trajectories are depicted in Figure 7.3 and a video is available at www.youtube.com/watch?v=CRQfhhICSj0&feature=youtu.be. Most importantly, the robot builds a clear occupancy grid map despite sensor noise and imperfect sensor synchronization. The autonomous exploration successfully guides the robot such that it builds a map of the reachable space without colliding with any obstacles.

The resulting map and exploration commands are generated in real time. The grid cells are stored as double-float variables in a vector, where the vector index is mapped to a location on the occupancy grid. Thus, the memory requirements are proportional to the number of grid cells, which is roughly the mapping area divided by the area of a grid cell. Since cell locations need not be saved because the mapping between cell index and location is known, memory requirements are reduced. On a Lenovo T540p laptop with an Intel Core i7-4900MQ processor (quad-core, 2.8GHz per core) and 16GB of RAM, the mean time for updating the occupancy grid is 0.0115 seconds, and the mean time to determine an exploration strategy is 0.6820 seconds. Time requirements for mapping and exploration are easily modified by changing grid cell resolution or the number of exploration pose candidates, respectively. In short, there is a tradeoff between computational speed and accuracy of mapping or exploration.

The autonomous exploration is governed by a policy to maximize expected entropy decreases, so entropy and entropy change with time (Figure 7.4) are important metrics. In



(a) $t = 0$ sec (b) $t = 20$ sec (c) $t = 40$ sec (d) $t = 60$ sec (e) $t = 80$ sec

(f) $t = 100$ sec (g) $t = 120$ sec (h) $t = 140$ sec (i) $t = 160$ sec (j) $t = 180$ sec

Figure 7.3: Experimental 2D Occupancy Grid Map Results

A robot autonomously explores the experimental environment and produces an occupancy grid map along the way. The robot (shaded green circle body with shaded blue circle sensor) is controlled toward the desired pose 1 sec in the future (unshaded gray robot), which is following a constrained least-squares trajectory (cyan curve). This trajectory is based on waypoints (black circles) from Dijkstra's algorithm to arrive at the optimal pose (unshaded green circle body with unshaded blue circle sensor) as proposed in this paper.

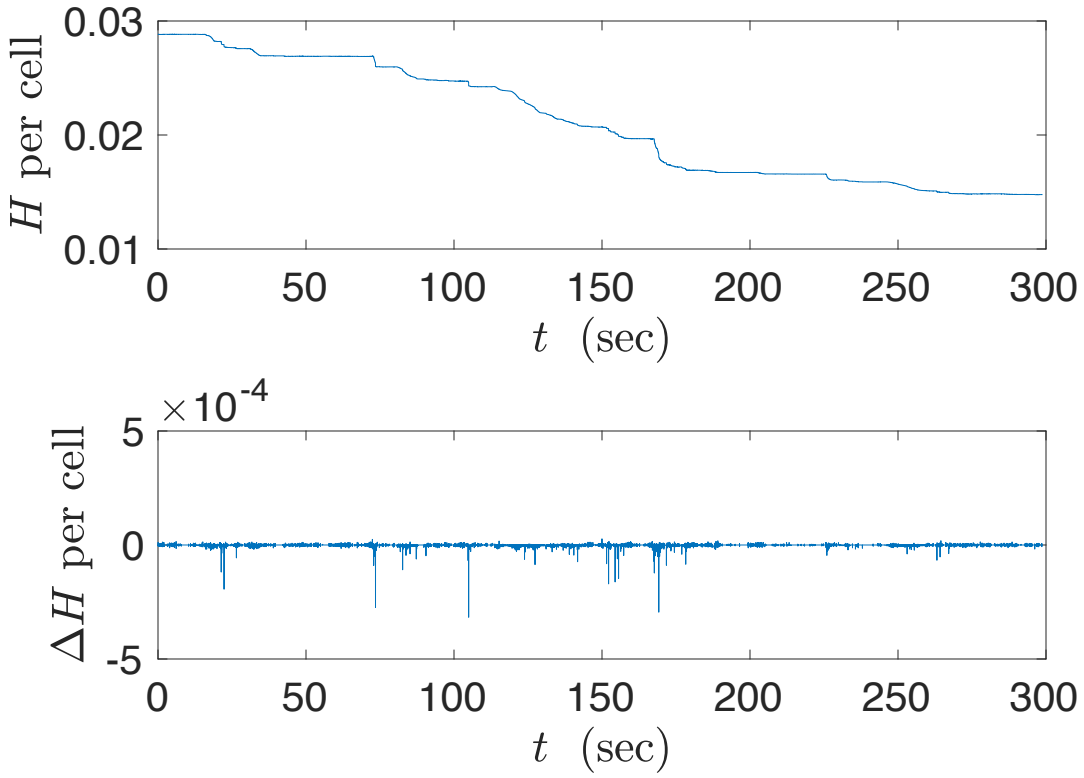


Figure 7.4: Entropy Change During 2D Experiment

The map entropy H averaged over all $90,601$ grid cells is reduced during experimentation. The quick changes ΔH per cell correspond to map updates when the robot captures new territory.

this example, the a priori occupancy probabilities of all cells are 0.1, so the entropies could temporarily rise for likely occupied cells. The total map entropy begins at roughly 2613 and decreases by roughly half to 1338. Many grid cells within the map limits are occluded by walls, preventing the blocked cells from occupancy probability changes. In some cases, the robot is able to update the cell occupancy probabilities; in other cases, cells fall inside regions occluded from all reachable space, and the robot cannot update these cells. Among those grid cells visible from collision-free locations, the vast majority of cells are updated in less than 3 min while following a very slow (5 cm/sec) trajectory.

In conclusion, this example shows how occupancy grid mapping and autonomous exploration are applied to a ground vehicle in a 2D environment such as a floor of a house

or building. Dijkstra's search is applied about the occupancy grid, and a smooth trajectory is easily tracked with a simple controller that tracks a polynomial least squares trajectory. These results show the efficacy of exact occupancy grid mapping and autonomous exploration. Next, we extend these experiments to 3D.

7.2 Geometric Control

The next two experiments involve a quadrotor unmanned aerial vehicle (UAV) mapping and exploring 3D environments. A flight controller that accounts for the nonlinear quadrotor attitude dynamics is required to track the trajectories provided by autonomous exploration. We choose geometric control because of its stability properties and simple form.

Geometric control is designed for rigid-body attitude dynamics, and is developed directly on $\mathsf{SO}(3)$ without local parameterizations such as Euler angles or quaternions. Therefore, this approach avoids singularities, ambiguities, and linearization errors. It has been developed for fully-actuated vehicles [29] and for under-actuated quadrotors [42, 41, 40, 21]. This approach is further extended to aerial load transportation [22, 23] and attitude estimation [61].

In the context of quadrotor control, the main idea is that a translational position-integral-derivative (PID) control produces a desired force in 3D, and the robot selects a desired attitude R_d and angular velocity Ω_d to align its propellers with this force and to correct its first body-fixed axis to a desired direction. The attitude and angular velocity errors from the current robot attitude R and angular velocity Ω are

$$e_R = (R_d^\mathsf{T} R - R^\mathsf{T} R_d)^\vee, \tag{7.1}$$

$$e_\Omega = \Omega - R^\mathsf{T} R_d \Omega_d, \tag{7.2}$$

respectively, where the *vee* map is defined in [40]. The quadrotor is capable of producing a

force $f \in \mathbb{R}$ and moment $M \in \mathbb{R}^3$,

$$f = (k_x e_x + k_v e_v + mge_3 - m\ddot{x}_d) \cdot Re_3, \tag{7.3}$$

$$M = -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega - J(\hat{\Omega} R^\mathsf{T} R_d \Omega_d - R^\mathsf{T} R_d \dot{\Omega}_d), \tag{7.4}$$

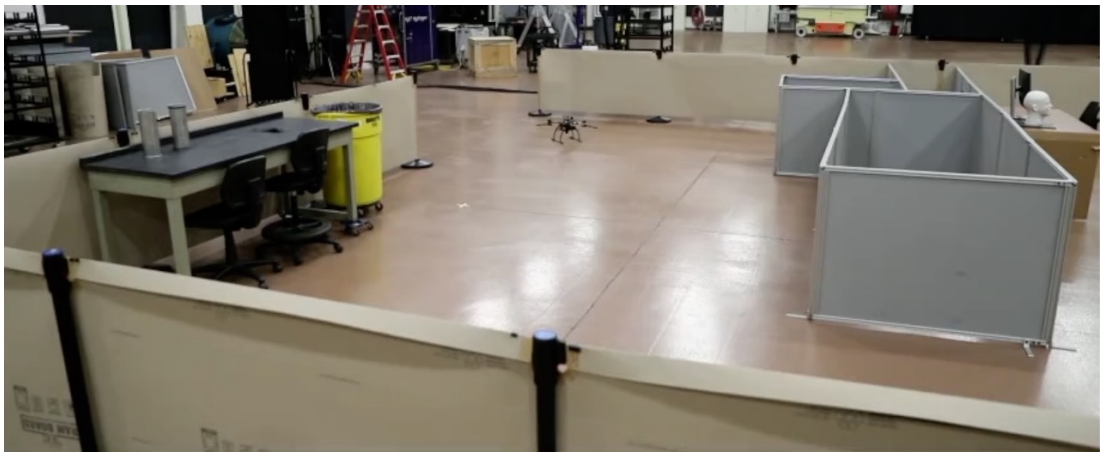
where e_x and e_v are position and velocity errors, respectively, k_x, k_v, k_R, k_Ω are positive gain constants, $\ddot{x}_d$ is the desired translational acceleration, $\dot{\Omega}_d$ is the desired rotational acceleration, m is the quadrotor mass, J is the quadrotor moment of inertia, $e_3 = [0, 0, 1]^\mathsf{T}$, and the *hat* map is defined in [40]. Since (7.3) and (7.4) require only trivial calculations, they are easily computed with onboard modules.

7.3 Aerial Exploration of a Large Space

In this section, we present results from a quadrotor as it autonomously explores a nearly vertically-uniform room. A robots builds a 3D map of its environment, which is projected onto a 2D map for exploration following the process described in Section 4.2. The software structure and sensor parameters are identical to those presented in Section 4.3.

7.3.1 Exploration Environment

The U.S. Naval Research Laboratory (NRL) has a large experimental space for testing known as the Laboratory for Autonomous Systems Research (LASR). Here, the experimental setup contains walls and objects to resemble a building floor plan. The perimeter walls are constructed with tan cardboard, which were suspended from 1 m tall stanchions, and an internal wall is built with gray metal sheeting mounted to $80/20$ supports [4]. Additionally, a hard flooring reflects some florescent lighting above. The objects inside the experimental environment include various trash cans, desks, and chairs in addition to a few miscellaneous objects. The map size is selected to slightly exceed the volume of the experiment such that the x-direction (positive east) spans 0.5 m to 10.3 m, the y-direction (positive north) spans



(a) North Region

(b) South Region

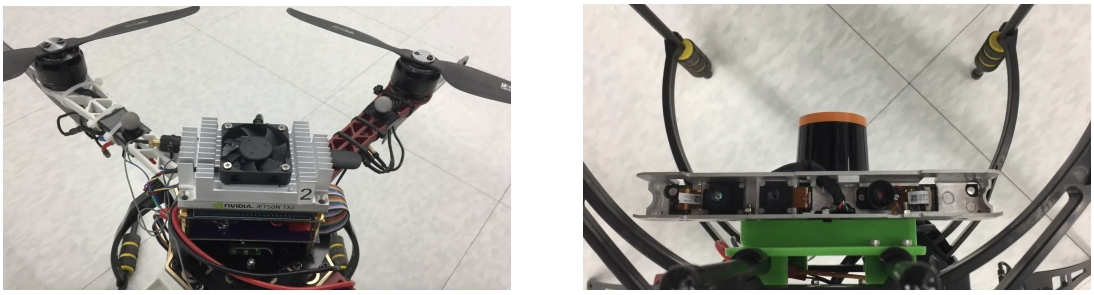
Figure 7.5: 3D Experimental Environment for Level Flight

The 3D experimental environment is designed to resemble a large room within an office building. The walls are built from from cardboard suspended from stanchions and gray sheeting with $80/20$ supports, and miscellaneous objects are placed around the room as well.

-8.3 m to 3.5 m, and the z-direction (positive vertically up) spans -0.15 m to 1.5 m. Two images of the experimental environment are displayed in Figure 7.5.

7.3.2 Hardware Structure

Several components are used for localization, sensing, and actuation. A Vicon motion capture system provides the transformation between a reference frame fixed to the world and



(a) Quadrotor Top (b) Quadrotor Bottom

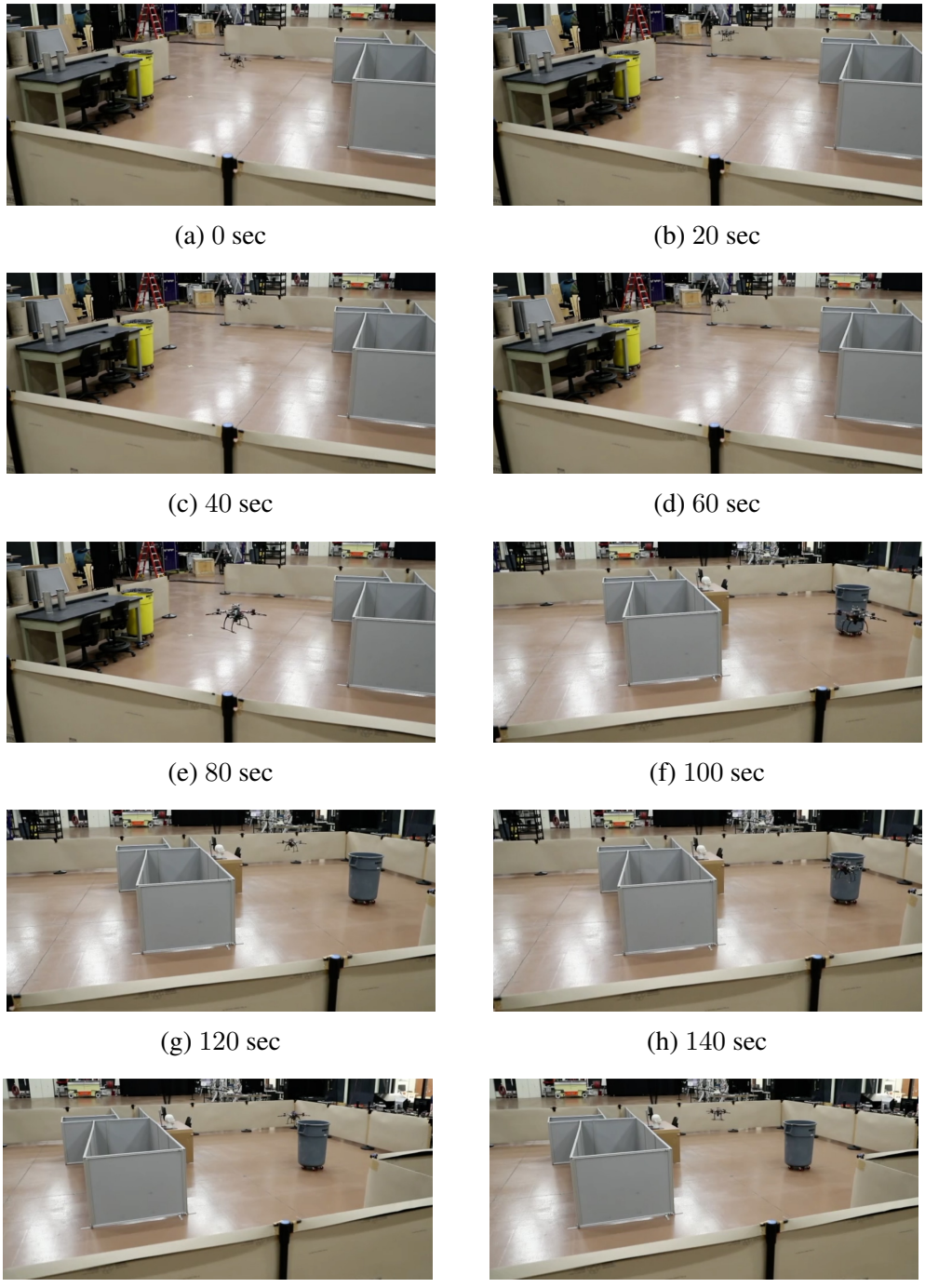
Figure 7.6: Quadrotor Platform for 3D Mapping and Exploration

A Jetson TX2 (quadrotor top) streams depth measurements via WIFI from an Asus Xtion and Hokuyo LIDAR (quadrotor bottom). The host computer builds a 3D probabilistic occupancy grid map from these measurements. For autonomous exploration, the host computer designs exploration trajectories, which an onboard flight controller can track in real-time. This platform is used for both level flight (Section 7.3) and non-level flight (Section 7.4).

the moving vehicle. This transformation, along with fixed transformations between the robot body and the sensors, provides sufficient information for the transformation from the world to each sensor frame. The sensor readings from the Xtion and Hokuyo are transmitted from an NVidia Jetson TX2 onboard the vehicle (Figure 7.6). The sensor readings are received via WIFI on a host computer with an Intel Core i7-6800K CPU (12×3.40GHz), which runs the mapping and exploration nodes. The host computer returns the exploration trajectories to the Jetson TX2 over WIFI. Then, the Jetson executes the geometric flight controller onboard. With this configuration, large processing tasks are avoided onboard the robot.

7.3.3 Experimental Results

The quadrotor takes off, completes a rotation to understand its immediate surroundings, then explores the space over 2 minutes and 47 seconds, shown in Figure 7.7. Figure 7.8 shows the probabilistic 3D map generated in real-time at four selected times, and the corresponding 2D projected maps (x-direction upward) are illustrated in Figure 7.9, where arrows represent pose candidates and a trajectory is shown to the optimal pose. Figure 7.10 shows the total map entropy decreasing over the experiment duration. A video of the experiment can be



(a) 0 sec (b) 20 sec

(c) 40 sec (d) 60 sec

(e) 80 sec (f) 100 sec

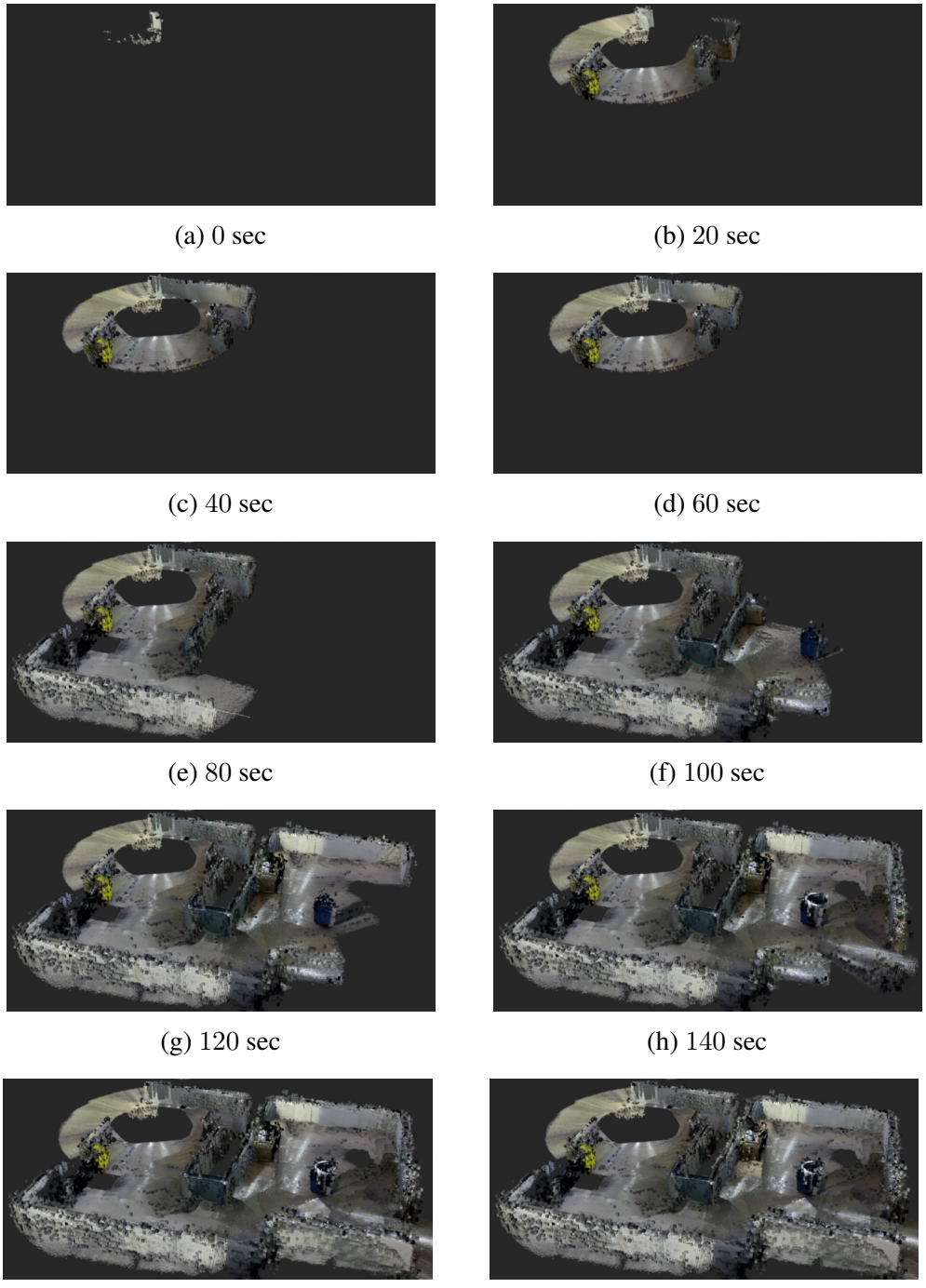
(g) 120 sec (h) 140 sec

(i) 160 sec (j) 167 sec (end)

Figure 7.7: Level Flight Experiment

During the first minute, the robot slowly takes off and does an initial rotation, then explores the space in the remaining time.

found at https://www.youtube.com/watch?v=I_1rXV2XRqk.



(a) 0 sec (b) 20 sec

(c) 40 sec (d) 60 sec

(e) 80 sec (f) 100 sec

(g) 120 sec (h) 140 sec

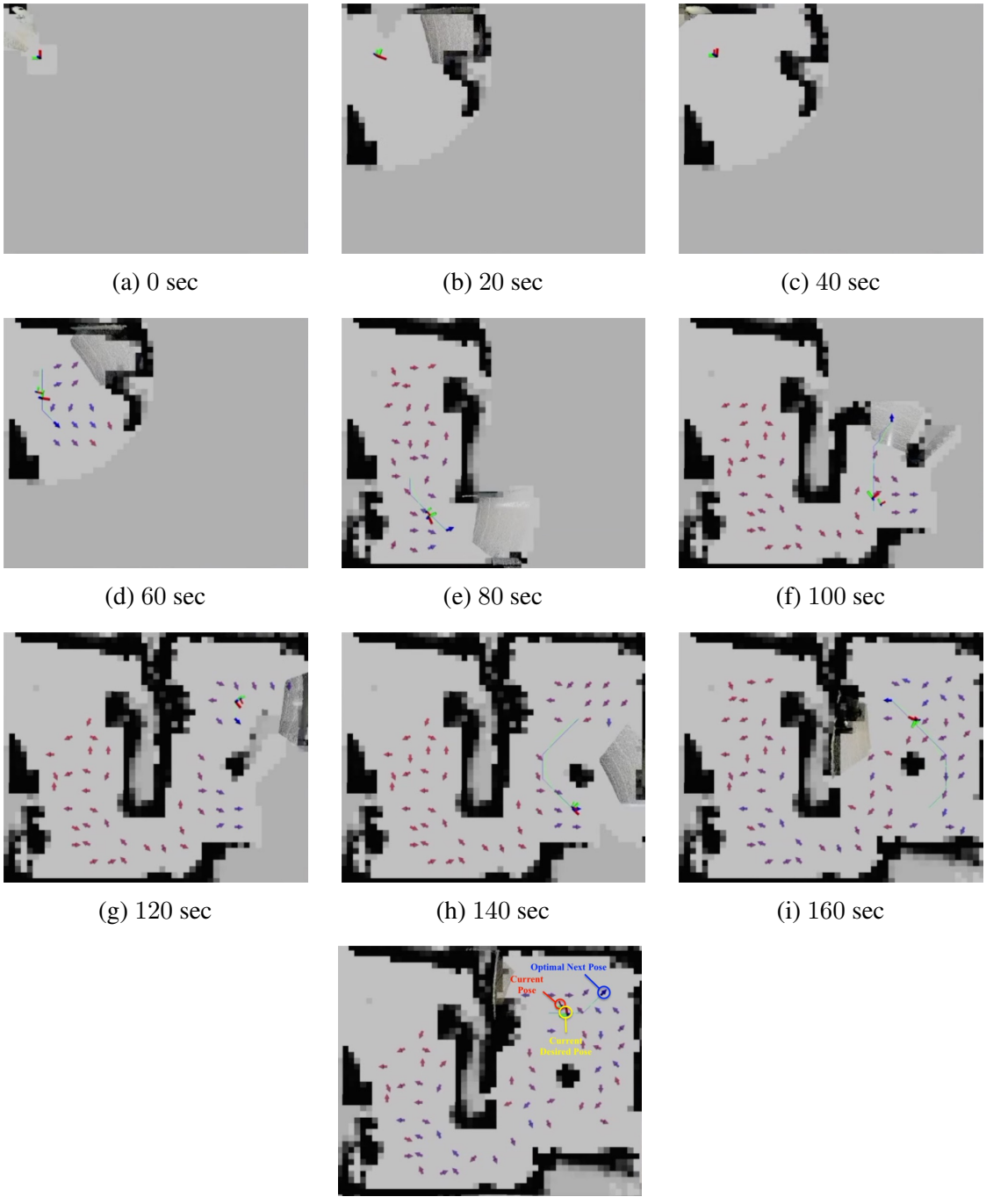
(i) 160 sec (j) 167 sec (end)

Figure 7.8: 3D Occupancy Grid Map During Level Flight Experiment

The 3D occupancy grid map is overlaid with point cloud measurements. The map probabilities directly rely on the Xtion and Hokuyo measurements and the sensor stochastic properties.

The robot is able to produce a 3D occupancy grid in under 3 minutes without a human providing a trajectory. The 3D map can be easily interpreted by a human, and provides sufficient information in real-time for autonomous exploration. Optimal poses and collision-free exploration trajectories are calculated by the exploration node in under 1 second. These quick decisions are vital to the autonomous exploration performance since the map is initially uncertain.

The projected map used for both collision and exploration has several benefits. The robot measures walls and objects with the 3D map, and nicely represents these as occupied spaces with the 2D projected map. Additionally, the exploration computation is low, allowing only for brief hover periods (less than 1 second) between trajectories. However, the robot exploration does not consider those cells far below the robot, particularly those near the ground. As a result, some floor regions are not captured by the 3D occupancy grid even though cells above the floor are well-known. These neglected cells on the floor motivate the full 3D exploration, shown in the next experiment.



(a) 0 sec (b) 20 sec (c) 40 sec

(d) 60 sec (e) 80 sec (f) 100 sec

(g) 120 sec (h) 140 sec (i) 160 sec

(j) 167 sec (end)

Figure 7.9: 2D Projected Occupancy Grid Map During Level Flight Experiment

The 2D projected map is easily produced in real-time. Candidate future poses, shown by arrows, have more blue color for larger objective functions, found with expected information gains and travel distances. Collision-free waypoints (blue line) serve as input to a constrained polynomial least-squares path (light green curve) for the robot controller to follow, where the desired pose (large axes) is tracked by the onboard controller from the current robot pose (small axes).

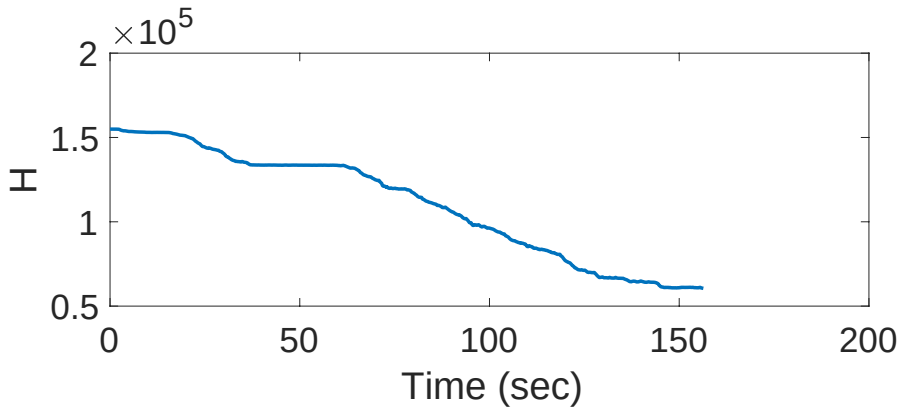


Figure 7.10: Entropy History During Level Flight Experiment

Map entropy decreases as the robot explores and captures more of the experimental environment.

(a) Northwest View

(b) Northeast View

Figure 7.11: 3D Exploration Space

The experimental space for full 3D exploration consists of normal lab objects such as tables and chairs, as well as a mannequin sitting at a desk and his parked bicycle.

7.4 Full 3D Exploration with a Quadrotor

In this final experiment, we introduce the quadrotor to a space with complex 3D objects. The robot generates a 3D map of this space, and performs autonomous exploration while considering the expected information gains of sample measurement rays with nonzero vertical components. Unlike prior examples, the robot also moves vertically during exploration and tracks the ground below the vehicle.

7.4.1 3D Exploration Setup and Parameters

The software follows the same structure as in Section 6.2, except the Gazebo simulated environment is replaced with experimental equipment. This equipment is identical to the prior experimental example (Section 7.3), except the experiment takes place at the Flight Dynamics and Control Lab (FDCL) inside the second floor of SEH, and the Hokuyo is no longer in use. This environment is smaller with only 6 cameras, where the x-direction (positive east) spans -4.5 m to 4.5 m, the y-direction (positive north) spans -4.0 m to 4.5 m, and the z-direction (positive vertically up) spans -0.25 m to 2.0 m, shown in Figure 7.11.

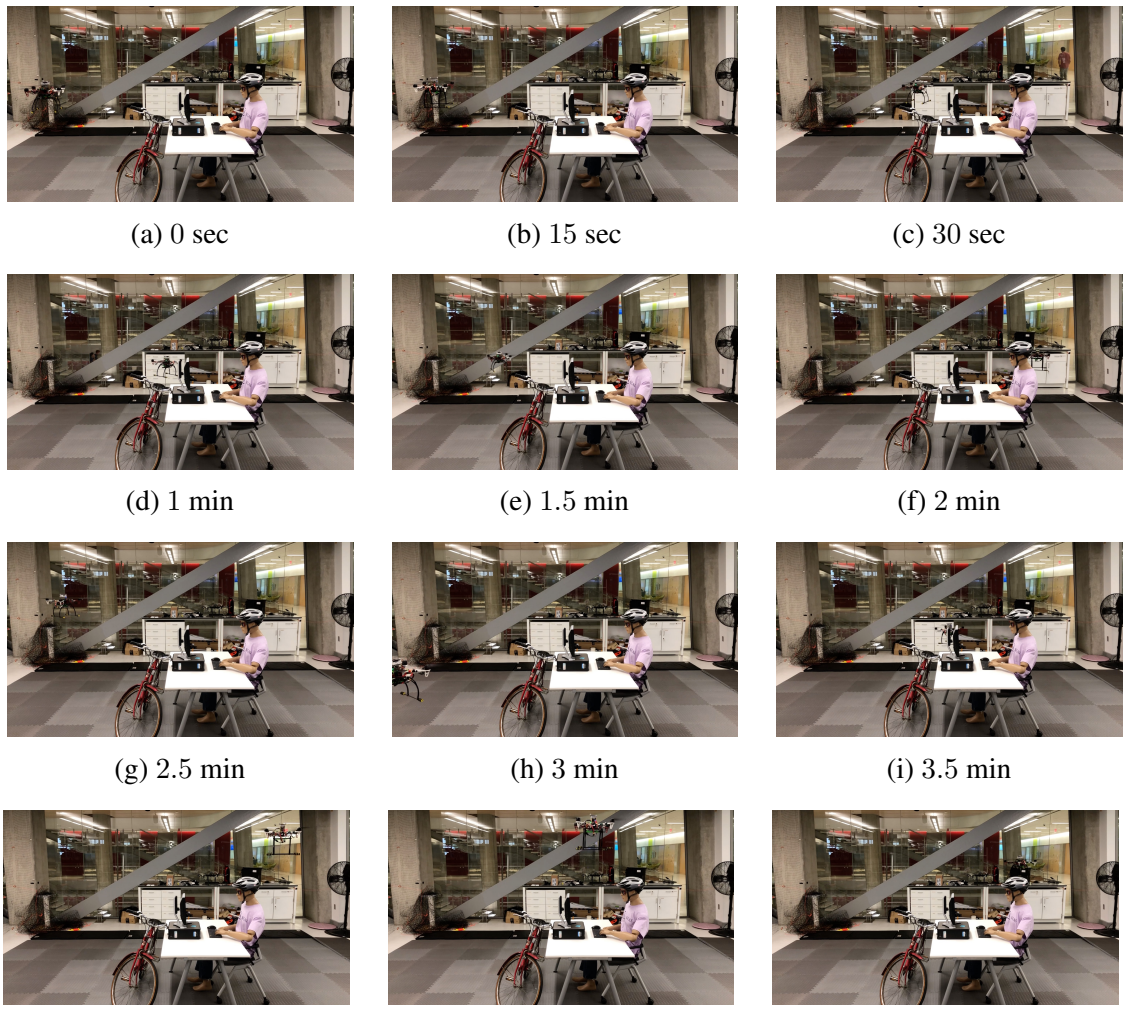
7.4.2 3D Exploration Results

The experimental results show that the robot successfully explores a 3D space, capturing occupied space above and below the robot. The quadrotor does an initial rotation over 20 sec, then explores the space with planned motions in all three dimensions, shown in Figure 7.12. The robot generates a 3D probabilistic map (Figure 7.13) and uses this for entropy prediction and collision-free motion planning with a 3D cost map (Figure 7.14). A video of this experiment is available at https://www.youtube.com/watch?v=2Q2_-d8kNu0.

The proposed 3D occupancy grid mapping and autonomous exploration approaches complement each other well. The floor and objects are well-captured, as this detail-oriented entropy-based exploration scheme tends to generate completely mapped regions before moving to new terrain, shown with Case 1 of the Mars exploration numerical example (Section 6.2). As a result, the entropy decreases fairly steadily over the experiment, shown in Figure 7.15.

A key improvement to prior experiments is the ability to consider the occupancy properties in full 3D. This allows the robot to understand that the mannequin occupies some space that the robot must avoid. However, the robot also recognizes that the space above the mannequin is free, and it chooses a trajectory above the mannequin to reach new terrain. At other times, the robot observes spaces from lower vantage points, which can be advantageous with a short sensor range and the forward direction of the Asus Xtion relative to the robot. These added capabilities are attributed to the full 3D mapping and exploration approach.

The controller is also highly important for following autonomous exploration trajectories accurately. The geometric controller tracks the desired trajectory to avoid collisions with obstacles in the room. The plot of 3D position is shown in Figure 7.16 and the desired commands and measured outputs of the controlled system are shown in Figures 7.17, 7.18, 7.19, and 7.20, and the total propeller force f from (7.3) and moment M from (7.4) are shown in Figure 7.21. An interesting aspect is that at 1.31 min, the robot temporarily obtains a poor estimate of the robot, but since this is short-lived, the robot quickly recovers



(a) 0 sec (b) 15 sec (c) 30 sec

(d) 1 min (e) 1.5 min (f) 2 min

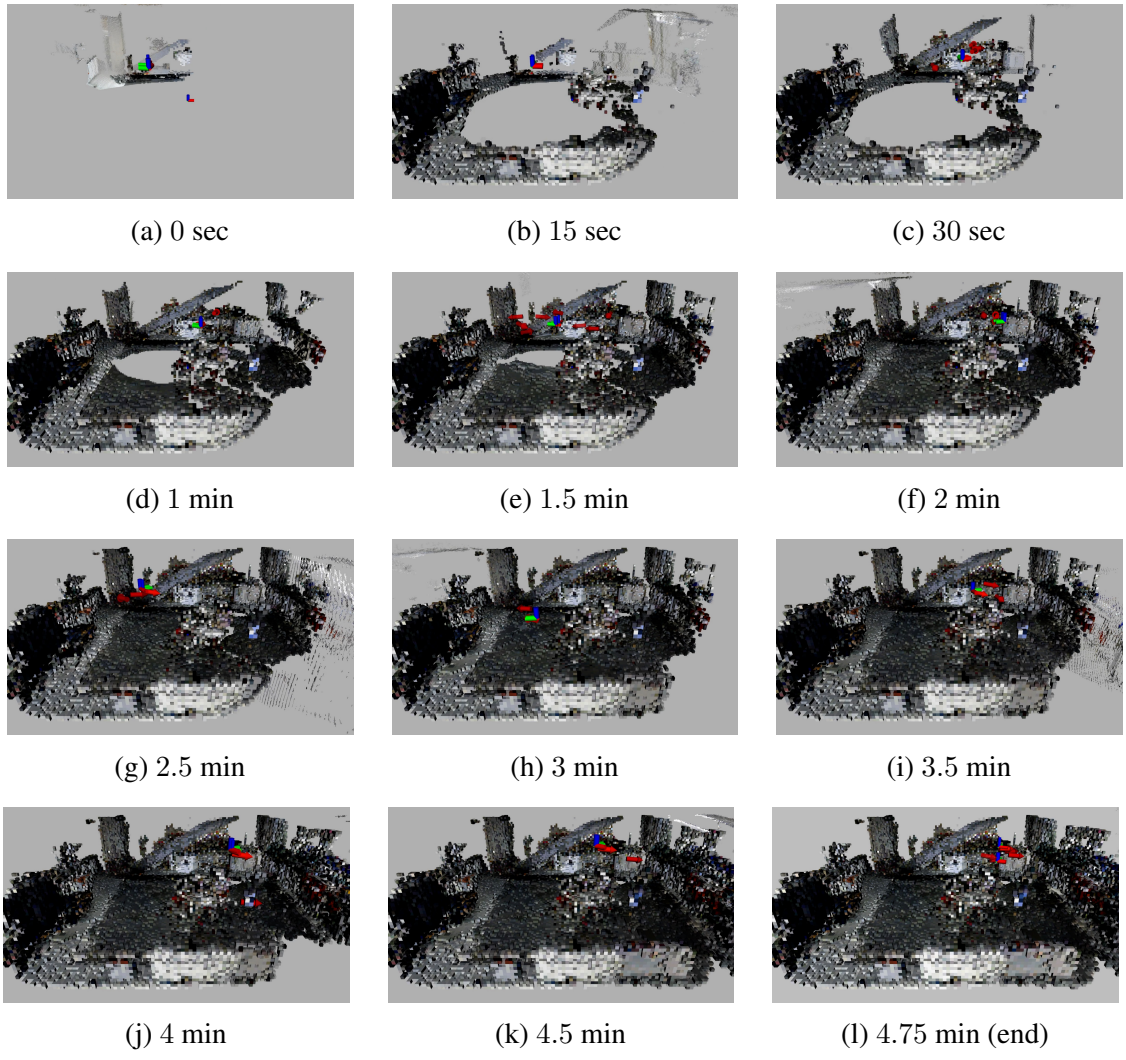
(g) 2.5 min (h) 3 min (i) 3.5 min

(j) 4 min (k) 4.5 min (l) 4.75 min (end)

Figure 7.12: Full 3D Autonomous Exploration Experiment

The robot completes a full yaw rotation over 20 seconds, then autonomously explores the space. At 4 min (j), the robot flies over the mannequin to explore the southeast corner of the room.

and continues exploring. In summary, geometric control successfully tracks the desired commands from autonomous exploration through uncertain environments.



(a) 0 sec (b) 15 sec (c) 30 sec

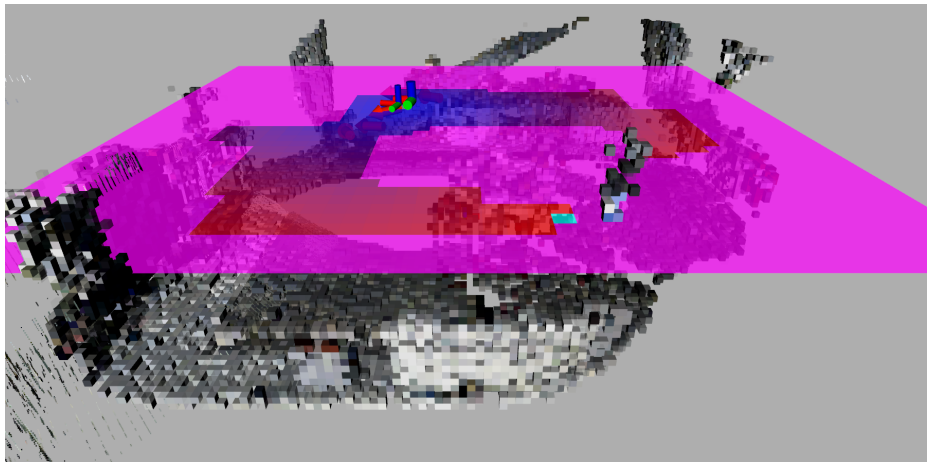
(d) 1 min (e) 1.5 min (f) 2 min

(g) 2.5 min (h) 3 min (i) 3.5 min

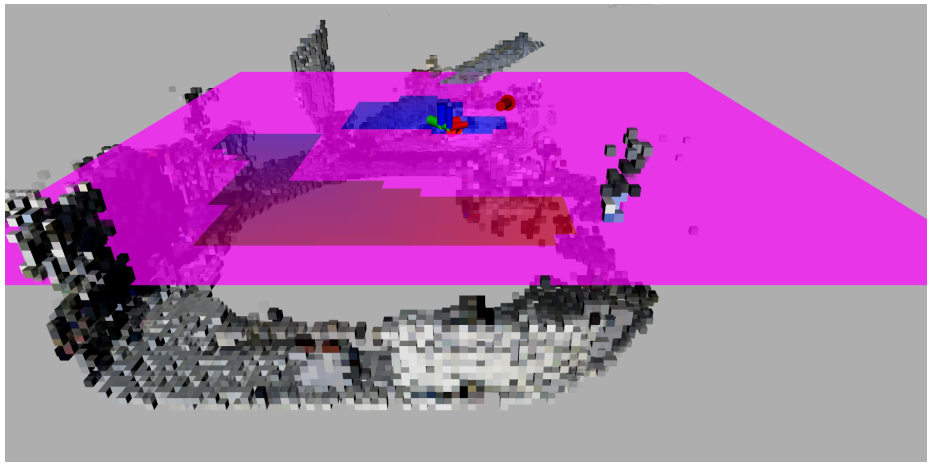
(j) 4 min (k) 4.5 min (l) 4.75 min (end)

Figure 7.13: 3D Occupancy Grid Map During Full 3D Autonomous Exploration

The robot (large axes) is controlled to track the desired pose (small axes), which follows a collision-free trajectory using Dijkstra's search toward candidates (top 75% shown with red arrows, with greater opacity for larger expected information gains).



(a) Connected Cost Map Cross-Section



(b) Disconnected Cost Map Cross-Section

Figure 7.14: 2D Cross-Sections of 3D Cost Maps

Two examples of the 3D cost maps from Dijkstra's search show connected and disconnected regions at the exploration height, where more blue regions have smaller distance costs, red regions have larger distance costs, and pink regions are unreachable due to collision risk. In (a), the robot does not have to change altitude to reach any of the collision-free poses at its current altitude. However in (b), the robot can only reach collision-free poses in the disconnected section by changing altitude and following the 3D cost map.

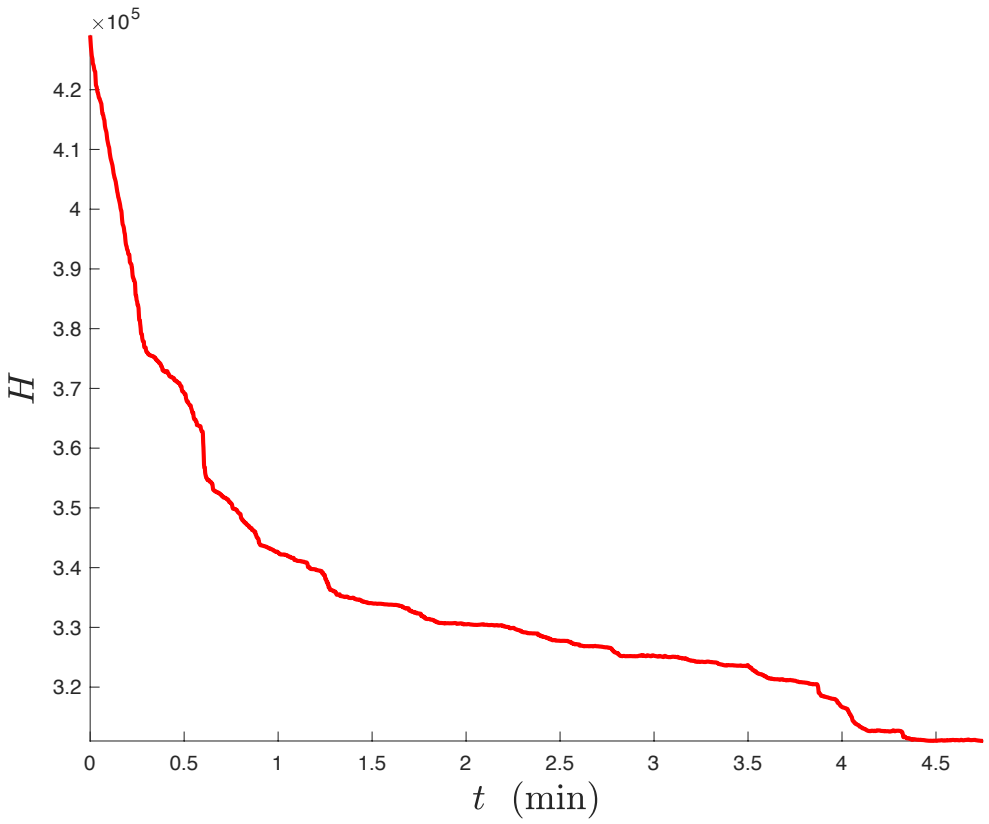


Figure 7.15: Entropy Change During Full 3D Experiment

The map entropy H consistently decreases during the full 3D exploration experiment. Toward the end of the experiment, most cells that can be captured by the Asus Xtion have been updated, slowing down the rate of entropy decrease.

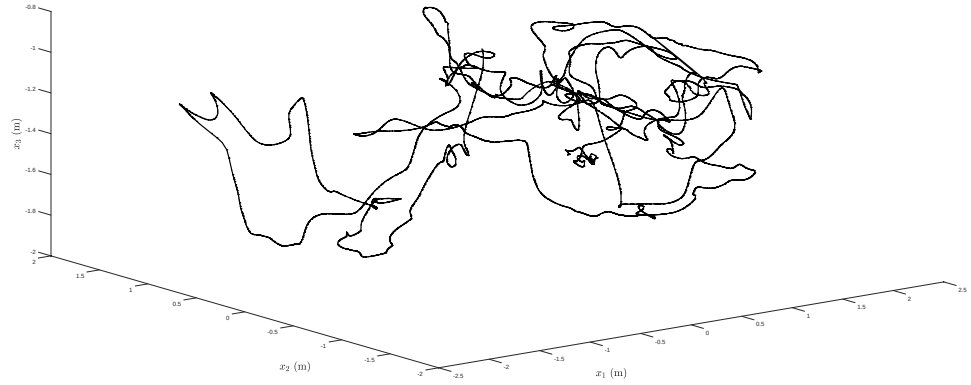


Figure 7.16: Robot Position over Full 3D Experiment

The history of robot position is shown in 3D with respect to a north-east-down world-fixed frame, such that there are all negative x_3 values, making the trajectory appear upside-down. The low locations (high relative to the ground) on the left of the figure ($x_1 \approx -1$ m, $x_2 \approx 0.5$ m, and $x_3 \approx -1.8$ m) correspond to flying over the mannequin.

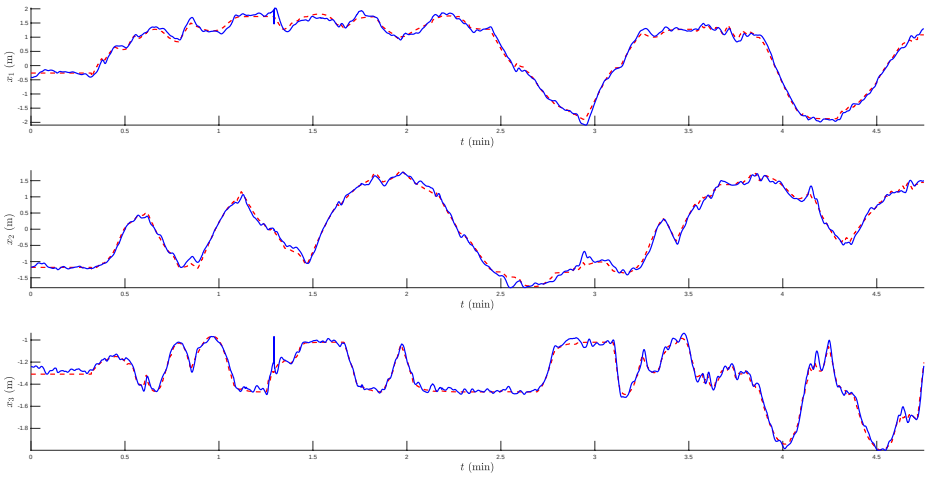


Figure 7.17: Geometric Control Position Tracking over Full 3D Experiment

The position of the robot (blue solid) tracks the desired position (red dashed) over the experiment.

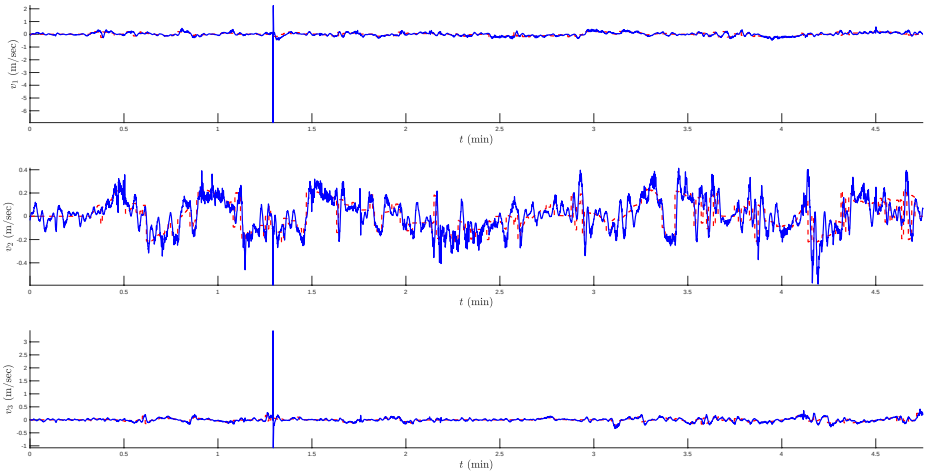


Figure 7.18: Geometric Control Velocity Tracking over Full 3D Experiment

The velocity of the robot (blue solid) tracks the desired velocity (red dashed) over the experiment.

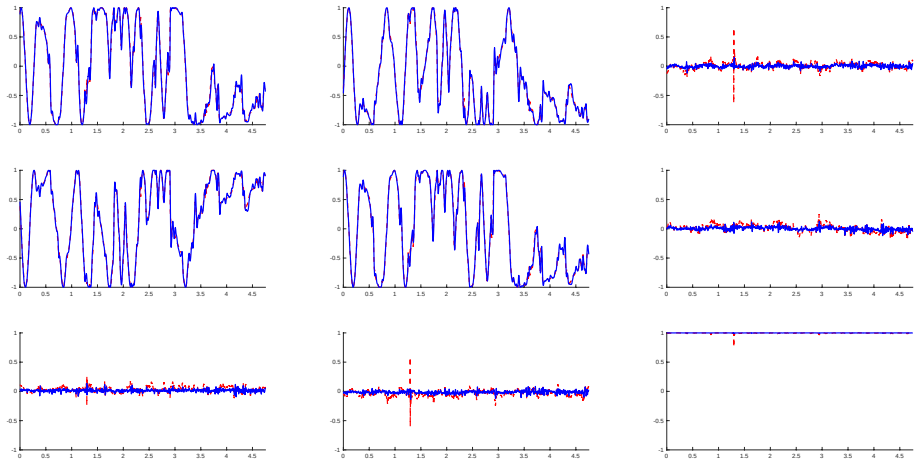


Figure 7.19: Geometric Control Attitude Tracking over Full 3D Experiment

The attitude of the robot (blue solid), shown with the rotation matrix between the north-east-down frame and the body-fixed frame, tracks the desired attitude (red dashed) over the experiment. The bottom-right plot shows the third body-fixed axis is mostly-aligned with the third world-fixed axis, which corresponds to maintaining level flight.

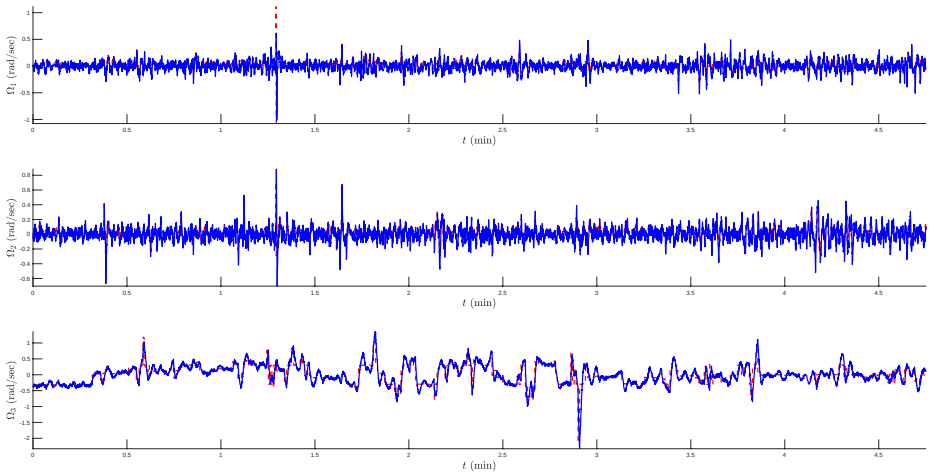


Figure 7.20: Geometric Control Angular Velocity Tracking over Full 3D Experiment

The angular velocity of the robot (blue solid) tracks the desired angular velocity (red dashed) over the experiment. The overall larger Ω_3 corresponds to yaw rotation to change the direction of the Xtion depth sensor.

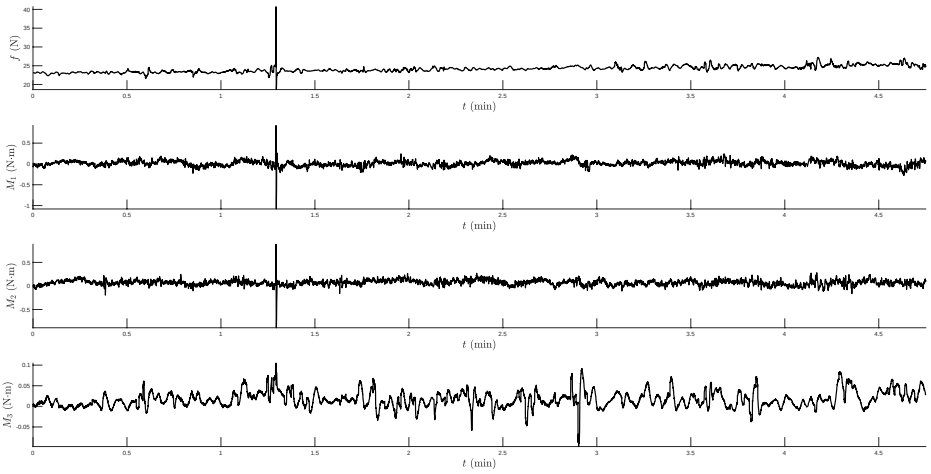


Figure 7.21: Geometric Control Force and Moments over Full 3D Experiment

The forces and moments provided by the geometric control are shown over the experiment. The values of f remain largely-steady to cancel gravitational forces. Changes in M_1 and M_2 correspond to roll and pitch movements to move around horizontally. Changes in M_3 correspond to correcting yaw for the Xtion direction.

Chapter 8: Conclusions

8.1 Summary of Contributions

In this dissertation, we propose several contributions in probabilistic occupancy grid mapping and autonomous exploration in 2D and 3D environments in single- and multi-vehicle scenarios. Computational efficiency and numerical accuracy are held paramount in derivations and algorithmic structure. Several numerical examples and experiments demonstrate the efficacy of each contribution.

The primary contribution is an exact and efficient solution to probabilistic occupancy grid mapping. Previously believed too computationally expensive, the exact solution can now be applied in real-time to 2D and 3D environments. The proposed novel Bayesian solution, referred to as the inverse sensor model, relies on grouping several possible map combinations with identical forward sensor models, rather than considering each map combination individually. This approach exploits properties of occupancy grid maps, and follows a recursive algorithm that avoids repeated calculations efficiently.

The solution to probabilistic occupancy grid mapping naturally extends to predicting future maps. The proposed autonomous exploration scheme capitalizes on map predictions using Shannon's entropy as an uncertainty measure. With the novel proposed approach, probabilities of future map measurements can be predicted, which can be used in conjunction with the inverse sensor model to predict future map entropy. Robots select poses to minimize expected entropy, equivalently maximizing map information gain. The algorithm is designed to be applied in real-time such that robots can handle uncertain environments without prior information.

The contributions in probabilistic occupancy grid mapping and autonomous exploration are extended in several ways. Probabilistic mapping and autonomous exploration are extended from 2D to 3D with two approaches: projecting a 3D map onto a 2D plane for

exploration, or using the full 3D map for entropy predictions. This concept is also extended to cooperative multi-vehicle autonomous exploration, where a bidding-based framework coordinates robotic efforts to understand the map. The approach follows a receding horizon, which improves collision-avoidance and substantially reduces unnecessary robotic actions. Furthermore, this approach is extended to autonomous patrol, where exponential map degradation incentivizes robot team members to periodically revisit areas. These extensions to occupancy grid mapping and autonomous exploration are demonstrated with numerous simulations and experimental results.

8.2 Future Directions

There are several possible future directions with this research. Most importantly, localization, where the robot determines its own position and attitude, should be included in conjunction with the proposed mapping approach, thereby completing the full simultaneous localization and mapping (SLAM) problem. Secondly, the cost map portion of Dijkstra's search can be expensive, particularly with multiple vehicles in a large space, or in 3D environments. A more efficient approach that provides collision-free distance costs during autonomous exploration could improve computational speed for autonomous exploration. Finally, enhancements in memory allocation and distributed computing are required for these algorithms to be applied in a decentralized manner for systems with more autonomy.

Bibliography

[1] Map-a-Planet: Data sets. USGS Astrogeology Research Program, https://www.mapaplanet.org/explorer/help/data_set.html, 2013.

[2] About ROS. Robot Operating System, http://www.ros.org/about-ros/, 2018.

[3] The industry's foundation for high performance graphics. OpenGL, https://www.opengl.org, 2018.

[4] Product basics. 80/20 University, https://www.8020.net/university-product-basics, 2018.

[5] Christian Dornhege Alexander Kleiner. A frontier-void-based approach for autonomous exploration in 3D. *Advanced Robotics*, 27(6):459–468, 2013.

[6] Rosiery Maia Anderson A. S. Souza and Luiz M. G. Gonçalves. *Current Advancements in Stereo Vision, Chapter 9: 3D Probabilistic Occupancy Grid to Robotic Mapping with Stereo Vision*. Intech, 2012.

[7] Franz Andert. Drawing stereo disparity images into occupancy grids: Measurement model and fast implementation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[8] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–386, 2005.

[9] Henry Carrillo, Philip Dames, Vijay Kumar, and José A. Castellanos. Autonomous robotic exploration using occupancy grid maps and graph SLAM based on Shannon and Rényi entropy. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015.

[10] Chen Chen and Yinhang Cheng. Research of mobile robot slam based on ekf and its improved algorithms. In *Third International Symposium on Intelligent Information Technology Application*, volume 1, pages 548–552. IEEE, November 2009.

[11] Chen X et al Chen H, Ding S. Global finite-time stabilization for nonholonomic mobile robots based on visual servoing. *International Journal of Advanced Robotic Systems*, 11(11), 2014.

[12] Han-Lim Choi, Luc Brunet, and Jonathan P. How. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25(4):912–926, August 2009.

[13] Howie Choset, Kevin Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Intelligent Robotics and Autonomous Agents. Massachusetts Institute of Technology, Cambridge, Massachusetts, 2005.

[14] Luc Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986.

[15] M. B. Dias and A. Stentz. A free market architecture for distributed control of a multirobot system. In *6th International Conference on Intelligent Autonomous Systems*, pages 115–122, Venice, Italy, July 2000.

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

[17] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: Part 1. *IEEE Robotics and Automation Magazine*, 13(2):99–110, June 2006.

[18] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer*, pages 46–57, 1989.

[19] N. Gasilov, M. Dogan, and V. Arici. Two-stage shortest path algorithm for solving optimal obstacle avoidance problem. *IETE Journal of Research*, 57(3):278–285, 2011.

[20] Brian P. Gerkey and Maja J. Matarić. Sold!: Auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 18(5):758–768, 2002.

[21] Farhad Goodarzi, Daewon Lee, and Taeyoung Lee. Geometric nonlinear pid control of a quadrotor uav on SE(3). In *2013 European Control Conference (ECC)*, pages 3845–3850. IEEE, July 17–19 2013.

[22] Farhad Goodarzi and Taeyoung Lee. Geometric stabilization of a quadrotor uav with a payload connected by flexible cable. In *Proceeding of the American Control Conference*, pages 4925–4930, 2014.

[23] Farhad Goodarzi and Taeyoung Lee. Stabilization of a rigid body payload with multiple cooperative quadrotors. *ASME Journal of Dynamic Systems, Measurement and Control*, 138(12), 2016.

[24] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: an efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, 2013.

[25] Coursera Inc. Robotics: estimation and learning, 2016.

[26] Steven G. Johnson. Saddle-point integration of c infinity bump functions. arXiv:1508.04376, August 19 2015.

[27] Dominik Joho, Cyrill Stachniss, Patrick Pfaff, and Wolfram Burgard. Autonomous exploration for 3D map learning. In Karsten Berns and Tobias Luksch, editors, *Autonome Mobile Systeme (AMS)*, pages 22–28. Springer, 2007.

[28] Evan Kaufman, Zhuming Ai, and Taeyoung Lee. Autonomous exploration by expected information gain from probabilistic occupancy grid mapping. In *Proceedings of the 2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, pages 246–251, San Francisco, USA, December 13–16 2016. IEEE.

[29] Evan Kaufman, Kiren Caldwell, Daewon Lee, and Taeyoung Lee. Design and development of a free-floating hexrotor uav for 6-dof maneuvers. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, MT, March 2014.

[30] Evan Kaufman and Taeyoung Lee. Autonomous aerial exploration for topological mapping of mars environments. In *AIAA Guidance, Navigation, and Control Conference*. submitted, 2019.

[31] Evan Kaufman, Taeyoung Lee, Zhuming Ai, and Ira S. Moskowitz. Bayesian occupancy grid mapping via an exact inverse sensor model. In *American Control Conference*, pages 5709–5715. IEEE, 2016.

[32] Evan Kaufman, Kuya Takami, Zhuming Ai, and Taeyoung Lee. Autonomous exploration with exact inverse sensor models. *Journal of Intelligent and Robotic Systems*, 2017.

[33] Evan Kaufman, Kuya Takami, Zhuming Ai, and Taeyoung Lee. Autonomous quadrotor 3D mapping and exploration using exact occupancy probabilities. In *The Second IEEE International Conference on Robotic Computing*, pages 49–55, Laguna Hills, 2018.

[34] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, pages 1437–1454, 2012.

[35] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[36] Andreas Kolling and Stefano Carpin. Extracting surveillance graphs from robot maps. In *nternational Conference on Intelligent Robots and Systems (IROS)*, pages 2323–2328, Nice, France, September 2008. IEEE/RSJ.

[37] Tomàš Krajník, Jaime Pulido Fentanes, Grzegorz Cielniak, Christian Dondrup, and Tom Duckett. Spectral analysis for long-term robotic mapping. In *International Conference on Robotics and Automation (ICRA)*, pages 3706–3711, Hong Kong, China, May 2014. IEEE.

[38] Tomàš Krajník, Jaime Pulido Fentanes, Marc Hanheide, and Tom Duckett. Persistent localization and life-long mapping in changing environments using the frequency map

enhancement. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 4558–4563, Daejeon, Korea, October 2016. IEEE/RSJ.

[39] Rainer Kümmerle, Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner. On measuring the accuracy of slam algorithms. *Autonomous Robots*, 27(4):387–407, 2009.

[40] T. Lee, M. Leok, and N. H. McClamroch. Control of complex maneuvers for a quadrotor UAV using geometric methods on SE(3). arXiv.

[41] T. Lee, M. Leok, and N. H. McClamroch. Nonlinear robust tracking control of a quadrotor UAV on SE(3). arXiv.

[42] Taeyoung Lee, Melvin Leok, and N.H. Mcclamroch. Geometric tracking control of a quadrotor uav on se(3). In *Proceedings of the IEEE Conference on Decision and Control*, pages 5420–5425, Atlanta, GA, December 2010.

[43] Ivan Maurović, Marja Dakulović, and Ivan Petrović. Autonomous exploration of large unknown indoor environments for dense 3D model building. In *Proceedings of the 19th World Congress*, pages 10188–10193, Cape Town, South Africa, August 2014. The International Federation of Automatic Control (IFAC).

[44] Daniel Meyer-Delius, Maximilian Beinhofer, and Wolfram Burgard. Occupancy grid models for robot mapping in changing environments. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[45] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Proceeding of the National Conference on Artificial Intelligence (AAAI)*, pages 593–598, Edmonton, Canada, 2002.

[46] Hans P. Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *IEEE Conference on Robotics and Automation*, 1985.

[47] Charles Pippin, Henrik Christensen, and Lora Weiss. Performance based task assignment in multi-robot patrolling. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 70–76, 2013.

[48] Katrin Pirker, Matthias Ruther, Horst Bischof, and Gerald Schweighofer. Fast and accurate environment modeling using three-dimensional occupancy grids. In *IEEE International Conference on Computer Vision Workshops*, 2011.

[49] David Portugal and Rui Rocha. Msp algorithm multi-robot patrolling based on territory allocation using balanced graph partitioning. In *Symposium on Applied Computing (SAC 2010)*, pages 1271–1276, Sierre, Switzerland, March 2010.

[50] Sanem Sariel and Tucker Balch. Real time auction based allocation of tasks for multi-robot exploration problem in dynamic environments. In *Integrating Planning into Scheduling: Papers from the 2005 AAAI Workshop*, 2005.

[51] Rahul Sawhney, K. Madhava Krishna, and Kannan Srinathan. On fast exploration in 2D and 3D terrains with multiple robots. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 73–80, 2009.

[52] P.G.C.N. Senarathne and Danwei Wang. Towards autonomous 3D exploration using surface frontiers. In *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 34–41, 2016.

[53] Reid Simmons, David Apfelbaum, Wolfram Burgard, Dieter Fox, Mark Moors, Sebastian Thrun, and Håkan Younes. Coordination for multi-robot exploration and mapping. In AAAI, editor, *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 852–858, 2000.

[54] David E. Smith, Maria T. Zuber, Sean C. Solomon, Roger J. Phillips, James W. Head, James B. Garvin, W. Bruce Banerdt, Duane O. Muhleman, Gordon H. Pettengill, 2 Frank G. Lemoine Gregory A. Neumann, 1, James B. Abshire, Oded Aharonson, C. David Brown, Steven A. Hauck, Anton B. Ivanov, Patrick J. McGovern, H. Jay Zwally, and Thomas C. Duxbury. The global topography of mars and implications for surface evolution. *Science*, 284(5419):1495–1503, May 1999.

[55] Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using Rao-Blackwellized particle filters. In *In RSS*, pages 65–72, 2005.

[56] Georg Tanzmeister, Julian Thomas, Dirk Wollherr, and Martin Buss. Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[57] Sebastian Thrun. Learning occupancy grids with forward models. In *Proceedings of the 2001 IEEE/RSU International Conference on Intelligent Robots and Systems*, 2001.

[58] Sebastian Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous Robots*, 15(2):111–127, 2003.

[59] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents. Massachusetts Institute of Technology, Cambridge, Massachusetts, 2005.

[60] Denis Wolf and Gaurav Sukhatme. Mobile robot simultaneous localization and mapping in dynamic environments. *Autonomous Robots*, 1(19):53–65, 2005.

[61] Tse-Huai Wu, Evan Kaufman, and Taeyoung Lee. Globally asymptotically stable attitude observer on so(3). In *Proceedings of the 54th IEEE Conference on Decision and Control*, pages 2164–2168, Osaka, Japan, December 2015.

[62] Kai Wurm, Armin Hornung, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proceeding of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, 2010.

[63] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151. IEEE, 1997.

[64] Brian Yamauchi. Frontier-based exploration using multiple robots. In ACM, editor, *Second International Conference on Autonomous Agents*, pages 47–53, 1998.

[65] Chuanbo Yan and Tao Zhang. Multi-robot patrol: A distributed algorithm based on expected idleness. *International Journal of Advanced Robotic Systems*, pages 1–12, 2016.

[66] Yanping Zhang and Yang Xiao. A patrolling scheme in wireless sensor and robot networks. In *The Third International Workshop on Wireless Sensor, Actuator and Robot Networks*, pages 513–518. IEEE, 2011.

[67] Cheng Zhu, Rong Ding, Mengxiang Lin, and Yuanyuan Wu. A 3D frontier-based exploration tool for mavs. In *IEEE 27th International Conference on Tools with Artificial Intelligence*, pages 348–352, 2015.

[68] Robert Zlot, Anthony Stentz, M. Bernardine Dias, and Scott Thayer. Multi-robot exploration controlled by a market economy. In IEEE, editor, *Proceedings of the 2002 IEEE International Conference on Robotics & Automation*, pages 3016–3023, 2002.

Appendix A: Proof of Proposition 1

Unnormalized Reduced Map Inverse Sensor Model For the reduced map of the l-th ray, the unnormalized probability that corresponds to the terms without $\eta_{t,l}$ in (2.4) is given by

$$\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \sum_{r \in \mathcal{M}_k} p(z_{t,l}|r, X_t) P(r|X_{1:t-1}, Z_{1:t-1}). \tag{A.1}$$

Recall $\mathcal{M}_k$ corresponds to the set of maps where the k-th cell is occupied. Define a subset $\mathcal{N}_{i,k} \subset \mathcal{M}_k$ for $1 \leq i \leq k$ be the set of maps where the i-th cell is the first occupied cell. More explicitly,

$$\mathcal{N}_{i,k} = \{r \in \mathcal{M}_k | \mathbf{r}_{l,i+}\} = \{r \in \mathcal{M}_k | \mathbf{r}_{l,1} = 0, \ldots, \mathbf{r}_{l,i-1} = 0, \mathbf{r}_{l,i} = 1, \mathbf{r}_{l,k} = 1\}.$$

Then, $\mathcal{M}_k$ can be written as $\mathcal{M}_k = \bigcup_{i=1}^{k} \mathcal{N}_{i,k}$. Using this, the summation over $\mathcal{M}_k$ at (A.1) can be decomposed of the summation over each $\mathcal{N}_{i,k}$ to obtain

$$\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \sum_{i=1}^{k} \left\{ \sum_{r \in \mathcal{N}_{i,k}} p(z_{t,l}|r, X_t) P(r|X_{1:t-1}, Z_{1:t-1}) \right\}.$$

This is motivated by the fact that the forward sensor model $p(z_{t,l}|r, X_t)$ is identical for all maps in $\mathcal{N}_{i,k}$, such that $p(z_{t,l}|r, X_t) = p(z_{t,l}|\mathbf{r}_{l,i+}, X_t)$ and it is moved left of the summation to obtain

$$\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \sum_{i=1}^{k} \left\{ p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) \sum_{r \in \mathcal{N}_{i,k}} P(r|X_{1:t-1}, Z_{1:t-1}) \right\}. \tag{A.2}$$

The last term of the above expression corresponds to the a priori probability of $\mathcal{N}_{i,k}$. When $i < k$, it is given by

$$\sum_{r \in \mathcal{N}_{i,k}} P(r|X_{1:t-1}, Z_{1:t-1}) \\ = \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}) P(\mathbf{r}_{l,k}|X_{1:t-1}, Z_{1:t-1}), \quad \text{(A.3)}$$

and when $i = k$,

$$\sum_{r \in \mathcal{N}_{k,k}} P(r|X_{1:t-1}, Z_{1:t-1}) = \left\{ \prod_{j=0}^{k-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} P(\mathbf{r}_{l,k}|X_{1:t-1}, Z_{1:t-1}). \quad \text{(A.4)}$$

Substituting (A.3) and (A.4) into (A.2), we obtain (2.6).

Complement of the Unnormalized Reduced Map Inverse Sensor Model An analytic expression for the complement of the unnormalized inverse sensor model is also required for obtaining the normalizer of the l-th ray at time t, namely $\eta_{t,l}$. Let $\bar{\mathcal{M}}_k$ be the set of maps where the k-th cell is unoccupied, i.e., $\bar{\mathcal{M}}_k = \{r \in \{0,1\}^{n_{r,l}} \,|\, \mathbf{r}_{l,k} = 0\}$. Similar to (A.1),

$$\tilde{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \sum_{r \in \bar{\mathcal{M}}_k} p(z_{t,l}|r, X_t) P(r|X_{1:t-1}, Z_{1:t-1}). \quad \text{(A.5)}$$

Let $\bar{\mathcal{N}}_{i,k} \subset \bar{\mathcal{M}}_k$ for $1 \leq i \leq n_{r,l}$ and $i \neq k$ be the set of maps where the i-th cell is the first occupied cell. More explicitly,

$$\bar{\mathcal{N}}_{i,k} = \{r \in \bar{\mathcal{M}}_k \,|\, \mathbf{r}_{l,1} = 0, \ldots, \mathbf{r}_{l,i-1} = 0, \mathbf{r}_{l,i} = 1, \mathbf{r}_{l,k} = 0\}.$$

Then, we have $\bar{\mathcal{M}}_k = \bigcup_{\substack{i=1 \\ i \neq k}}^{n_{r,l}} \bar{\mathcal{N}}_{i,k}$. Note that the forward sensor model $p(z_{t,l}|r, X_t)$ is identical

for any maps in $\bar{\mathcal{N}}_{i,k}$, such that $p(z_{t,l}|r, X_t) = p(z_{t,l}|\mathbf{r}_{l,i+}, X_t)$. Similar to (A.2),

$$\tilde{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = \sum_{\substack{i=1 \\ i\neq k}}^{n_{r,l}} \left\{ p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) \sum_{r\in\bar{\mathcal{N}}_{i,k}} P(r|X_{1:t-1}, Z_{1:t-1}) \right\}, \tag{A.6}$$

where the last term corresponds to the a priori probability of $\bar{\mathcal{N}}_{i,k}$. When $i < k$, it is given by

$$\begin{aligned} &\sum_{r\in\bar{\mathcal{N}}_{i,k}} P(r|X_{1:t-1}, Z_{1:t-1}) \\ &\quad = \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}) P(\bar{\mathbf{r}}_{l,k}|X_{1:t-1}, Z_{1:t-1}), \end{aligned} \tag{A.7}$$

and when $k < i$,

$$\sum_{r\in\bar{\mathcal{N}}_{i,k}} P(r|X_{1:t-1}, Z_{1:t-1}) = \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}). \tag{A.8}$$

Substituting (A.7) and (A.8) into (A.6), we obtain

$$\begin{aligned} \tilde{P}(\bar{\mathbf{r}}_k|z_{t,l}, X_{1:t}, Z_{1:t-1}) &= P(\bar{\mathbf{r}}_{l,k}|X_{1:t-1}, Z_{1:t-1}) \\ &\times \left[\sum_{i=1}^{k-1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{l,i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}) \right] \\ &+ \left[\sum_{i=k+1}^{n_{r,l}+1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_{l,j}|X_{1:t-1}, Z_{1:t-1}) \right\} p(z_{t,l}|\mathbf{r}_{i+}, X_t) P(\mathbf{r}_{l,i}|X_{1:t-1}, Z_{1:t-1}) \right], \end{aligned} \tag{A.9}$$

where $p(z_{t,l}|\mathbf{r}_{(n_r+1)+}, X_t)$ corresponds to the forward sensor model of maximum reading and $P(\mathbf{r}_{l,n_{r,l}+1}|X_{1:t-1}, Z_{1:t-1}) = 1$ for convenience for the empty map case.

Normalizer We have

$$P(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) + P(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) = 1.$$

Since they share the same normalizer $\eta_{t,l}$, this implies

$$\eta_{t,l} = [\tilde{P}(\mathbf{r}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1}) + \tilde{P}(\bar{\mathbf{r}}_{l,k}|z_{t,l}, X_{1:t}, Z_{1:t-1})]^{-1}.$$

Substituting (2.6) and (A.9), and rearranging, we obtain (2.7).

Appendix B: Proof of Proposition 2

Single Ray Expected Value of Entropy In this appendix, we prove the expected value of entropy for a single measurement ray. We consider the expected entropy due to measurement ray z at pose X,

$$\mathrm{E}[H(P(m|X,z))] = \int_{z_{\min}}^{z_{\max}} H(P(m|X,z))p(z|X)dz. \tag{B.1}$$

We discretize the measurement ray space such that z falls on points along the measurement ray intersecting with grid cell edges. The discretized expected value of (B.1) is

$$\mathrm{E}[H(P(m|X,z))] = \sum_{k=1}^{n_r+1} \Big\{ H(P(m|X,z_k))P(z_k|X) \Big\}, \tag{B.2}$$

where index z_k is the distance from the location of pose X to the k-th grid cell of the reduced map $r \subset m$ for this ray, known from geometry (see Figure 2.2).

Standing alone, the term $P(z_k|X)$ from (B.2) has a convoluted meaning because the depth z_k does not directly depend on the map. However, we present a method to obtain this discretized probability. Following the assumption that z is discretized to known distances, the probabilities are proportional to their densities, as the area under the density curve is infinitesimal and fixed. Accounting for all cases, the probability is

$$P(z_k|X) = \frac{p(z_k|X)}{\sum_{i=1}^{n_r+1} p(z_i|X)}. \tag{B.3}$$

Conveniently, this density corresponds to the inverse normalizer defined in (2.2) as the denominator from Bayes' rule,

$$p(z_k|X) = \eta_k^{-1}, \tag{B.4}$$

where η_k^{-1} is defined in (2.7) as the inverse sensor normalizer corresponding to a measurement of the k-th cell of reduced map r. Thus, the inverse normalizer is

$$\eta_k^{-1} = \sum_{i=1}^{n_r+1} \left\{ \prod_{j=0}^{i-1} P(\bar{\mathbf{r}}_j) \right\} p(z_k|\mathbf{r}_{i+}, X) P(\mathbf{r}_i). \tag{B.5}$$

By substituting (B.3) and (B.4) into (B.2), the expected entropy from individual measurement ray z at X is

$$\mathrm{E}[H(P(m|X,z))] = \left(\sum_{i=1}^{n_r+1} \eta_i^{-1} \right)^{-1} \sum_{k=1}^{n_r+1} \left\{ H(P(m|X,z_k)) \eta_k^{-1} \right\}, \tag{B.6}$$

where η_k^{-1} is taken from (B.5). Here, (B.6) provides the expected entropies for cells inside the field of view of reduced map r. The entropies of cells outside the field of view remain unchanged.