Approximate Continuous Belief Distributions for Exploration

Wennie Tabib

CMU-CS-19-108

May 2019

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Nathan Michael, Co-Chair
Red Whittaker, Co-Chair
Nancy Pollard
Debadeepta Dey, Microsoft Research

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by a Paul and Daisy Soros Fellowship for New Americans and the National Aeronautics and Space Administration under grant numbers NNX14AL66H, NNX16AD98G, NNX15CK15P, and NNX16CK16C. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Abstract

Efficient and robust robotic exploration has the potential to solve significant challenges and save lives when disasters occur underground. Cave rescues to extricate trapped or lost spelunkers are difficult and demanding endeavors performed dozens of times each year in the United States in environments that are often neither mapped nor surveyed and have limited to nonexistent communications due to the convoluted nature of underground voids. Underground nuclear waste storage facilities become inaccessible to humans when radiation leaks occur, so efficient pose estimation and mapping to localize radiation leaks is of the utmost importance. Catastrophic sinkholes appear suddenly in areas of karst terrain throughout the United States, swallowing homes, and trapping residents in debris.

Without reliable communications, robots must operate autonomously to efficiently explore these subterranean environments, but many Simultaneous Localization and Mapping (SLAM) techniques do not generate maps fit for active perception. Occupancy grid map techniques are typically used after solving the SLAM problem to generate feasible trajectories. The advantage of occupancy grid maps is they may be updated quickly, but the gains in speed come at the cost of either memory efficiency or fidelity. In exploration contexts, systems maneuvering in large environments must elect to decrease the resolution of the occupancy grid map in order to mitigate explosive memory demands or suffer increasingly slower speeds when manipulating occupancy grid maps with small cell sizes. For computationally constrained systems, the latter is prohibitive, but employing low-resolution environment representations has significant disadvantages. For example, small passageways and hazards may be obscured and rich details obliterated.

Gaussian Mixture Models (GMMs) are well suited to compactly represent sensor observations and model the structural correlations present in the environment. These generative models are advantageous as compared to voxelized representations that assume independence between cells and lose dependencies between spatially distinct locations. GMM-based perception tasks such as registration have been studied, but solutions are either not real-time viable or have not been evaluated with large, real-world datasets. There are few works that address the viability of leveraging these models for tasks such as SLAM and exploration, because a significant challenge to overcome is the time needed to create these models.

This thesis develops information-theoretic exploration with approximate continuous distributions that unifies high-resolution, low memory footprint environment modeling with occupancy mapping techniques while remaining amenable to local and global pose updates. This is possible through innovations in robust distribution to distribution registration, arbitrary resolution occupancy modeling, real-time information-theoretic exploration, and simultaneous localization and mapping. These developments are evaluated in simulation, with real-world datasets, and onboard an aerial system and tested in complex environments. The results demonstrate that leveraging compact generative models yields substantial gains over state-of-the-art methods in model fidelity, accuracy, and memory efficiency.

Acknowledgments

First, I would like to thank my advisors, Profs. Nathan Michael and Red Whittaker, for providing insight, guidance, and feedback while giving me the freedom to pursue interesting research throughout my time at CMU. I am also grateful to my thesis committee members, Prof. Nancy Pollard and Dr. Debadeepta Dey, who provided valuable perspectives and advice throughout my thesis.

I would also like to thank my colleagues in the Resilient Intelligent Systems Lab and Lunar group at CMU for invaluable collaborations and memorable experiences, especially Cormac O'Meadhra, Kshitij Goel, Mosam Dabhi, Curtis Boirum, Micah Corah, Aditya Dhawale, Shobhit Srivastava, John Yao, Vibhav Ganesh, Tim Lee, Chris Cunningham, Heather Jones, Shaurya Shankar, Ellen Cappo, Arjav Desai, Matt Collins, Alex Spitzer, Xuning Yang, Logan Ellis, Moses Bangura, and Vishnu Desaraju. Thanks also to Clare Cui, Maitreya Naik, Logan Wan, and Angad Sidhu for developing one of the aerial systems used in this work. I am also grateful to Chuck Whittaker, Nora Kazour, Karen Widmaier, Deb Cavlovich, and Catherine Copetas for your support and help throughout the years.

Thanks also go out to the NASA Space Technology Fellowship Program Team, including my mentor, Larry Matthies, and program manager, Claudia Meyer. Thanks also to the Paul and Daisy Soros Fellowship Program Team and, in particular, Stan Heginbotham. This thesis would not have been possible without your generous support.

I would also like to thank my colleagues at Astrobotic Technology, Inc. for enabling the NASA STTR that started all of this work. In particular, Kevin Peterson, Steve Huber, Kerry Snyder, Eric Amoroso, and Fraser Kitchell. Thanks also go out to Rob Mueller at the Kennedy Space Center for taking an interest in this work and funding it through the STTR program. I'd also like to thank my collaborators at the West Virginia Cave Conservancy and Carroll Bassett, in particular, for letting me fly an aerial system in Rapps Cave.

I owe a deep debt of gratitude to my friends and family who supported me throughout the years. To Lauren Schneider and Dean Thompson, thank you for providing a home away from home. To Nancy and Dale Osbun, thank you for your love and support. To Sandie and Don Thomas, thank you for your friendship and guidance over the years. Special thanks also to my mother, sister, and brother for believing in me all these years.

Finally, I would like to thank my husband, Brian, for the incredible amount of support over these last years. Thank you for waking up at 4AM and driving with me to West Virginia to fly robots in caves, dealing with long separations while I worked at JPL, believing in me, and doing more than I thought was possible from one man alone. This thesis would not exist without you, so I dedicate this work to you.

To my husband, Brian.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Robotic exploration in subterranean voids demands robust operation due to the lack of global positioning systems and limited or non-exist communications. Image-based navigation techniques fail in darkness even with active illumination as the sensor may be washed out if the robot approaches a surface too closely or takes a turn too tightly, so alternative sensing modalities, such as depth and LiDAR sensors, must be employed to estimate pose. In particular, cave rescue is challenging because the interiors of caves typically have not been mapped. For example, of the 4378 documented caves in Virginia, only 1348 have been mapped [79]. Dozens of cave rescues are performed each year when spelunkers become lost, succumb to rockfall, cold temperatures, or contaminated air, and require rescue [79]. These rescues are difficult, demanding and require highly specialized training to locate and extricate an injured spelunker.

Subterranean storage of nuclear waste, or nuclear waste management, is a domain that desperately needs robotic solutions, because when radiation leaks occur, humans cannot be deployed to find and seal the leaks without risking exposure. The Waste Isolation Pilot Plant (WIPP), designed to store nuclear waste for 200,000 years, experienced a radiation leak in February 2014 that exposed several workers and led to its closure for three years [17]. Robotic systems must estimate pose and create maps with high accuracy and detail to enable localization of the source of leaks.

(a) (b) (c)

Figure 1.1: (a) Cave rescue in Tennessee. (b)Waste Isolation Pilot Plant (Image Source: U.S. Dept. of Energy). (c) A $330\,\mathrm{m}$ *deep sinkhole opened up in Gautemala in 2007 killing two people and forcing evacuation of 1000. (Image Source: National Geographic)*

Sinkholes are catastrophic and sudden events that occur in karst terrain around the world, and may be hundreds of meters deep. These events are triggered due to either natural events, such as weaknesses in karst terrain, or man-made negligence, like aging water systems. While some sinkholes may cause minimal damage and are only a few meters wide and deep, others have been recorded to be hundreds of meters deep (see Fig. 1.1c) [86]. Efficient robotic exploration and mapping of these environments can determine the cause of collapse and localize survivors.

Active perception methodologies seek to evaluate and select actions that increase the information content derived from sensor observations. Exploration is one such active perception manifestation that selects actions to maximize coverage of an unknown environment. To enable exploration, state-of-the-art systems are typically composed of several concurrent processes that operate on the same pointcloud data to perform a variety of perception tasks, which results in a complex and monolithic software system.

For example, information theoretic exploration systems must estimate pose, which typically requires maintaining a high-resolution occupancy map that scales as the size of the environment increases. In addition, a separate occupancy representation may be stored with similar or greater memory demands for action evaluation and selection. Processes must also be run to identify hazards along the current trajectory and re-plan safe, feasible trajectories given the most up-to-date

representation of the environment. All perceptual tasks must be robust to uncertainty and amenable to local and global pose updates.

Furthermore, commercial off-the-shelf sensors, that produce observations composed of hundreds of thousands or millions of points per second, are ubiquitous and cheap. Processing this data on computationally constrained autonomous systems may be intractable because the data is too dense, contains too much noise or too many outliers, or because the points do not encode the spatial correlations present in the environment.

1.1 Challenges

Three challenges are identified that must be overcome to enable robust, efficient exploration in subterranean environments on computationally constrained systems:

- **Efficient, high-resolution, and compact perceptual modeling:** High-fidelity environment representations that are amenable for transmission across low-bandwidth communications channels.
- **Occupancy modeling at arbitrary resolution:** The perceptual model must admit a method to derive occupancy in real-time to enable information-theoretic exploration.
- **Local and global consistency in environment representation:** The environment representation must remain amenable to local and global updates in pose.

High-fidelity and efficient perceptual modeling is well suited to a formulation based on Gaussian mixture models (GMMs). The GMM is a generative model that represents data points as a finite number of Gaussian distributions (see Chapter 3 for additional details). The compactness of these models may be leveraged to enable transmission of high-resolution environment data between systems or operators in environments with limited communications.

However, applying GMMs to the problem of perceptual modeling in real-world environments is challenged by the limited processing capabilities available onboard most small-scale, autonomous

systems such as micro aerial vehicles. The time to train a GMM scales with the number of samples in the dataset and desired number of mixture components. Furthermore, the availability of low-cost, high-resolution depth sensors has exploded in recent years and low-power computing has not kept pace. As a result, many powerful perceptual modeling techniques are not real-time viable for compute-constrained robotic systems.

1.2 Thesis Contributions

This thesis seeks to address these challenges and enable efficient exploration through the development and validation of a unified high-resolution perceptual and occupancy modeling framework that leverages Gaussian Mixture Models (GMMs) to represent the environment and is amenable to local and global pose updates. Sensor observations are encoded as generative models that require significantly less memory than storing raw pointclouds, are more robust to noise and outliers, and preserve geometric structure of the environment. These models may be used to estimate pose, derive occupancy and close the loop.

The contributions of this thesis are summarized below and detailed in the subsequent chapters.

Chapter 4: Information-theoretic Exploration An exploration framework that leverages occupancy grid maps for environment representation and motion primitives constructed on a state space lattice. An information-theoretic cost function that maximizes the mutual information between future sensor observations and the map is used to select actions. The approach yields an efficient method for maximizing coverage of significantly rich, 3D environments [82].

Chapter 5: On-Manifold GMM Registration A robust formulation for registering pointclouds represented as GMMs by minimizing the squared L2 norm between two distributions that outperforms state-of-the-art registration methods [83].

Chapter 6: Arbitrary Resolution Occupancy Modeling using GMMs A method for deriving occupancy from GMM maps that outperforms the state-of-the-art occupancy modeling ap-

proaches [62].

Chapter 7: Local Occupancy Mapping using GMMs for Exploration A method that leverages GMMs for high-fidelity perceptual modeling and from which occupancy may be derived to enable real-time information-theoretic exploration evaluated in simulation and onboard an aerial system equipped with a LiDAR sensor.

Chapter 8: Simultaneous Localization and Mapping using GMMs An extension of the registration approach that employs a pose graph formulation for simultaneous localization and mapping and develops a method to close the loop.

Chapter 2

Related Work

This thesis seeks to enable efficient exploration with respect to a high-fidelity perceptual model that is compact, encodes occupancy, and is amenable to local and global pose updates. Techniques from pointcloud and distribution-to-distribution registration, occupancy modeling, simultaneous localization and mapping, and exploration are leveraged and extended. This chapter provides a brief overview of related works in these areas.

2.1 Registration

The state-of-the-art navigation and mapping algorithms can be segregated into image-based and depth-based pose estimation techniques. Feature-based image processing algorithms fail in environments that do not enable feature extraction and matching, particularly, in low-lighting conditions and in the presence of repetitive structure. Dense methods [60, 75] operate directly on pointclouds, but are typically more sensitive to initialization.

Direct methods may be leveraged to estimate pose from high-frame rate depth or RGB-D sensor observations, but require motions between frames to be small and rely on the brightness constancy assumption which does not hold in subterranean environments [40, 49]. Alismail et al. [1] seek to overcome the limitations of feature-based and direct methods by developing a hybrid method

that employs bit-plane features within a direct alignment framework, but the approach fails in subterranean environments when taking tight turns due to washout when the light source output is constrained to be very close to the image sensor.

The Iterative Closest Point (ICP) algorithm [7] computes point correspondences and identifies the optimal alignment of corresponding pairs in the least-squares sense. While fast, it is sensitive to outliers and initialization, and many variants have been proposed to overcome these limitations [25, 56]. Generalized ICP [74] increases the robustness of ICP by modeling the local structure of the data as a disk centered on each point and normal and matching via a plane-to-plane distance metric. A significant limitation of standard ICP approaches is the use of hard correspondences, but while soft correspondence methodologies have been shown to yield improved performance, these methods come at increased computational cost [84]. While one could argue that downsampling the pointcloud is a viable method for increasing computational speed, this operation may result in loss of information [43].

The Normal Distribution Transform (NDT) framework learns anisotropic multivariate normal distributions over points assigned to a discrete set of bins in the observed space and performs registration by minimizing the L2 norm between two NDT maps [80]. This approach discretizes the space, which limits the representational power of the distributions and increases the computational burden of the framework. The approach developed in this thesis (Chapter 5) is distinct from this work in that the role of the determinant is explicitly accounted for and all possible correspondences between the two distributions are considered.

Approaches have been proposed that employ GMMs to represent pointclouds [27, 28, 31, 45]. Alignment is performed by minimizing a distance metric as in [45], or integrating into the model learning stage through an expectation-maximization (EM) framework [27, 31]. EM-based approaches assume that the pointclouds to be matched are both observations of the same scene, which is not realistic in practice as measurements often consist of large non-overlapping regions.

Methods that leverage probabilistic generative models to register point sets have been demonstrated to be more robust to outliers and noise [26, 31]. However, state-of-the-art methods are either not real-time viable [31] or tested on a single scan with varying rotations, translations and rates of downsampling so that the data is virtually identical for every test [26]. Prior work has demonstrated the viability of real-time GMM computation through hardware acceleration [28].

2.2 Occupancy Modeling

Occupancy grid mapping [29] is a technique used to generate volumetric 2D or 3D environment models. The environment is modeled as a set of discrete cells with an associated probability of occupancy so that each cell may be classified as free, occupied or unknown. The probabilistic representation is critical for enabling a consistent map representation in the presence of sensor noise and random measurements due to dynamic obstacles or reflective surfaces in the environment. However, occupancy grid maps suffer from fixed grid resolution, which leads to high memory complexity - quadratic in 2D and cubic in 3D, and an independence assumption between cells in the grid. OctoMap [38] overcomes some of these limitations by employing a hierarchical structure based on octrees that performs a recursive subdivision to successively smaller voxel sizes until a minimum voxel size is reached. However, the downside of the octree approach is that the cost of a query is $\mathcal{O}(\log n)$, where n is the number of nodes. In contrast, the cost of performing a lookup in an occupancy grid map, when implemented efficiently, is $\mathcal{O}(1)$. Osman et al. [63] propose to overcome the independence assumption by using ray potentials and surface priors to update conditionally dependent occupancy cells in a consistent manner.

The Normal Distribution Transform Occupancy Map (NDT-OM) [73] is a semi-continuous environment modeling approach that leverages a 3D occupancy grid map and stores a uniformly weighted Gaussian density in each occupied cell. NDT mapping mitigates against errors in discretization to a certain extent through the incorporation of the Gaussian density, but suffers from

large memory requirements due to the need to retain the voxelized representation of the environment.

OCallaghan and Ramos [64] introduce the Gaussian Process Occupancy Map (GPOM), which is a continuous environment model, by learning a representation of the environment using Gaussian Process regression. However, the implementation proposed by OCallaghan and Ramos [64] is not real-time viable. Kim and Kim [50] develop a more computationally tractable method by dividing the training data into small subsets and applying a mixture of GPs and subsequently further improve the efficiency through the application of sparse Gaussian processes [51]; however, the training time remains too large for real-time applications. Jadidi et al. [42] develop incremental GPOM methods that efficiently learn separate models for free and occupied space, which are combined to generate a probability of occupancy.

Doherty et al. [24] discretize the environment and store two hyper-parameters in each cell used to calculate occupancy. When a sensor observation is received, the parameters in the cell are recursively updated with the result from a sparse kernel. Inference is performed using a test-data octree. Hilbert Maps [71] train a classifier in a reproducing Hilbert space and have been proposed as an efficient method of learning continuous occupancy models.

2.3 Exploration

Planetary exploration systems to date have largely left questions about the cost of transmitting data to operators unaddressed. Arora et al. [3] leverage Bayesian networks to create a mapping between multi-modal sensor data and variables of interest coupled with a Monte Carlo Tree Search planner to maximize information gathering with the goal of limiting operator interaction given the significant delay induced by interplanetary travel. However, the cost of transmitting data to enable information sharing with human operators and planetary scientists is not discussed.

Hahn et al. [36] assume a tele-operated vehicle and develop a method to identify victims in a search and rescue scenario that fuses thermal and depth observations. Bandwidth constraints of the environment representation are not reported in the analysis of their approach. Papachristos et al. [65] develop an aerial robot to explore visually degraded, GPS-denied environments by fusing NIR camera and inertial data and build a high-resolution map by storing pointclouds, which is prohibitive to transmit across low-bandwidth communications infrastructure.

Many approaches leverage voxel-based occupancy modeling strategies for information-theoretic planning. Charrow et al. [16] develop a real-time information theoretic planning approach that maximizes the information gain in the environment by calculating the mutual information between a sensor observation and map represented as an occupancy grid. The occupancy grid map discretizes the environment into 3D voxels and stores a probability of occupancy in each voxel [29]. The downside of this map representation is that the memory demands become explosive as the environment to explore becomes larger or the desired resolution of the map becomes finer.

Oleynikova et al. [61] develop Voxblox to produce Truncated Signed Distance Fields (TSDFs) from Euclidean Signed Distance Fields (ESDFs) and demonstrate the results on an aerial robot. The approach outperforms Octomap [38]; however, while Voxblox may be dynamically resized, it is limited to a fixed-resolution voxel size.

2.4 Simultaneous Localization and Mapping

Whelan et al. [87] develop ElasticFusion for RGB-D simultaneous localization and mapping and represent the environment as a set of surfels. A joint cost function is employed to estimate pose with both geometric and photometric information from depth observations. To close the loop, the authors employ a deformation graph and maintain a list of active and inactive surfels. The approach does not consider LiDAR observations.

Behley and Stachniss [6] develop a SLAM formulation for 3D laser range data inspired by

ElasticFusion. The authors opt for a frame-to-model ICP method that registers a raw pointcloud to a surfel model. Loop closure detection proceeds by checking nearby poses against the current laser scan with ICP and rendering a view of the current map to determine if the loop closure leads to a consistent map given the current scan.

Chapter 3

Background

This chapter provides the mathematical foundations from which later chapters build. A mathematical description of the Gaussian Mixture Model is detailed in Section 3.1 that includes a description of the K-Means++ (Section 3.1.1) and Expectation Maximization (Section 3.1.2) algorithms to learn a GMM from data. Section 3.1.3 describes implementation details for efficient computation of these algorithms and Section 3.1.4 provides a performance analysis.

3.1 Gaussian Mixture Model

A GMM is a probability distribution that represents multivariate data as a weighted combination of M Gaussian distributions. The probability density of the GMM is represented as

$$p(\boldsymbol{x}|\boldsymbol{\xi}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)$$

where $\boldsymbol{x} \in \mathbb{R}^D$, π_m is a weight such that $0 \leq \pi_m \leq 1$, $\sum_m^M \pi_m = 1$, and $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda})$ is a D-dimensional Gaussian density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Lambda}$.

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}) = \frac{|\boldsymbol{\Lambda}|^{-1/2}}{(2\pi)^{D/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) \right)$$

The parameters of the distribution are compactly represented as $\boldsymbol{\xi} = \{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m\}_{m=1}^M$. Estimating the parameters of a GMM remains an open area of research [39]. Given the density function $p(\boldsymbol{x}|\boldsymbol{\xi})$ and observations $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x} \in \mathbb{R}^D$ assumed to be independent and identitically distributed with distribution p, the density for the samples is

$$p(\boldsymbol{X}|\boldsymbol{\xi}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\xi}) = \mathcal{L}(\boldsymbol{\xi}|\boldsymbol{X})$$

where $\mathcal{L}(\boldsymbol{\xi}|\boldsymbol{X})$ is called the likelihood function and the goal is to find the $\boldsymbol{\xi}^*$ that maximizes $\mathcal{L}$ [8]

$$\boldsymbol{\xi}^* = \arg\max_{\boldsymbol{\xi}} \ \mathcal{L}(\boldsymbol{\xi}|\boldsymbol{X})$$

It is analytically easier to maximize $\ln(\mathcal{L}(\boldsymbol{\xi}|\boldsymbol{X}))$, but the presence of the summation over m inside the logarithm make $\boldsymbol{\xi}$ difficult to compute and taking the derivative of this log likelihood function and setting to zero is made intractable because the resulting equations are no longer in closed form [9]. Instead, latent variables $b_{nm} \in \boldsymbol{B}$ are introduced that take a value of 1 if the sample $\boldsymbol{x}_n$ is in cluster m and 0, otherwise (called a 1-of-M coding scheme). A new likelihood function is defined $p(\boldsymbol{X}, \boldsymbol{B}|\boldsymbol{\xi}) = \mathcal{L}(\boldsymbol{\xi}|\boldsymbol{X}, \boldsymbol{B})$, called the complete data likelihood.

The Expectation step in Expectation Maximization finds the expected value of $\ln p(\boldsymbol{X}, \boldsymbol{B}|\boldsymbol{\xi})$ by the following function [8]

$$Q(\boldsymbol{\xi}, \boldsymbol{\xi}^i) = E\Big[\ln p(\boldsymbol{X}, \boldsymbol{B}|\boldsymbol{\xi})|\boldsymbol{X}, \boldsymbol{\xi}^i \Big]$$

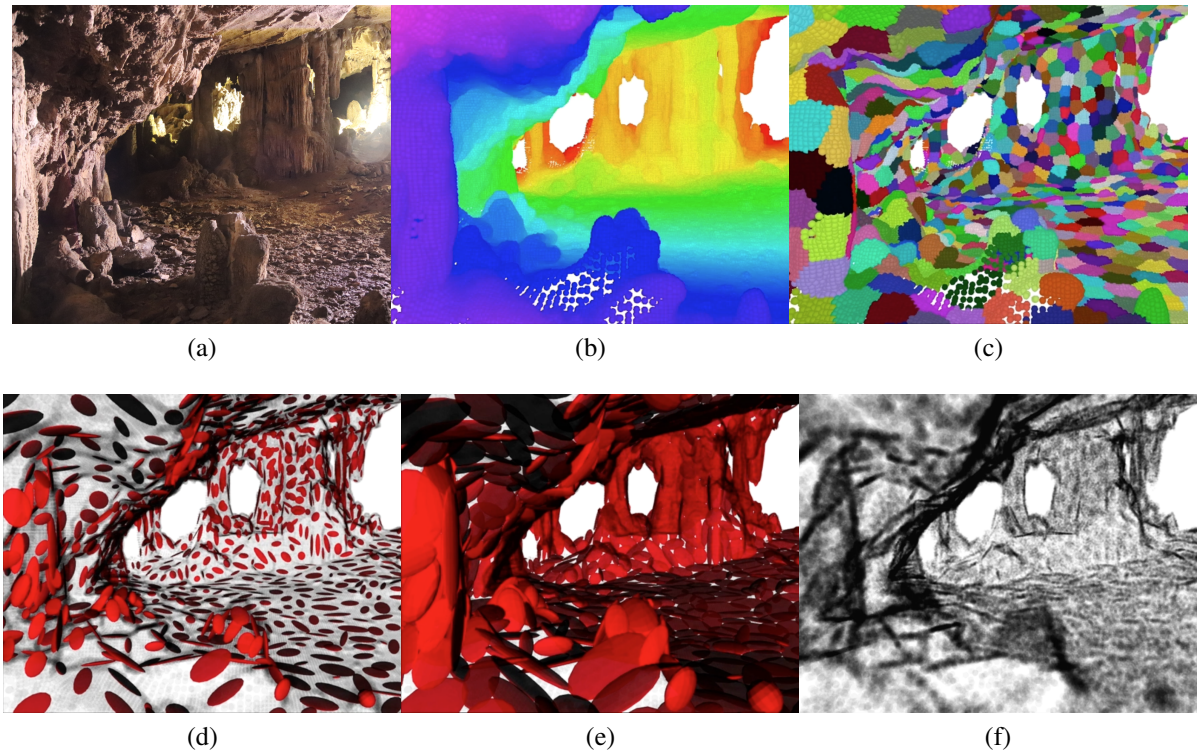


Figure 3.1: Overview of the process to create a GMM from a sensor observation. (a) illustrates an image taken from Rapps Cave in Greenbrier, WV. (b) is a corresponding pointcloud colored according to viewing distance (red is further away and blue is closer to the camera). (c) Each color corresponds to a cluster created during initialization via the K-Means++ algorithm. The cluster labels assigned by K-Means++ are used by EM to maximize the log likelihood of the data given the parameters. The red ellipsoids in (d) and (e) are 1-sigma and 2-sigma representations of the covariance matrices learned after running EM, respectively. Because the GMM is a generative model, the distribution can be sampled from to obtain the model shown in (f).

The Maximization step maximizes the expectation of the previous equation:

$$\boldsymbol{\xi}^{i+1} = \arg\max_{\boldsymbol{\xi}} Q(\boldsymbol{\xi}, \boldsymbol{\xi}^{i})$$

Each iteration of these steps is guaranteed to increase the log likelihood and ensure that the algorithm converges to a local maximum of the likelihood function [8].

3.1.1 Initialization

Kolouri et al. [52] find the EM algorithm to be sensitive to the choice of initial parameters and Jian

and Vemuri [45] prove that random initialization causes the EM algorithm to converge to a bad critical point with high probability. Because the EM algorithm does not guarantee convergence to a global optimum, the initialization is critical for convergence to a good stationary point. The work detailed in this thesis implements the K-Means++ algorithm [4], which differs from K-Means in the initial centroid assignment. The K-Means++ algorithm [4] proceeds in the following manner:

1a. Choose first center c_1 uniformly at random from $\boldsymbol{X}$.

1b. Select new center c_i from $\boldsymbol{x} \in \boldsymbol{X}$ with probability $\frac{D(\boldsymbol{x})^2}{\sum_{\boldsymbol{x} \in \boldsymbol{X}} D(\boldsymbol{x})^2}$, where $D(\boldsymbol{x})$ denotes the shortest distance from $\boldsymbol{x}$ to the closest cluster center already chosen $\{c_1, \ldots, c_{i-1}\}$.

1c. Repeat Step 1b. until k centers $\mathcal{C} = \{c_1, \ldots, c_k\}$ are selected.

2. For each $i \in 1, \ldots, k$ set cluster C_i to the set of points in $\boldsymbol{X}$ closer to c_i than they are to c_j for all $j \neq i$.

3. For each $i \in 1, \ldots, k$ set c_i to be the center of mass of all points in C_i: $c_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$.

4. Repeat steps 2 and 3 until $\mathcal{C}$ no longer changes.

Steps 1a.-1c. are unique to the K-Means++ algorithm; however, steps 2-4 come from the standard K-Means algorithm. The K-Means++ algorithm has advantages over the standard K-Means algorithm in that it provides approximation guarantees for the optimality of the algorithm that improves the speed and accuracy. Several variants of K-Means are proposed in the literature [22, 30, 37], but the work developed in Elkan [30] is leveraged in this thesis as it was found to achieve the best performance. Elkan [30] employs the triangle inequality and maintains lower and upper bounds on distances between points and centers to increase efficiency by avoiding costly distance calculations. A visualization of the clustering approach on the pointcloud in Fig. 3.1b may be seen in Fig. 3.1c.

3.1.2 Expectation Maximization

The EM algorithm proceeds with the following steps

1. Initialize $\boldsymbol{\mu}_m$, $\boldsymbol{\Lambda}_m$ and π_m with the method detailed in Section 3.1.1.

2. **E step**. Evaluate the responsibilities γ_{nm} using the current parameters $\boldsymbol{\mu}_m$, $\boldsymbol{\Lambda}_m$ and π_m:

$$\gamma_{nm} = \frac{\pi_m \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)}{\sum_{j=1}^{M} \pi_j \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Lambda}_j)} \tag{3.1}$$

3. **M step**. Estimate the new parameters $\boldsymbol{\mu}_m$, $\boldsymbol{\Lambda}_m$ and π_m using the current responsibilities, γ_{nm}.

$$\boldsymbol{\mu}_m^{i+1} = \frac{\sum_{n=1}^{N} \gamma_{nm} \boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_{nm}} \tag{3.2}$$

$$\boldsymbol{\Lambda}_m^{i+1} = \frac{\sum_{n=1}^{N} \gamma_{nm} (\boldsymbol{x}_n - \boldsymbol{\mu}_m^i)(\boldsymbol{x}_n - \boldsymbol{\mu}_m^i)^T}{\sum_{n=1}^{N} \gamma_{nm}} \tag{3.3}$$

$$\boldsymbol{\pi}_m^{i+1} = \sum_{n=1}^{N} \frac{\gamma_{nm}}{N} \tag{3.4}$$

4. Evaluate the log likelihood

$$\ln p(\boldsymbol{X} | \boldsymbol{\xi}) = \sum_{n=1}^{N} \ln \left(\sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m) \right) \tag{3.5}$$

   and check for convergence of either the parameters or the log likelihood. If convergence is not achieved, iterate again from step 2.

While the K-means algorithm performs a hard assignment of data points to clusters, the EM algo-

rithm makes a soft assignment based on posterior probabilities [9]. A visualization of the result of running EM on Fig. 3.1c may be seen in Figs. 3.1d and 3.1e.

3.1.3 Implementation Details

Efficient formulation of Eqs. (3.1) to (3.4) is critical for real-time viability on embedded systems. This section provides a computationally efficient formulation of the EM algorithm to enable real-time viability. Optimizations may be made in the E step by exploiting the logarithm of the responsibilities to convert the multiplications and divisions into sums and differences. Once the log of the responsibilities are calculated, the exponent of the result yields the correct responsibilities.

$$\gamma_{nm} = \frac{\pi_m \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)}{\sum_{j=1}^{M} \pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_j)} \tag{3.6}$$

$$\ln \gamma_{nm} = \ln \pi_m + \ln \left(\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m) \right) - \ln \left(\sum_{j=1}^{M} \pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_j) \right) \tag{3.7}$$

The logarithm of the multivariate Gaussian probability density function (Eq. (3.11)) may also be more efficiently computed by leveraging the inverse of the lower triangular cholesky decomposition (Eq. (3.9)) instead of the using the precision matrix as shown in Eq. (3.14) [14, 66].

$$\boldsymbol{\Lambda}_m = \boldsymbol{L}\boldsymbol{L}^T \tag{3.8}$$

$$\boldsymbol{P}_m = \boldsymbol{L}^{-1} \tag{3.9}$$

$$\boldsymbol{P}_m^T \boldsymbol{P}_m = \boldsymbol{\Lambda}_m^{-1} \tag{3.10}$$

$$\begin{aligned} &\ln \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m) \\ &= \ln \left((2\pi)^{-D/2} |\boldsymbol{\Lambda}_m|^{-1/2} \exp \left(-\frac{1}{2} (\boldsymbol{x}_n - \boldsymbol{\mu}_m)^T \boldsymbol{\Lambda}_m^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_m) \right) \right) \end{aligned} \tag{3.11}$$

$$= \ln\left((2\pi)^{-D/2}|\boldsymbol{\Lambda}_m|^{-1/2}\exp\Big(-\frac{1}{2}(\boldsymbol{x}_n-\boldsymbol{\mu}_m)^T\boldsymbol{P}_m^T\boldsymbol{P}_m(\boldsymbol{x}_n-\boldsymbol{\mu}_m)\Big)\right) \tag{3.12}$$

$$= -\frac{1}{2}\left(D\ln(2\pi)+\ln(|\boldsymbol{\Lambda}_m|)+(\boldsymbol{x}_n-\boldsymbol{\mu}_m)^T\boldsymbol{P}_m^T\boldsymbol{P}_m(\boldsymbol{x}_n-\boldsymbol{\mu}_m)\right) \tag{3.13}$$

$$= -\frac{1}{2}\left(D\ln(2\pi)+\sum_j\Big(\boldsymbol{P}_m(\boldsymbol{x}_n-\boldsymbol{\mu}_m)\Big)_j^2\right)+\Big(\sum_j\ln(\text{diag}(\boldsymbol{P}_m))_j\Big) \tag{3.14}$$

$\ln(|\boldsymbol{\Lambda}_m|^{-1/2})$ is equivalent to taking the sum of the logarithm of the diagonal entries of $\boldsymbol{P}_m$ as shown in: $\sum\ln(\text{diag}(\boldsymbol{P}_m))$. $(\boldsymbol{x}_n-\boldsymbol{\mu}_m)^T\boldsymbol{P}_m^T\boldsymbol{P}_m(\boldsymbol{x}_n-\boldsymbol{\mu}_m)$ is also equivalent to taking the sum of squares of the vector entries that result after computing $\boldsymbol{P}_m(\boldsymbol{x}_n-\boldsymbol{\mu}_m)$.

The implementation of the M step may be made slightly more efficient by noting that both the updated means and covariances, $\boldsymbol{\mu}_m^{i+1}$ and $\boldsymbol{\Lambda}_m^{i+1}$ rely on the unnormalized weights. Therefore, the calculation of the updated weights may be decomposed into two steps so that

$$\begin{aligned}\pi_m^* &= \sum_{n=1}^N\gamma_{nm}\\ \boldsymbol{\mu}_m^{i+1} &= \sum_{n=1}^N\frac{\gamma_{nm}\boldsymbol{x}_n}{\pi_m^*}\\ \boldsymbol{\Lambda}_m^{i+1} &= \sum_{n=1}^N\frac{(\boldsymbol{x}_n-\boldsymbol{\mu}_m^{i+1})(\boldsymbol{x}_n-\boldsymbol{\mu}_m^{i+1})^T\boldsymbol{\gamma}_{nm}}{\pi_m^*}\end{aligned}$$

before normalizing the weights:

$$\pi_m^{i+1} = \frac{\pi_m^*}{N}$$

This formulation eliminates redundant calculations when computing $\boldsymbol{\mu}_m^{i+1}$ and $\boldsymbol{\Lambda}_m^{i+1}$.

3.1.4 Performance Analysis

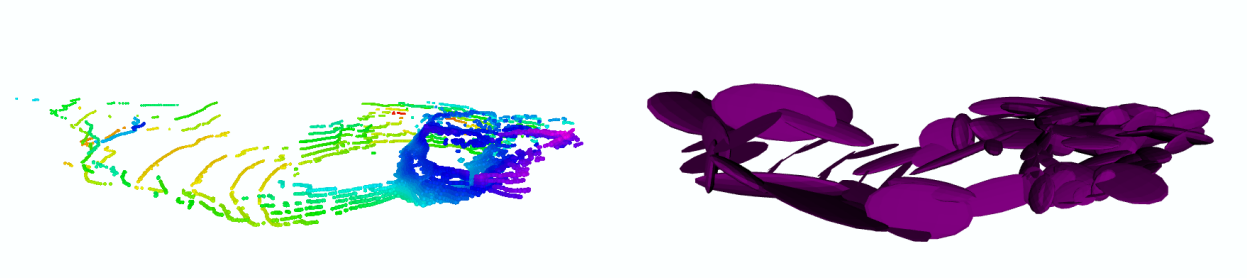
An analysis is conducted for sensor observations from three different environments illustrated in Fig. 3.2. The Gaussian Mixture Model formulation discussed in this chapter is benchmarked

Sensor Obs.	Num. Points	KMeans++ Initialization (s)	KMeans (Elkan) (s)	KMeans++ Total (s)	E Step (s)	M Step (s)	EM Total (s)	KMeans++ & EM Total (s)
Cave	27764	0.248	0.924	1.172	0.734	0.502	1.236	2.408
Mine	17843	0.150	0.420	0.570	0.410	0.294	0.704	1.274
TUM	241399	2.570	3.070	5.640	13.150	5.500	18.650	24.290

Table 3.1: Runtimes for each stage of the Gaussian Mixture Model pipeline. The first column details the dataset corresponding to the stastics in the other columns. The second column provides the number of points in each observation. Note that the more points in the sensor observation, the longer the runtime. For all sensor observations, the runtime of EM exceeds the runtime of the initialization procedure (i.e., KMeans++ Total).

in C++. For each sensor observation, a probability distribution is learned consisting of 100 components. The code is profiled on a late-2013 15" Macbook Pro with a $2.60\,\mathrm{GHz}$ Intel Core i7-4960HQ CPU and $16\,\mathrm{GB}$ RAM. All profiling is run single-threaded.

Table 3.1 illustrates the profiling results. The columns of the table consist of the sensor observation under consideration, KMeans++ profile results, EM profile results, and total time cost. For all sensor observations, the cost of running EM exceeds the cost of initialization using KMeans++. The time to learn a GMM from the TUM RGB-D sensor observation is larger than for the LiDAR observations due to the significantly larger number of points in the sensor observation. The EM algorithm is computationally expensive for a GMM with M components and a dataset with N samples, the size of the responsibility matrix is $N \times M$ (Eq. (3.1) defines how to compute one entry in the responsibility matrix). All N samples from the dataset are used to update the weights, means, and covariances in the GMM (Eqs. (3.2) to (3.4)). Techniques are detailed in later chapters to reduce the computational complexity of learning a GMM from sensor observations.

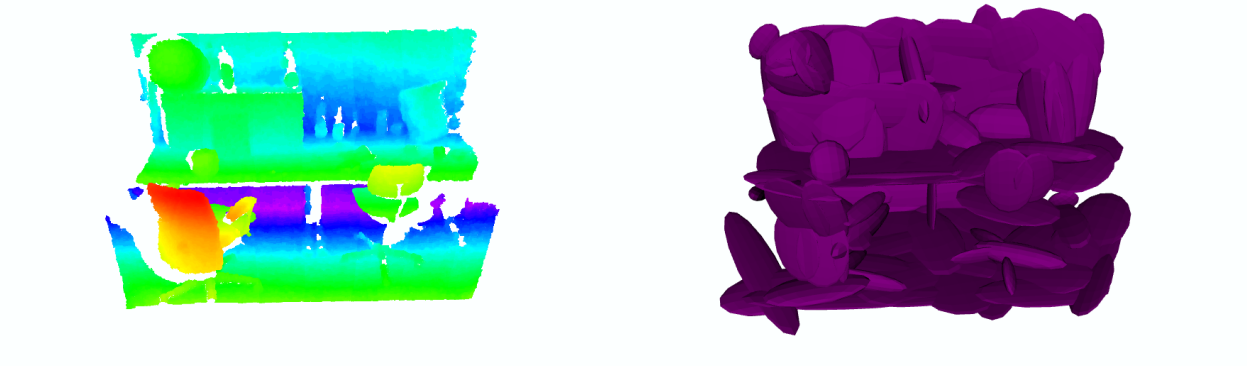


(a) Cave LiDAR observation

(b) Cave GMM

(c) Mine LiDAR observation

(d) Mine GMM



(e) TUM RGB-D observation

(f) TUM GMM

Figure 3.2: (a) A 3D laser scan of a cave consisting of 27,764 points and corresponding GMM (shown in (b)). (c) illustrates a LiDAR observation taken in a mine consisting of 17,843 points and the corresponding GMM is shown in (d). A RGB-D observation in an office environment consisting of 241,399 points and corresponding GMM shown in (f). All GMMs consist of 100 mixture components. All sensor observations in the left-hand column are displayed in the sensor frame, but note that the sensor frame of the RGB-D sensor points in the direction of +z. Colors correspond to z-height.

Chapter 4

Information-Theoretic Exploration

This chapter presents a baseline information-theoretic exploration framework that leverages occupancy grid maps to represent the environment and quantify the mutual information between the map and potential sensor observations. A 3D state-space lattice is constructed from which dynamically-feasible trajectories are sampled to estimate the information-theoretic reward. This exploration formulation is designed to run in real-time on a constrained computer and serves to demonstrate the viability of the proposed method for deployment to hardware in later chapters.

4.1 Approach

An overview of the relevant system model (a quadrotor micro air vehicle) is presented in Section 4.1.1 and the motion planning approach is detailed in Section 4.1.2. Section 4.1.3 presents the map representation, restates results from Charrow et al. [16] in the context of the exploration of 3D environments, and presents a method to compress the map representation for increased computational efficiency while preserving relevant information [59]. Section 4.1.4 presents the integrated exploration strategy toward generation of real-time trajectories that are both energy-efficient and informative.

4.1.1 System Model

A quadrotor system model is presented with reference to notation and details by Mahony et al. [53] and used throughout this work; however, the model and proposed approach may be readily adopted to other aerial or terrestrial vehicle models.

The quadrotor with position (r), radial velocity (ω), mass (m), and inertia ($\mathcal{I}$), moves according to the Newton-Euler equations

$$F = m\ddot{r} \quad \tau = \mathcal{I}\dot{\omega} + \omega \times \mathcal{I}\omega \tag{4.1}$$

where the net forces and moments acting on the system are F and τ, respectively. Here, $F = F_a + F_g$ is the sum of applied and gravitational forces. As the rotors are aligned with the body z-axis, $F_a = R \begin{bmatrix} 0 & 0 & f_a \end{bmatrix}^{\mathsf{T}}$ where R is the rotation matrix between the body and inertial frames.

The rotors have angular velocities $\varpi_{1:4}$ and spin in alternating directions. By the static thrust assumption [12, 53] the rotors produce thrusts and torques,

$$t_i = c_t \varpi_i^2, \quad q_i = c_q \varpi_i^2 \tag{4.2}$$

respectively, while c_t and c_q are associated scaling factors. From (4.2), the rotor torques and thrusts are related to the body force and moments by

$$\Phi = \Gamma \Omega \tag{4.3}$$

where $\Phi = \begin{bmatrix} f_a & \tau_{1:3} \end{bmatrix}^{\mathsf{T}}$, $\Omega = \begin{bmatrix} \varpi_{1:4}^2 \end{bmatrix}^{\mathsf{T}}$, and Γ is the mixer matrix [53]. This quadrotor model is known to be differentially flat [32].

4.1.2 Trajectory Generation and Motion Primitive Graph Planning

The trajectories are generated as time-parameterized 5^{th} order polynomials with boundary value constraints that minimize the integral of jerk squared for the four flat outputs, x, y, z, and ψ using the method developed by Mueller et al. [58].

Similar to the approach proposed by Pivtoraiko et al. [68], this work employs a state-space lattice, $\hat{X}$, that consists of a collection of *motion primitives*, M. A motion primitive is computed given two boundary constraints $\hat{x}_i, \hat{x}_j \in \hat{X}$, $i, j \in \{0, \ldots, N-1\}$ with $i \neq j$. A boundary constraint $\hat{x}_i$ is specified with the flat outputs and their derivatives, $(r, \psi, \dot{r}, \dot{\psi}, \ddot{r}, \ddot{\psi})$. To represent motion primitives in a state-space lattice, the boundary constraints are stored in N uniquely identified *nodes*. Figure 4.1a illustrates three motion primitives that form a *dictionary*. Each motion primitive drives the vehicle from some initial boundary condition to some final boundary condition. Motion primitives are stored as edges between nodes in the dictionary, $D = (\hat{X}, M)$. Feasibility of a primitive is ensured by imposing limits on the maximum acceleration and velocity over the duration of the primitive.

Two optimizations are made to the approach in [68]. 1) Constraints are imposed to ensure all motion primitives begin and end at lattice states with pre-defined velocity and acceleration in the body frame x-axis. These constraints ensure that the vehicle flies in the body-direction to enable obstacle detection via sensors that point in the direction of forward motion. 2) The motion primitives are formulated in the body frame with the heading at the start of each motion primitive set to zero with respect to the body frame. These optimizations significantly reduce the size of the search space over the motion primitives and reduce computation as the body-frame graph is reused at every planning iteration.

Motion Primitive Graph A dictionary of motion primitives (see Fig. 4.1a) can be reused to create 2D and 3D graphs as depicted in Figs. 4.1b and 4.1c by appending the dictionary to leaf

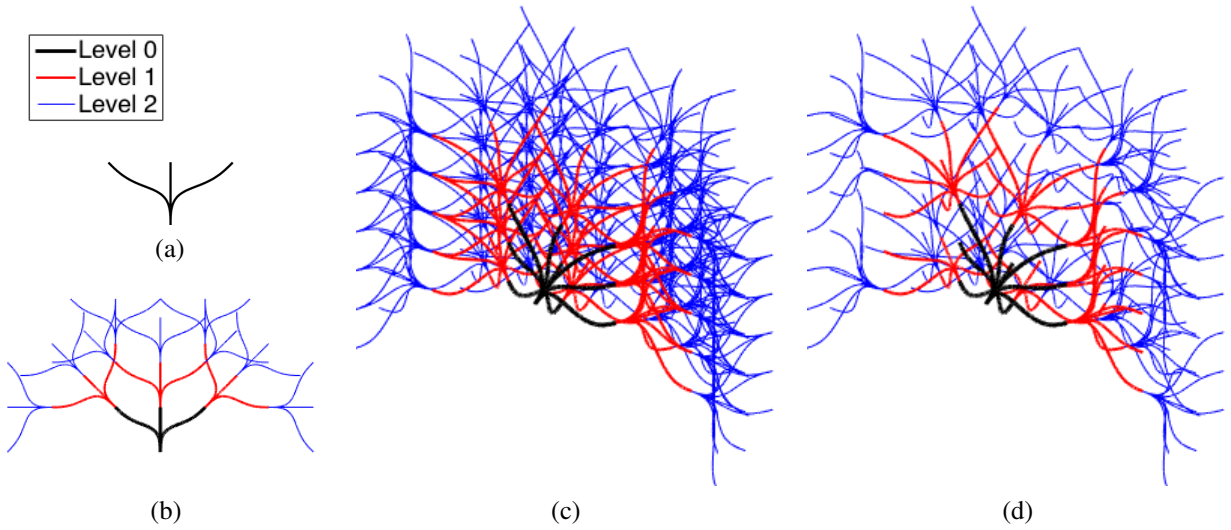


Figure 4.1: (a) A dictionary of three motion primitives in 2D. (b) A graph with depth three constructed from the dictionary. (c) A graph in 3D constructed from a dictionary of ten motion primitives (associated dictionary not shown). (d) The graph from (c) after pruning. Sub-optimal and redundant edges are pruned to decrease the number of primitives to search over during exploration.

nodes up to a specified depth. Each successive level is constructed from the same dictionary. For example, the space covered in Fig. 4.1b is formed by repetition of the dictionary in Fig. 4.1a to a depth of 3.

As the expansion of nodes at run-time is computationally expensive, graphs are pre-computed to enable fast search through many trajectories during exploration. One graph is computed for each possible initial state consisting of velocity, acceleration, and jerk. The resulting graph may contain tens of thousands (or millions) of vertices. Therefore, *Dijkstra*'s algorithm, a single-source shortest path algorithm, is employed to prune the graph [20]. The result is a minimum spanning tree that contains the lowest cost trajectories from the root vertex to any other vertex in the graph (see Fig. 4.1d). The refined graph is pre-computed with linear query time lookups in the worst case.

4.1.3 Mutual Information and Mapping

Uncertainty in a probability distribution is quantified through entropy measures. The most common and the only one satisfying all of Shannon's axioms is the Shannon entropy [21],

$$H(X) = -\int p(x) \log p(x) dx. \tag{4.4}$$

The Shannon mutual information is the expected reduction in entropy of one random variable from observation of another

$$I_S(X, Y) = H(X) - H(X|Y). \tag{4.5}$$

This can also be defined in terms of the Kullback-Leibler divergence [21, 69] which describes the difference between two probability distributions:

$$D_{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x)} dx, \tag{4.6}$$

$$I_S(X, Y) = D_{KL}(p(X, Y)||p(X)p(Y)). \tag{4.7}$$

This leads to the definition of mutual information measures based on other divergences such as Cauchy-Schwarz quadratic mutual information (CSQMI)

$$I_{CS}(X, Y) = -\log \frac{(\iint p(x, y)p(x)p(y)dydx)^2}{\iint p(x, y)^2 dxdy \iint p(x)^2 p(y)^2 dxdy} \tag{4.8}$$

which has integrals inside the logarithm that can be computed analytically and efficiently [15, 69].

Mutual Information for Occupancy Grids The environment is modeled with a 3D occupancy grid (OG), a computationally efficient representation (see Fig. 4.2) [29, 85]. Charrow et al. [16]

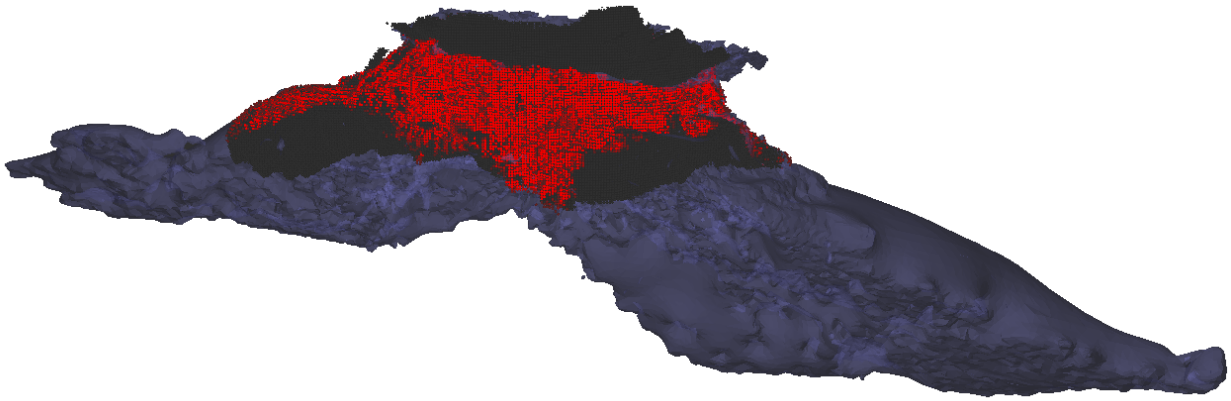
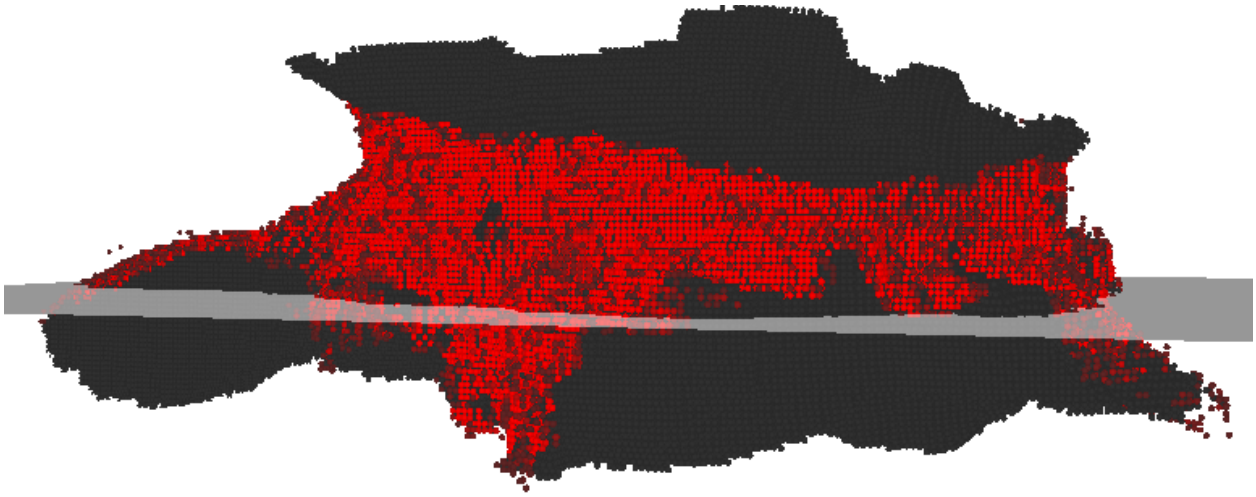


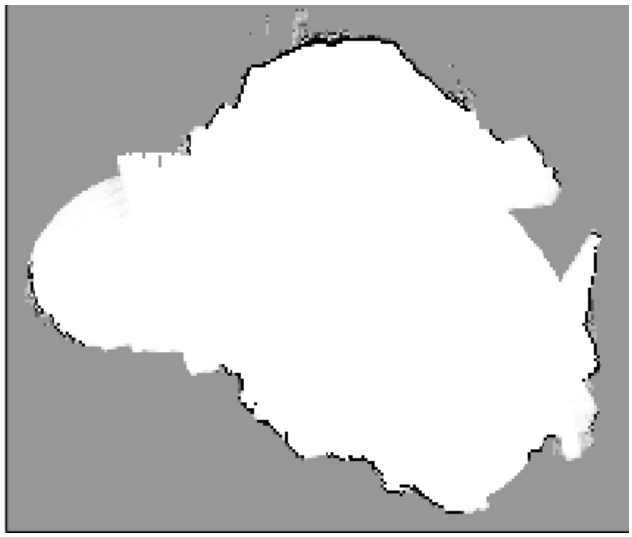
Figure 4.2: A partially explored OG from a simulation trial with occupied (red) and free (black) cells and $0.1\,\mathrm{m}$ *resolution.*

present a closed form solution to CSQMI for a single ranging measurement with an approximate formulation of $O(n)$ time-complexity for sets of conditionally dependent measurements from generic multi-beam ranging sensors where n is the number of cells intersected. Figure 4.3 depicts a representative CSQMI reward distribution for sensor observations over a slice of the OG originally shown in Fig. 4.2. Although the distribution over the entire space is shown, the CSQMI is only evaluated for sensor measurements generated during execution of trajectories considered during planning. Note that the evaluation of CSQMI also considers conditional dependencies between sensor measurements that result in a value less than or equal to the sum of the CSQMI of individual measurements considered in isolation.

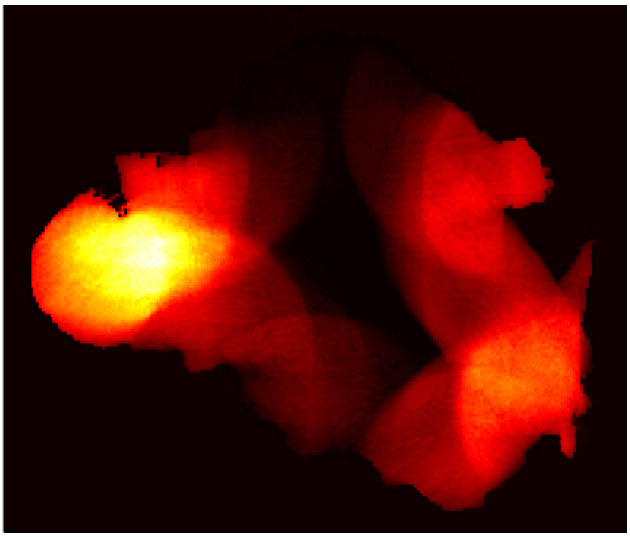
To obtain a rule for computing reduced resolution representations of OGs for use in CSQMI calculation (not planning), Nelson and Michael [59] employ the Principle of Relevant Information (PRI) [69], a minimization over a random variable X of $(H(X) + \lambda D(X||X_0)$, where H is a measure of entropy and D is a divergence measure. This approach produces a probability distribution with reduced entropy while minimizing the difference from the original distribution. Using $\lambda = 1$, they obtain a rule that maps a set of cells to a single probability that is any of $(0, 1, \frac{1}{2})$ based on the product of the odds ratios. Although initially evaluated on 2D OGs, this approach readily extends to 3D OGs as evident when compressing the map in Fig. 4.2 and shown in Fig. 4.4. This rule is ap-



(a) 3D OG and intersecting plane

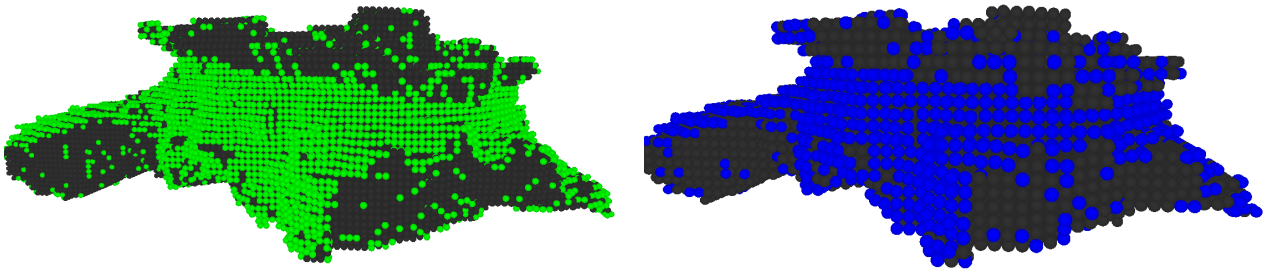


(b) 2D OG



(c) CSQMI

Figure 4.3: Illustrative cross-section (a) from the OG shown in Fig. 4.2 to form a 2D OG (b) with free (white), occupied (black), and unknown (gray) cells. (c) The CSQMI reward is evaluated for a planar sensor at different positions over this cross section to create a heat map (brighter colors corresponding to increased reward). This reward surface is non-smooth and sometimes flat due to occlusions and the limited sensor range.



(a) $n = 1$ (b) $n = 2$

Figure 4.4: The OG from Fig. 4.2 with resolution reduced (a) one and (b) two times. Free cells are dark and occupied cells are colored while unknown cells are not shown.

plied recursively such that the cell dimension at reduction level n increases by a factor of $\alpha = 2^n$. To encourage similar behavior between compression levels, we normalize the map prior, p, such that the expected penetration distance of a ray into a row of unknown cells remains constant on a reduced resolution OG. By equating the expected penetration distance in the base and compressed maps

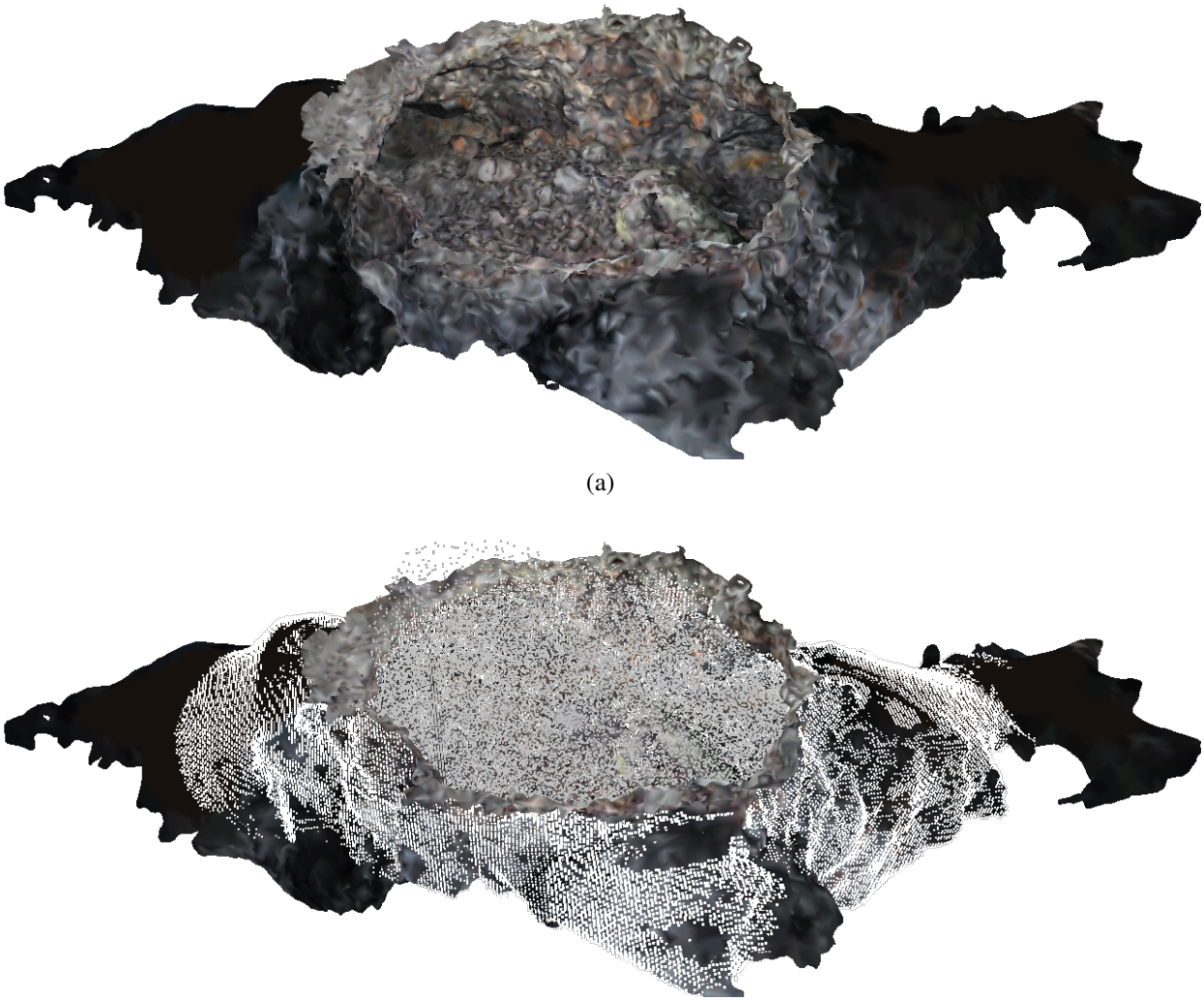
$$\sum_{n=1}^{\infty} \alpha p_\alpha n(1-p_\alpha)^n = \sum_{n=1}^{\infty} pn(1-p)^n \tag{4.9}$$

a formula is obtained that relates the occupancy prior of the compressed map, p_α to the prior of the base map, p_0,

$$p_\alpha = \frac{\alpha p_0}{1 - p_0 + \alpha p_0}. \tag{4.10}$$

4.1.4 Integrated Exploration Approach

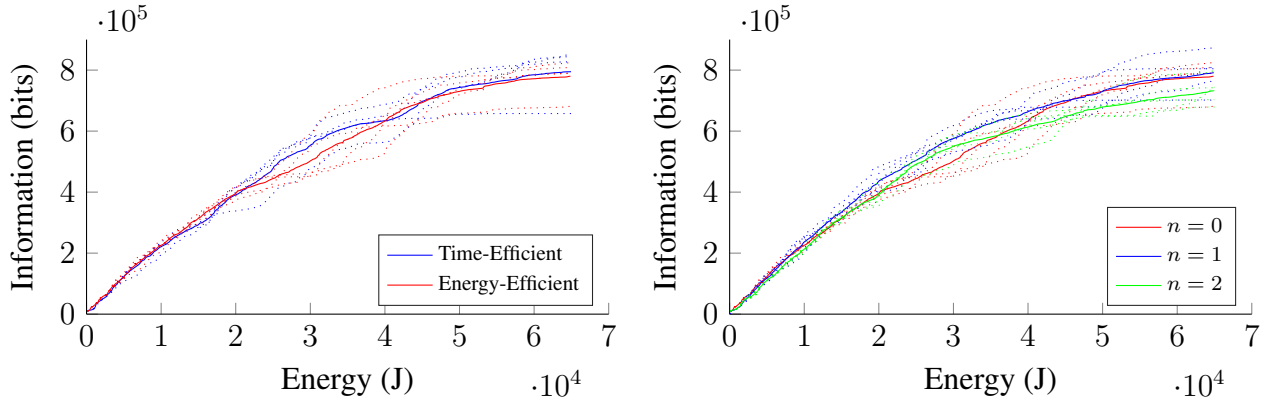
To evaluate plans, two objectives are considered: 1) time-efficient exploration, $\mathbf{V_T} = \frac{\mathbf{I}}{\mathbf{T}}$, and 2) energy-efficient exploration, $\mathbf{V_E} = \frac{\mathbf{I}}{\mathbf{E}}$. Both objectives compute information reward with respect



(a)

(b)

Figure 4.5: (a) Angled view of the Indian Tunnel skylight and (b) sensor measurements taken during exploration (shown in white).



(a) Time-Efficient vs. Energy-Efficient

(b) Reduced Resolution

Figure 4.6: Information gained (reduction in Shannon entropy) versus time and energy costs. (a) Energy- and time-efficient objectives are roughly equivalent for the operational domain. (b) Evaluation of mutual information on reduced resolution maps leads to improved exploration performance. Dotted lines indicate individual runs and solid lines represent mean values.

to the total duration of a plan, $\mathbf{T}$, or the integral of power usage, P_e denoted by $\mathbf{E}$.[1] It will be shown that these objectives produce nearly identical performance for the operating conditions detailed in this paper as power usage remains roughly constant. As the calculation of information reward, $\mathbf{I}$, is computationally expensive, we refer to the heuristic proposed in earlier work [16] and compute the mutual information at motion primitive endpoints on the compressed map.

At each exploration update, the information reward and expenditure cost are computed for a subset of plans in the graph given available computation time, typically on the order of 10^5 plans per exploration cycle (e.g., 1 Hz). Each motion primitive evaluation consists of referencing pre-computated traversal costs, collision-checking, as well as computation of associated information gain. A trajectory is selected according to the objectives noted above in order to yield a maximally

[1] The total energy consumption for the system is defined as

$$P_e = \sum_{i=1}^{4} \left(c_q \varpi_i^3 + \frac{r c_q^2}{k_e^2} \varpi_i^4 \right) + P_a$$

where r is the motor internal resistance, k_e the motor torque and voltage constant, and P_a a lumped parasitic term. We defer to the works of Bangura et al. [5] and Morbidi et al. [57] for more extensive discussion of quadrotor power models.

informative and efficient exploratory motion plan.

4.2 Results

Simulation design The approach is tested with a simulated quadrotor exploring the Indian Tunnel skylight environment. The simulation captures the power model and rotor dynamics of the vehicle.

The quadrotor model parameters are reported in Table 4.1. Parameters of the motion primitive graph are detailed in Table 4.2 and specify the range and discretization of each value. The motion primitive state-space lattice specifies 3D motions and has a depth of 4. The set of motion primitives used to build the graph requires 33MB of space and is thus large enough that the graph must be pre-computed but small enough that it can be computed in under 10 minutes on a desktop class processor. The approach applies generally to systems with one or more ranging sensors. A time-of-flight camera is simulated that produces a 24×38 (reduced to $6 \times 9 = 54$ beams for CSQMI calculation) depth-image with a $43.6° \times 34.6°$ field-of-view and $10\,\mathrm{m}$ range. The sensor is aligned with the x-axis with the longer dimension of the field-of-view aligned vertically for more effective scanning (yaw) behavior. The implementation of the CSQMI computation is optimized but remains the most expensive operation during planning.

The approach is evaluated by rate and quantity of of information gain (reduction in Shannon entropy, (4.4)). versus energy expended. These metrics are appropriate for planetary robots that explore an environment given finite energy reserves rather than complete exploration.

Results Trials comparing objectives $\mathbf{V_T}$ and $\mathbf{V_E}$ are shown in Fig. 4.6a. In all of the experiments, results are shown for ten minutes of exploration that expend roughly the capacity of a single $2250\,\mathrm{mA\,h}$ battery. The choice of primitives limits linear acceleration to $0.5\mathrm{m\,s^{-2}}$ to ensure feasibility and sensing such that the control inputs are dominated by gravity compensation. Power usage is constant, making the choice of objective inconsequential. Thus, time cost is an ap-

propriate substitute for energy cost under typical conditions barring sufficiently aggressive flight. In Fig. 4.6b results are shown for reduced resolutions as shown in Fig. 4.4. Performance benefits are observed (summarized in Table 4.3) primarily for $n = 1$ which performs consistently well throughout the duration of the experimental trials. For $n = 2$, degradation in performance appears due to aggressive reduction in the resolution of the OG signifying a trade-off between the trajectory evaluation rate and fidelity. These results are summarized in Table 4.3. Figure 4.5 shows the pit mesh and cloud of empty points after exploration and highlights the extent of coverage for a typical experimental trial.

Table 4.1: System Parameters

m	$0.507\ \mathrm{kg}$	k_e	$4.0 \cdot 10^{-3}\ \mathrm{V\,s\,rad^{-1}}$
c_t	$1.158 \cdot 10^{-6}\ \mathrm{N\,m\,s^2\,rad^{-2}}$	r	$0.125\ \Omega$
c_q	$1.969 \cdot 10^{-8}\ \mathrm{N\,m\,s^2\,rad^{-2}}$	P_a	$8.4\ \mathrm{W}$

Table 4.2: State-Space Lattice Parameterization

	start	end	Δ		start	end	Δ
x (m)	0	1	1	ψ (rad)	$\frac{-\pi}{2}$	$\frac{\pi}{2}$	$\frac{\pi}{4}$
y (m)	-1	1	1	$\|\|v\|\|$ (m/s)	0	0.5	0.25
z (m)	-1	1	1	$\|\|a\|\|, \dot{z}, \ddot{z}, \dot{\psi}, \ddot{\psi}$	0	0	-

	CSQMI Rates (kHz)		Information (kilo-bits)		
n	View	Plan	20kJ	40kJ	60kJ
0	2.03	0.58	397	633	771
1	3.11	0.90	**438**	**665**	**776**
2	5.42	1.55	393	614	716

Table 4.3: Reduced Resolution Exploration Performance

4.3 Conclusions

This chapter demonstrated an autonomous, exploration and mapping framework for unstructured cave and pit environments that operates quickly and efficiently in 3D while avoiding hazards. Real-time motion planning is achieved via a finite-horizon approach in the form of multiple state-space

lattice graphs. A computationally tractable, information-theoretic objective function based on the evaluation of CSQMI at each lattice state combined with compression techniques enables the evaluation of thousands of views per second. The framework is viable for real-time exploration and, given the presented operating conditions, that the time-efficient and energy-efficient approaches yield equivalent performance. The compression strategy ensures real-time viability and enables computationally tractable, real-time exploration.

Chapter 5

On-Manifold GMM Registration

While the previous chapter enables real-time exploration performance given known estimates of state, there are several limitations in converting the raw sensor observations into a grid map. First, the discretization of the imposed grid structure reduces the fidelity of the resulting model. Furthermore, real-world deployment of these algorithms must mitigate the effects of pose estimation drift and enable local and global consistency in the map. To overcome the limitations of grid-based mapping techniques, this chapter presents a method to model sensor observations as Gaussian Mixture Models (GMMs) and estimate pose via an on-manifold distribution to distribution registration formulation that computes an optimal transform between two GMMs by minimizing the squared L2 norm. The method is evaluated with real-world datasets of LiDAR observations in subterranean environments and RGB-D observations of a cluttered office environment.

5.1 Approach

Figure 5.1 illustrates the proposed approach. First, pointclouds (Fig. 5.1a) are converted to full anisotropic GMMs (Fig. 5.1b). Next, Section 5.1.1 formulates an objective function for registration from the correlation integral that maximizes the overlap between two distributions. Section 5.1.2 derives the closed-form equations for the gradient and Hessian. Section 5.1.3 details the optimiza-

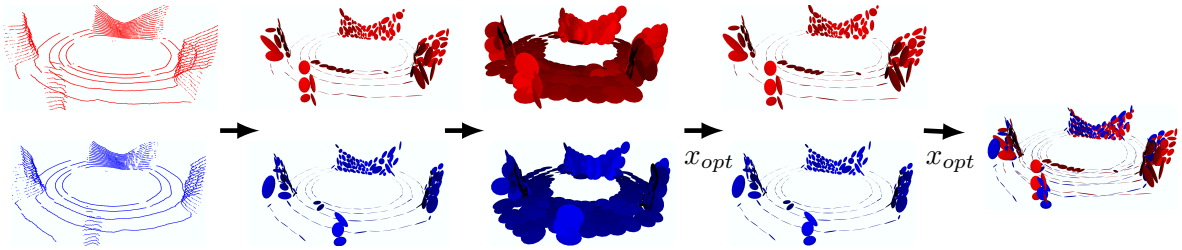


Figure 5.1: Overview of methodology: (a) Pointclouds taken at two different poses (shown in red and blue) are converted into (b) GMMs with anisotropic covariances. The covariances are converted to (c) isotropic-planar (isoplanar) and an optimization is run that minimizes the squared L2 norm between the two distributions. The output of the optimization x_{opt} is used to seed a second optimization with anisotropic covariances. The results are rigid transformation parameters that align the GMMs (shown in (e)).

tion framework that optimizes first with GMMs with isoplanar covariances (Fig. 5.1c) that serves to smooth the cost function and reduce local minima followed by a refinement optimization using the original anisotropic covariances (Fig. 5.1d) to obtain accurate registration parameters and align the distributions (Fig. 5.1e).

5.1.1 Registration

GMM to GMM registration, commonly referred to as distribution to distribution (D2D) registration, is achieved by aligning modeled probability densities. Given two GMMs, $\mathcal{G}_0(\boldsymbol{x})$ and $\mathcal{G}_1(\boldsymbol{x})$, with potentially different numbers of components, the transformation parameters $\boldsymbol{\theta}$, are sought that minimize the *distance* between $\mathcal{G}_0(\boldsymbol{x})$ and $T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta})$. $\mathcal{G}_0(\boldsymbol{x})$ and $\mathcal{G}_1(\boldsymbol{x})$ are defined as:

$$\mathcal{G}_0(\boldsymbol{x}) = \sum_{m}^{M} \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)$$

$$\mathcal{G}_1(\boldsymbol{x}) = \sum_{k}^{K} \tau_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\nu}_k, \boldsymbol{\Omega}_k)$$

and $T(\cdot, \boldsymbol{\theta})$ is the rigid transformation function consisting of a rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$. A transformed GMM with parameters $\{\tau_k, \boldsymbol{\nu}_k, \boldsymbol{\Omega}_k\}$ is expressed as

$$T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta}) = \sum_{k=1}^{K} \tau_k \mathcal{N}(\boldsymbol{x} | \boldsymbol{R}\boldsymbol{\nu}_k + \boldsymbol{t}, \boldsymbol{R}\boldsymbol{\Omega}_k\boldsymbol{R}^T) \tag{5.1}$$

Following the success of D2D alignment for NDT-map [80] and in the original work demonstrating GMMs for point set alignment [45], the squared L2 norm is employed as the measure of distance between the target GMM $\mathcal{G}_0(\boldsymbol{x})$ and the transformed source GMM $T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta})$. The cost function to minimize may be written as:

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg\min_{\boldsymbol{\theta}} \int \|\mathcal{G}_0(\boldsymbol{x}) - T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta})\|_2^2 \mathrm{d}\boldsymbol{x} \\ &= \arg\min_{\boldsymbol{\theta}} \int \left(\|\mathcal{G}_0(\boldsymbol{x})\|_2^2 + \|T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta})\|_2^2 - 2\mathcal{G}_0(\boldsymbol{x}) T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta}) \right) \mathrm{d}\boldsymbol{x} \end{aligned} \tag{5.2}$$

Eqn. (5.2) may be simplified by noting that the first term does not depend on the transformation parameters $\boldsymbol{\theta}$ and that energy in a distribution (which is given by the second term) is invariant under rigid transformation. Intuitively, this means that application of the transformation parameters does not change the shape of the distribution. Thus, the minimization of the squared L2 distance corresponds to maximization of the correlation of the two distributions:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \int 2\mathcal{G}_0(\boldsymbol{x}) T(\mathcal{G}_1(\boldsymbol{x}), \boldsymbol{\theta}) \mathrm{d}\boldsymbol{x} \tag{5.3}$$

The objective function (5.3) has the same extrema as that of the Cauchy-Schwarz divergence [48], due to the monotonicity of the log function. In the case of two GMMs, the correlation integral of (5.3) has a closed-form solution given by [48]

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_{m=1}^{M} \sum_{k=1}^{K} \pi_m \tau_k \mathcal{N}(\boldsymbol{\mu}_m | \boldsymbol{R}\boldsymbol{\nu}_k + \boldsymbol{t}, \boldsymbol{\Lambda}_m + \boldsymbol{R}\boldsymbol{\Omega}_k\boldsymbol{R}^T) \tag{5.4}$$

which may be written as

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} -F, \tag{5.5}$$

where

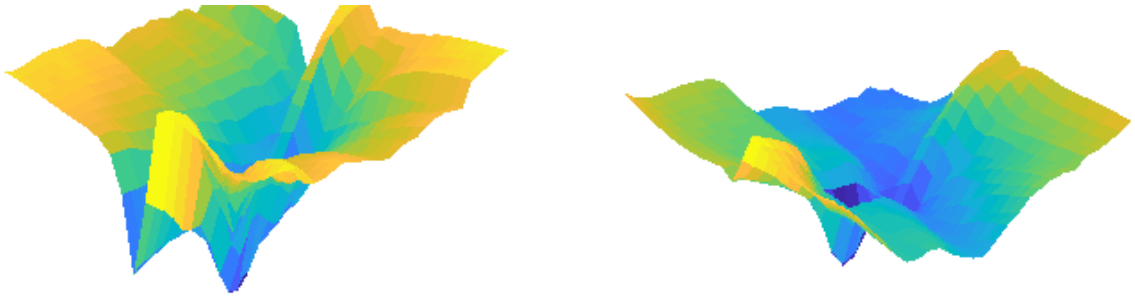
$$\begin{aligned} F &= \sum_{m=1}^{M}\sum_{k=1}^{K} f_{mk} \\ f_{mk} &= \pi_m \tau_k \frac{|\boldsymbol{\Sigma}_{mk}|^{-1/2}}{(2\pi)^{3/2}} \exp\left(-\frac{1}{2}\boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk} \right) \\ \boldsymbol{y}_{mk} &= \boldsymbol{\mu}_m - \boldsymbol{R}\boldsymbol{\nu}_k - \boldsymbol{t} \\ \boldsymbol{\Sigma}_{mk} &= \boldsymbol{\Lambda}_m + \boldsymbol{R}\boldsymbol{\Omega}_k\boldsymbol{R}^T \end{aligned}$$

The goal is to maximize the correlation, which is an integral whose value depends on the transformation parameters $\boldsymbol{\theta}$. Using [45, 48], the correlation integral may be cast as the minimization problem in (5.5) that finds the alignment that maximizes the overlap between the distributions. The optimal rigid transformation parameters may be solved for by setting the gradient of (5.5) to zero. The Hessian contains information about the local curvature of the function, so it increases the rate of convergence of the optimization. The gradient and Hessian are derived in closed-form in Section 5.1.2. An unconstrained optimization problem is cast that optimizes objective (5.5) via a trust-region method (Section 5.1.3).

The cost function is distinct from that of [80], in that the role of the determinant is explicitly accounted for in the cost function. That is, f_{mk} becomes

$$\pi_m \tau_k \frac{1}{(2\pi)^{3/2}} \exp\left(-\frac{1}{2}\boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk} \right) \tag{5.6}$$

when the determinant is removed. The determinant acts to increase the weight applied to compo-



(a) (b)

Figure 5.2: The non-convexity of the cost function is evident. The cost function with the determinant (a) exhibits a wider basin of attraction than the cost function without (b).

nents that have high certainty such that these components have greater attractive forces. The effect of the determinant on the shape of the cost function is shown in Fig. 5.2. Retaining the determinant in the cost function increases the complexity of the expression derived for both the gradient and the Hessian, however, empirical results suggest that inclusion of the determinant increases robustness.

The objective (5.5) is highly non-convex due the the nature of the GMM, as is demonstrated empirically in Fig. 5.2. Furthermore, the optimization domain $\mathrm{SE}(3)$, which is the product manifold $\mathrm{SO}(3) \times \mathbb{R}^3$, is non-convex due to the non-convexity of $\mathrm{SO}(3)$. The structure of the manifold can be better captured by leveraging a minimal axis-angle parameterization that uniquely defines a rotation through the exponential map. This reparmeterization enables the use of more straightforward, unconstrained optimization techniques. Details of the rotation parameterization may be found in Appendix A.

5.1.2 Gradient and Hessian

The gradient on the manifold $\mathbb{R}^3 \times SO(3)$ is derived and has coordinates $\boldsymbol{\xi} = (\boldsymbol{t}, \boldsymbol{u})$. For brevity, the partial derivative is expressed as $\boldsymbol{R}_{\xi_a} = \frac{\partial \boldsymbol{R}}{\partial \xi_a}$. The derivatives $\boldsymbol{R}_{u_a}$ and $\boldsymbol{R}_{u_b u_a}$ are given in Appendix A. For the remaining derivatives, $\boldsymbol{R}_{t_a} = \mathbf{0}$, $\boldsymbol{t}_{u_a} = \mathbf{0}$, $\boldsymbol{t}_{t_a} = \boldsymbol{e}_a$, $\boldsymbol{t}_{t_b t_a} = \mathbf{0}$ and both mixed

second derivatives are zero. The gradient is given by

$$\frac{\partial F}{\partial \xi_a} = \sum_{m=1}^{M} \sum_{k=1}^{K} f_{mk}(d_{mk}^{(a)} + q_{mk}^{(a)}) \tag{5.7}$$

where

$$d_{mk}^{(a)} = -Tr\left\{\boldsymbol{\Omega}_k \boldsymbol{R}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{R}_{\xi_a}\right\} \tag{5.8}$$

$$q_{mk}^{(a)} = \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1}(\boldsymbol{R}_{\xi_a}\boldsymbol{\nu}_k + \boldsymbol{t}_{\xi_a}) + \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{R}\boldsymbol{\Omega}_k \boldsymbol{R}_{\xi_a}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk} \tag{5.9}$$

Using the compact representation of the gradient (5.7), the Hessian is expressed as

$$\frac{\partial^2 F}{\partial \xi_b \partial \xi_a} = \sum_{m=1}^{M} \sum_{k=1}^{K} f_{mk}\left[(d_{mk}^{(a)} + q_{mk}^{(a)})(d_{mk}^{(b)} + q_{mk}^{(b)}) + (D_{mk}^{(ba)} + Q_{mk}^{(ba)})\right] \tag{5.10}$$

where $D_{mk}^{(ba)} = \frac{\partial}{\partial \xi_b} d_{mk}^{(a)}$, $Q_{mk}^{(ba)} = \frac{\partial}{\partial \xi_b} q_{mk}^{(a)}$, and $d_{mk}^{(b)}$ and $q_{mk}^{(b)}$ are the same as $d_{mk}^{(a)}$ and $q_{mk}^{(a)}$, but refer to elements of ξ_b instead of ξ_a. The expressions are expanded as

$$D_{mk}^{(ba)} = -Tr\left\{\boldsymbol{\Omega}_k\left(\boldsymbol{R}_{\xi_b}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{R}_{\xi_a} - \boldsymbol{R}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_{mk}^{(b)} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{R}_{\xi_a} + \boldsymbol{R}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{R}_{\xi_b \xi_a}\right)\right\} \tag{5.11}$$

$$\begin{aligned} Q_{mk}^{(ba)} = -\Bigg(&\boldsymbol{j}_{mk}^{(b)^T} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{j}_{mk}^{(a)} - \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_{mk}^{(b)} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{j}_{mk}^{(a)} + \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{H}_{mk}^{(ba)} - \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_{mk}^{(a)} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{j}_{mk}^{(b)} \\ &+ \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_{mk}^{(b)} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_{mk}^{(a)} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk} - \frac{1}{2} \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_{mk}^{(ba)} \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk}\Bigg) \end{aligned} \tag{5.12}$$

where

$$\boldsymbol{j}_{mk}^{(a)} = -\boldsymbol{R}_{\xi_a} \boldsymbol{\nu}_k - \boldsymbol{t}_{\xi_a}$$

$$\boldsymbol{Z}_{mk}^{(a)} = \boldsymbol{R}_{\xi_a} \boldsymbol{\Omega}_k \boldsymbol{R}^T + \boldsymbol{R}\boldsymbol{\Omega}_k \boldsymbol{R}_{\xi_a}^T$$

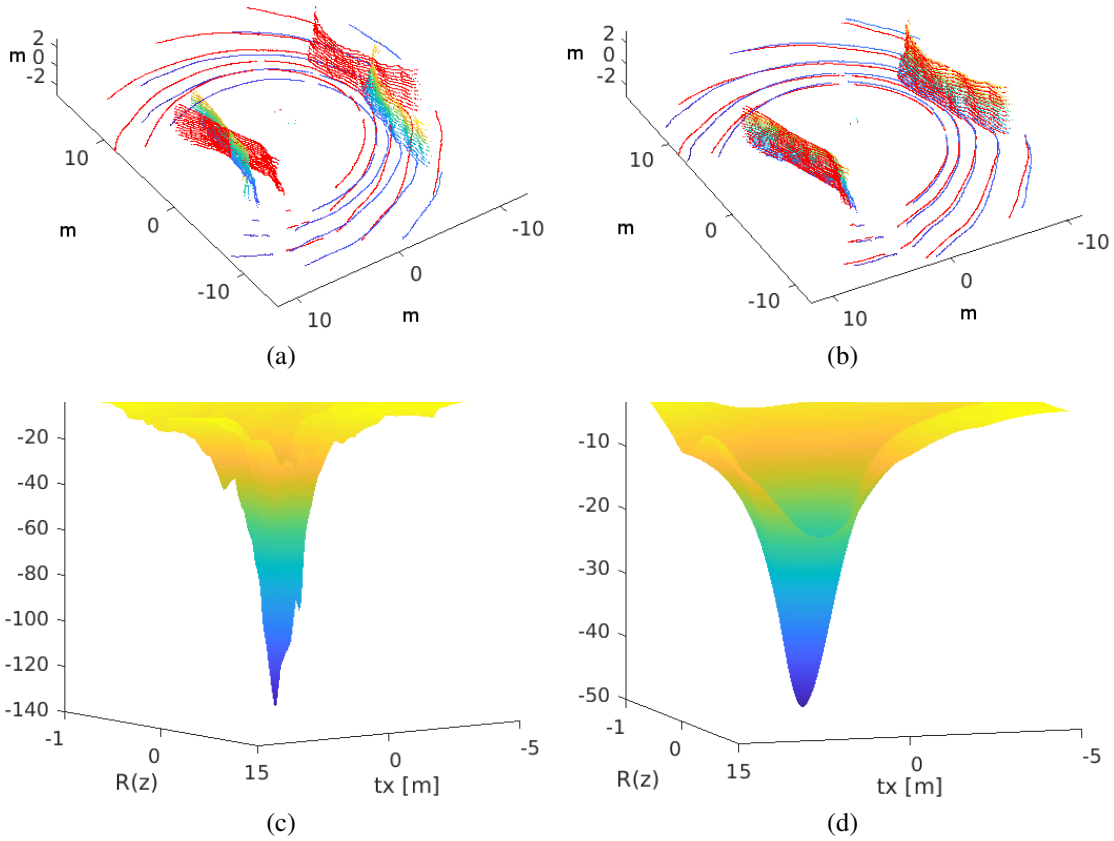


Figure 5.3: Natural environments typically have sparse features, rendering feature based matching difficult. Target (red) and source (blue-yellow) point clouds are initially misaligned in (a) and the goal is to align them as in (b). When the pointclouds are represented as GMMs with fully anisotropic covariances the resulting cost function has the steep and non-convex form of (c). Converting the anisotropic covariances into isotropic-planar covariances yields a cost function that is almost completely convex as shown in (d) and enables the accurate registration shown in (b).

$$\boldsymbol{H}_{mk}^{(ba)} = \frac{\partial \boldsymbol{j}_a^{mk}}{\partial \xi_b} = -\boldsymbol{R}_{\xi_b \xi_a} \boldsymbol{\nu}_k$$

$$\boldsymbol{Z}_{mk}^{(ba)} = \boldsymbol{R}_{\xi_b \xi_a} \boldsymbol{\Omega}_k \boldsymbol{R}^T + \boldsymbol{R}_{\xi_a} \boldsymbol{\Omega}_k \boldsymbol{R}_{\xi_b}^T + \boldsymbol{R}_{\xi_b} \boldsymbol{\Omega}_k \boldsymbol{R}_{\xi_a}^T + \boldsymbol{R} \boldsymbol{\Omega}_k \boldsymbol{R}_{\xi_b \xi_a}^T$$

Please see Appendix B for the detailed derivation of both the gradient and the Hessian.

5.1.3 Optimization Framework

As stated in Section 5.1.1, the cost function to minimize is highly non-convex, which may result in a poor quality registration unless the initialization is close to the actual solution. A technique

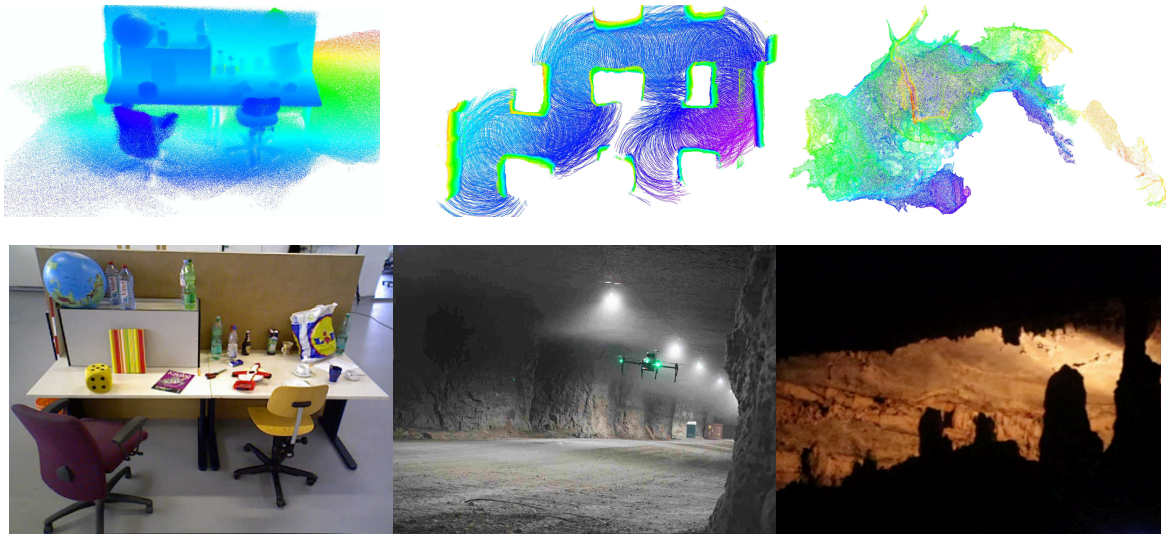
similar to that of GICP [74] is employed to smooth the cost function and prevent the optimizer from becoming trapped in local minima. Segal et al. [74] note that pointclouds taken from two different positions sample a two-dimensional manifold in three-dimensional space, so each measured point only provides a constraint along its surface normal. When the sampled points are represented as a probabilistic generative model, the covariance about the mean is modified to model the high certainty about the surface normal direction and low certainty along the local plane. The modified covariance is computed as the eigen decomposition $\mathbf{\Lambda}_m = \boldsymbol{U}_m \boldsymbol{D}_m \boldsymbol{U}_m^T$ and $\mathbf{\Lambda}_m$ is replaced with

$$\mathbf{\Lambda}_m = \boldsymbol{U}_m \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \epsilon \end{bmatrix} \boldsymbol{U}_m^T \tag{5.13}$$

where ϵ is a small constant representing the covariance along the normal. This has the effect of smoothing the cost function (see Fig. 5.3d for a representative example). However, this change in cost function is not guaranteed to preserve the minimum at the same location. Therefore, the optimization is run twice: once with the modified isotropic-planar (isoplanar) covariances specified in [74] to smooth the cost function followed by a second run with the original cost function seeded with the results from the first run. The second optimization quickly converges.

The cost function proposed by Stoyanov et al. [80] to remove the contribution of the determinant (see (5.6)) is compared to the proposed approach, called the Isoplanar Hybrid approach, in Section 5.2.1.

An unconstrained optimization problem is realized through the on-manifold parameterization of the objective function (5.5). The Riemannian trust-region method with conjugate gradients is used to optimize the objective, which is implemented in the MATLAB manifold optimization toolbox, manopt [13].



(a) TUM (b) Mine (c) Cave

Figure 5.4: (a) An RGB-D dataset of a cluttered office environment and laser datasets of a (b) mine (Image credit: Near Earth Autonomy) and (c) unstructured cave environment are employed to evaluate the proposed approach. The top row of images are from left-to-right a reconstruction of depth observations from the TUM RGB-D SLAM dataset and benchmark, laser observations of a mine environment, and Faro scans of the undeveloped cave. The bottom row consists of images of the environments.

5.2 Results

The selection of the Isoplanar Hybrid approach as the proposed approach is first motivated by comparing to variants of the GMM registration in Section 5.2.1 and then by comparing to the state-of-art registration methods, NDT D2D[1] [80], GICP[2] [74], and NICP[3] [75], in Section 5.2.2. The performance is evaluated with three datasets consisting of RGB-D and Velodyne VLP-16 laser observations illustrated in Fig. 5.4.

The GMM registration algorithms consider all pairs of components instead of just the closest pairs as is done in [80]. Preliminary tests that incorporate a d_{max} parameter analogous to GICP's to limit the number of component pairs considered during registration suggests a considerable increase in speed; however, the d_{max} parameter is highly dependent on the observed environment

[1]https://github.com/OrebroUniversity/perception_oru-release
[2]https://github.com/PointCloudLibrary/pcl
[3]https://github.com/yorsh87/nicp

and correct selection of this parameter is a challenge in its own right so the timing results are not reported.

The metrics used to evaluate the proposed method are the Root Mean Square Error (RMSE) as detailed in [81] and the odometric error between consecutive frames. The RMSE is defined as the relative pose error at time step i:

$$\mathbf{E}_i := \left(\mathbf{Q}_i^{-1}\mathbf{Q}_{i+1}\right)^{-1}\left(\mathbf{P}_i^{-1}\mathbf{P}_{i+1}\right) \tag{5.14}$$

$$RMSE(\mathbf{E}_{1:n}) := \left(\frac{1}{n-1}\sum_{i=1}^{n-1}\|trans(\mathbf{E}_i)\|^2\right)^{1/2} \tag{5.15}$$

where $trans(\mathbf{E}_i)$ refers to the translational components of the relative pose error $\mathbf{E}_i$, the estimated trajectory $\mathbf{P}_1, \dots \mathbf{P}_n \in SE(3)$ and the ground truth trajectory $\mathbf{Q}_1, \dots, \mathbf{Q}_n \in SE(3)$. The odometric error is computed as the translation and rotation error between frames 1 and j where $j \in [1, n]$:

$$\mathbf{E}_j := \left(\mathbf{Q}_1^{-1}\mathbf{Q}_j\right)^{-1}\left(\mathbf{P}_1^{-1}\mathbf{P}_j\right) \tag{5.16}$$

$$OE(\mathbf{E}_{1:n}) := \|trans(\mathbf{E}_j)\| \tag{5.17}$$

For both RMSE and odometric errors, the rotation errors are similarly computed.

5.2.1 Evaluation of GMM Registration Variants

Two evaluations are conducted that motivate the selection of the Isoplanar Hybrid approach as the proposed approach. Each evaluation highlights the strengths and weaknesses of the GMM registration variants. There are five variations considered: the *m-GMM Isoplanar* method leverages the isoplanar covariances of (8.5) with cost function (5.5); the proposed *m-GMM Isoplanar Hybrid* approach performs an optimization with cost function (5.5) and isoplanar covariances (8.5) followed by a refinement optimization using (5.5) and anisotropic covariances seeded with the results from the first optimization; the *m-GMM No Det.* variant performs a single optimization using the cost

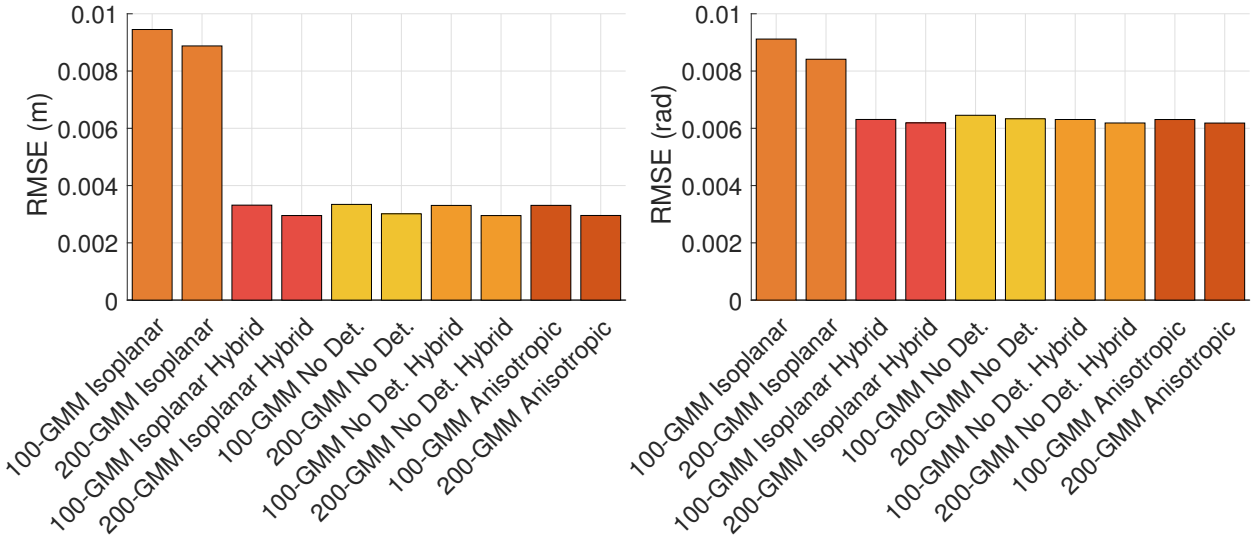
function from (5.6) and anisotropic covariances; the *m-GMM No Det. Hybrid* approach first performs an optimization using the cost function from (5.6) and anisotropic covariances followed by a second optimization with the cost function from (5.5) and anisotropic covariances seeded with the results from the first optimization; and the *m-GMM Anisotropic* approach performs a single optimization using the cost function from (5.5) and anisotropic covariances. In each case the m stands for the number of components in the GMMs.

TUM Dataset

The first evaluation is performed with a complete TUM RGB-D SLAM benchmark dataset consisting of over 2500 depth images shown in Fig. 5.4a. The RMSE results are shown in Fig. 5.5. The Isoplanar approach yields poorer registration results as compared to the other approaches because the cost function changes the shape of the covariances to bias certainty along the direction of the normal, which has the effect of making the optimization more robust to large pose differences between consecutive scans but introduces some error in the final registration results for this dataset where the difference in pose is small (approximately 1-2 cm and 1-2 degrees between consecutive scans). Note that the Isoplanar Hybrid approach overcomes this problem by running a second optimization with the anisotropic covariances.

Mine Dataset

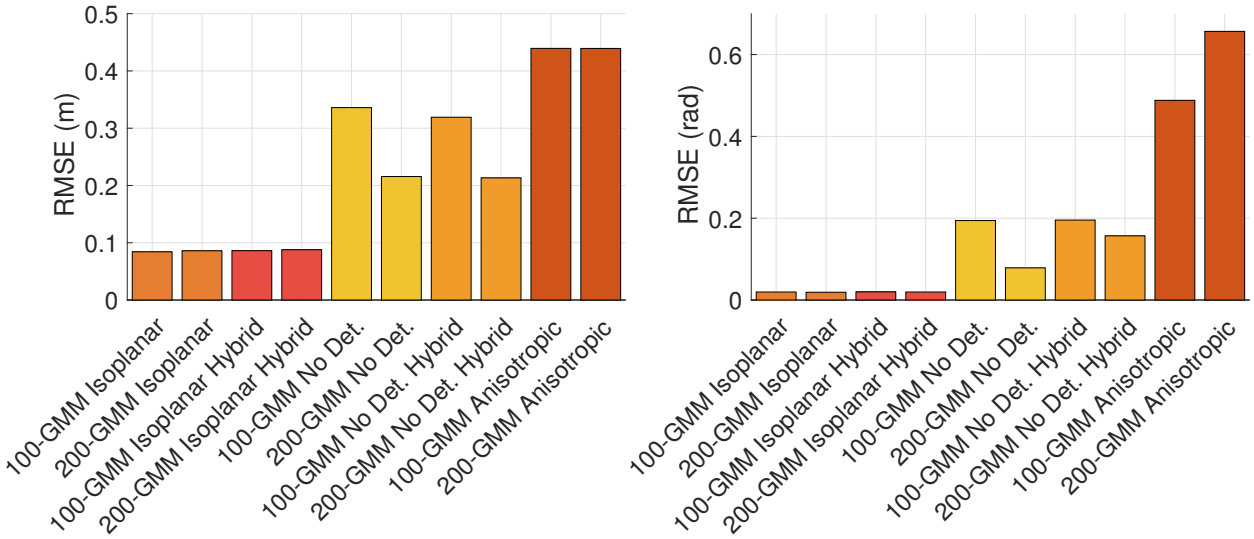
The second evaluation is performed with more than 300 laser observations from a mine environment shown in Fig. 5.4b. The RMS errors are shown in Fig. 5.6. The translation and rotation differences between successive observations is much larger in this dataset than the TUM dataset (20-60 cm and 2-10 degrees, respectively). In this case the Isoplanar approaches perform best because the effect of biasing the covariances to have higher certainty in the direction of the normal is to make the optimization more robust to larger translations and rotations between successive observations. The Anisotropic registration is unable to overcome local minima (representative ex-



(a) TUM Translation RMSE

(b) TUM Rotation RMSE

Figure 5.5: RMS errors for translation and rotation for each pair of consecutive observations in the TUM dataset consisting of more than 2500 depth images. The Isoplanar Hybrid approach overcomes the poor registration results of the Isoplanar variant by running a refinement optimization with anisotropic covariances.



(a) Mine Frame-to-frame Translation RMSE

(b) Mine Frame-to-frame Rotation RMSE

Figure 5.6: RMS errors for translation and rotation for each pair of consecutive observations in the Mine dataset consisting of more than 300 laser scans. The Isoplanar approaches have the best performance because biasing the covariances along the normal direction makes makes the optimization more robust to large translations and rotations.

ample shown in Fig. 5.3) which results in poor registration performance and the approaches that remove the determinant have a much narrower basin of attraction (representative example shown in Fig. 5.2) as compared to the Isoplanar approaches, which yields an optimization that is much more sensitive to initialization.

Because the Isoplanar Hybrid registration approach obtains accurate registration results in both evaluations, it is used to evaluate against state-of-art registration approaches.

5.2.2 Evaluation against State-of-the-art Methods

The Isoplanar Hybrid GMM approach is compared to NDT D2D, GICP, and NICP. The cell size of the NDT D2D registration and Isoplanar Hybrid GMM registration approaches is varied in the evaluations to determine the cell sizes and numbers of components that produce the best performance for each dataset. Varying the d_{max} parameter of GICP, which is the maximum matching threshold, does not significantly affect accuracy or timing performance so the parameter is conservatively set to be 100 for all tests. NICP provides optimized parameters for depth and laser sensors so the appropriate configuration file is used depending on the sensor used in the dataset.

Timing results are reported in Table 5.1. All of the tests are run with a single thread on an Intel i7-6700K CPU with 32GB of RAM. The evaluations assume that GMMs and NDT maps are pre-computed so only the time taken to register the distributions is counted. Hardware accelerated methods exist to create GMMs (e.g., [28, 77]) so the time to create these distributions is not counted. In addition, the NICP algorithm operates in image space, so to test the laser pointcloud datasets, the pointclouds are projected to a spherical 1000×1500 depth image per the parameters checked into the NICP codebase. One may note that the reported timing results are much larger than what is advertised in [75], and the reason for this is this work operates on the full-resolution 480×640 depth images to analyze the effectiveness of the registration algorithms, whereas the images in [75] are 120×160—a factor of 16× fewer points—which accounts for the difference in

Dataset	GICP (s)	NDT (s)	NICP (s)	GMM (s)
TUM	3.3565	0.0357	0.3703	0.2959
Mine	0.3238	0.0045	0.3754	0.3810
Cave	0.5536	0.0030	0.4904	0.3905

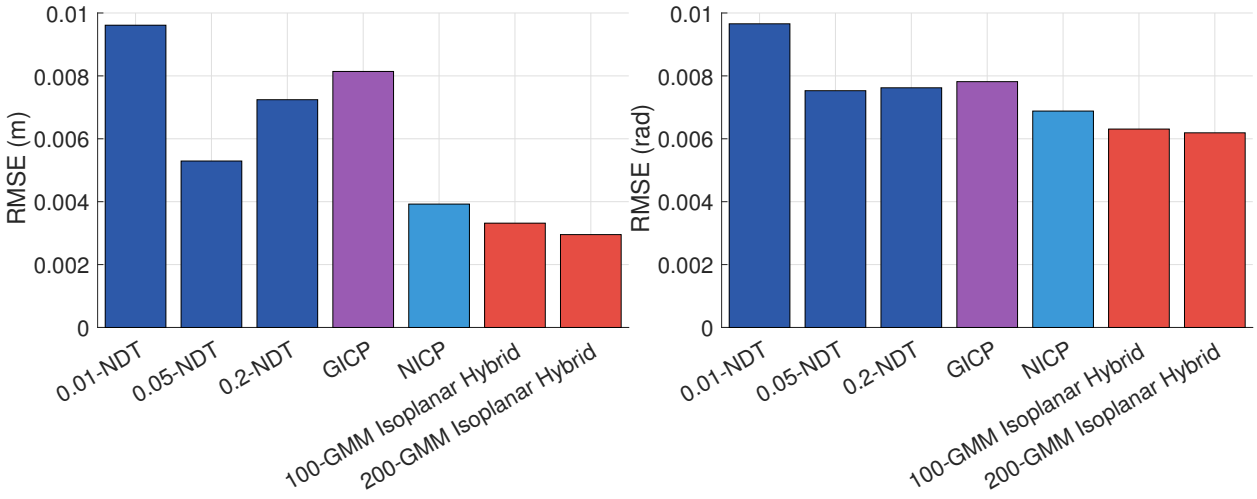
Table 5.1: Timing results for each method. The 100-GMM Isoplanar Hybrid approach remains within reasonable bounds of timing as compared to the state of the art.

timing results.

TUM Dataset

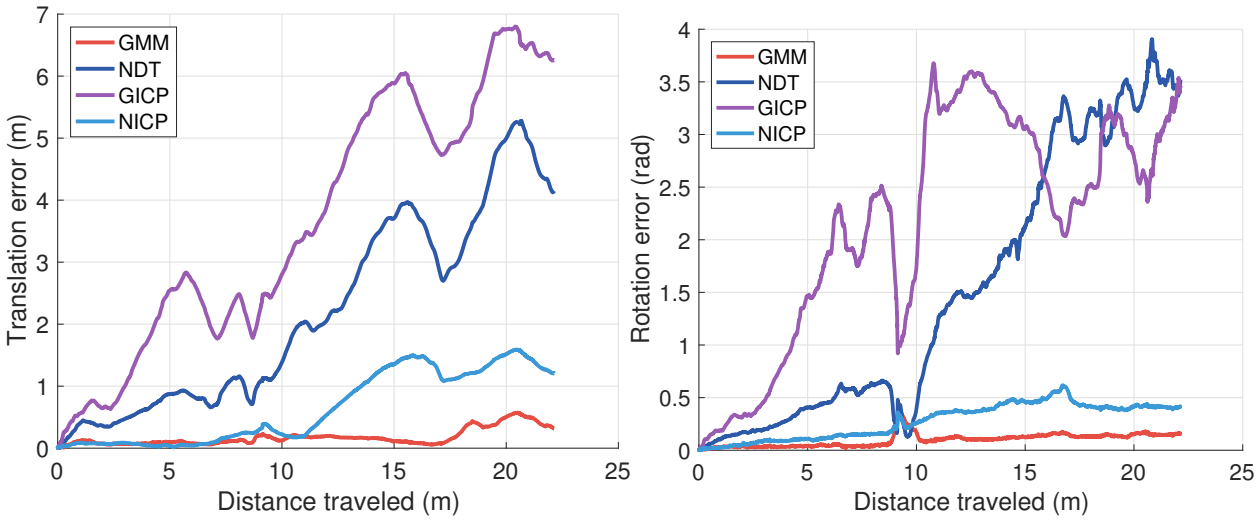
Each method is evaluated with the approximately 2500 sensor observations from the TUM dataset shown in Fig. 5.4a. The RMSE values shown in Fig. 5.7 are used to determine the optimal parameters for the NDT cell size and number of GMM components to use to compute the odometric error.

Figure 5.7 illustrates that for the TUM dataset, an NDT cell size of 0.05 m produces the best performance and 100 components in the Isoplanar Hybrid registration method is sufficient. The odometric error is shown in Fig. 5.8 and two views of the reconstructed paths are shown in Fig. 5.9. GICP produces the poorest performance for this dataset, which is similar to the behavior of the m-GMM Isoplanar variant from Fig. 5.5 that also uses the isoplanar covariances without a refinement optimization. The NDT approach fares slightly better in terms of translation performance which aligns with the behavior of the m-GMM No Det. registration variant from Fig. 5.5. NICP performs better than the NDT approach but has about 0.9 m more error in translation and 0.26 rad. error than the 100-GMM Isoplanar Hybrid approach by the end of the trajectory.



(a) TUM Translation RMSE (b) TUM Rotation RMSE

Figure 5.7: The RMS error for (a) translation and (b) rotation demonstrate the 100-GMM Isoplanar Hybrid approach is sufficient to outperform the other approaches for frame-to-frame RMSE.



(a) TUM Translation Error (b) TUM Rotation Error

Figure 5.8: The (a) translation and (b) rotation error statistics are plotted as a function of distance traveled. The proposed approach has lower odometric error compared to the state of the art.

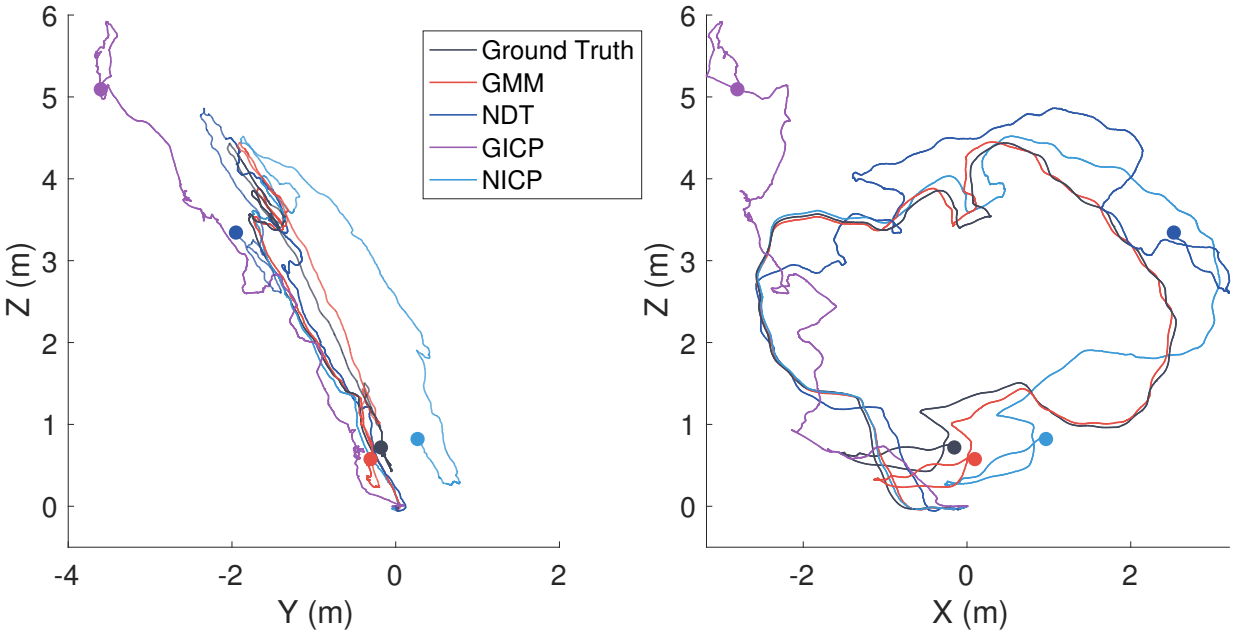
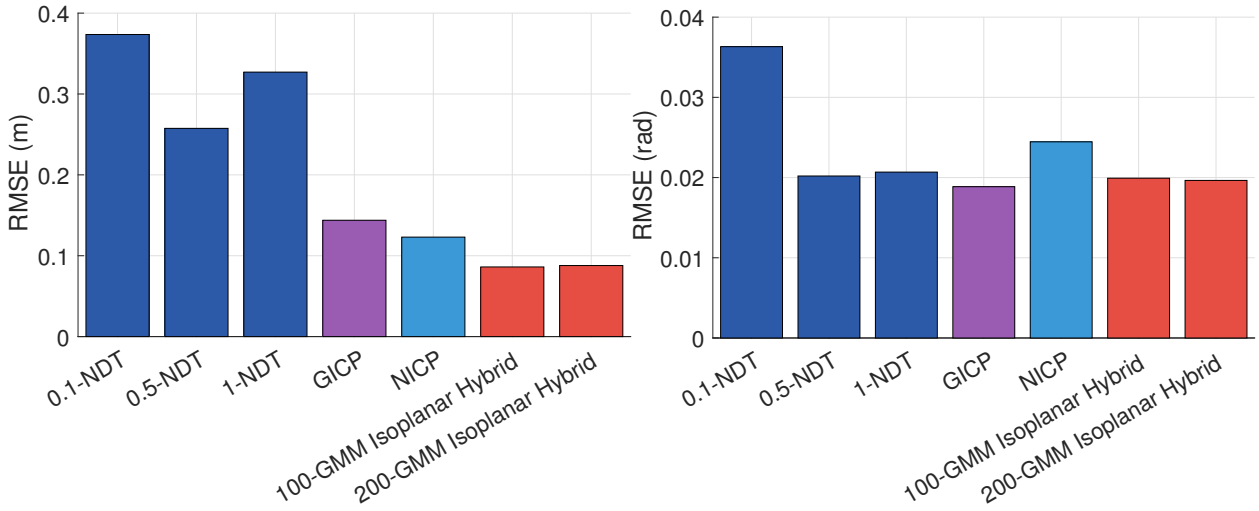


Figure 5.9: Two views of the reconstructed paths for the TUM dataset. The trajectories begin at (0,0,0) and end at the colored dots. Note the 100-GMM Isoplanar Hybrid approach closely matches the ground truth trajectory as compared to the other approaches.

Mine Dataset

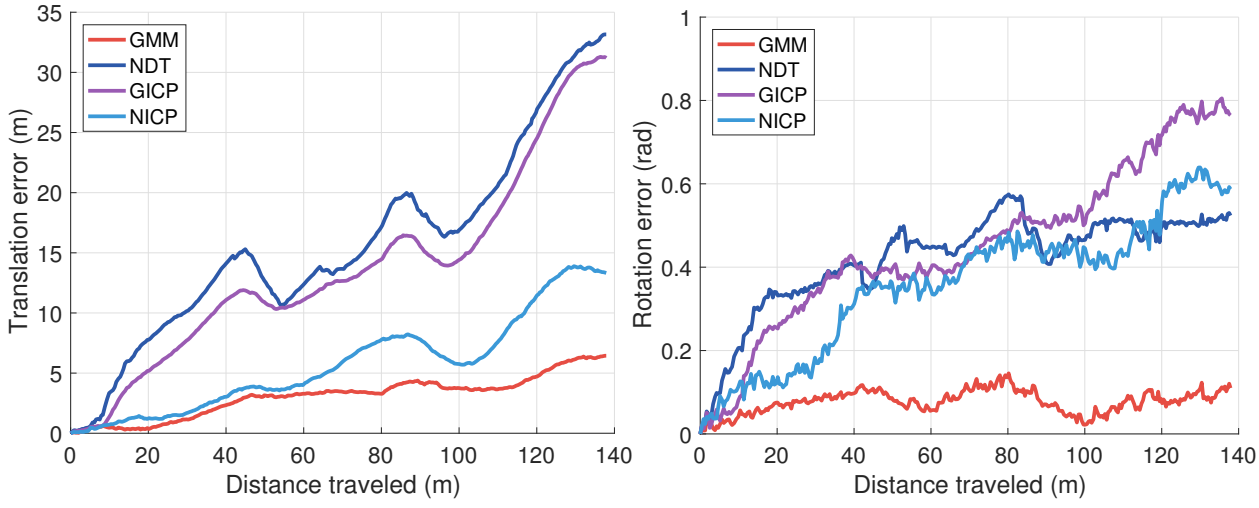
The next dataset is the Mine dataset consisting of more than 300 laser scans, and exhibits significantly larger rotations and translations as compared to the TUM dataset (20-60 cm and 2-10 degree differences between consecutive observations). The RMS errors for each method are shown in Fig. 5.10. The NDT with cell size equal to 0.5 m performs best and 100 components for the GMM registration are sufficient to outperform the state of the art in terms of translation RMSE. While NDT outperformed the GICP results for the TUM dataset, GICP outperforms NDT for the mine dataset as shown in Fig. 5.11. The estimated trajectories using each of the methods are shown in Fig. 5.12.



(a) Mine Translation RMSE

(b) Mine Rotation RMSE

Figure 5.10: The (a) translation and (b) rotation RMSE for the Mine dataset are shown. A cell size of 0.5m performs best for the NDT map. The GMM registration approach performs well with 100 components.



(a) Mine Translation Error

(b) Mine Rotation Error

Figure 5.11: The (a) translation and (b) rotation odometric error statistics for the Mine dataset demonstrate that the 100-GMM Isoplanar Hybrid approach significantly outperforms the other approaches.

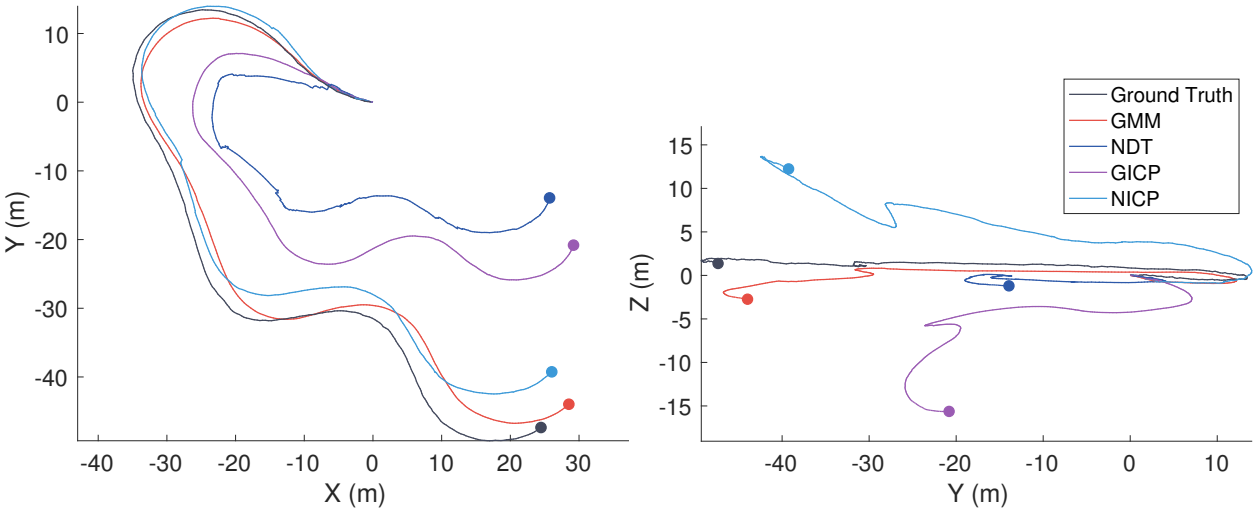


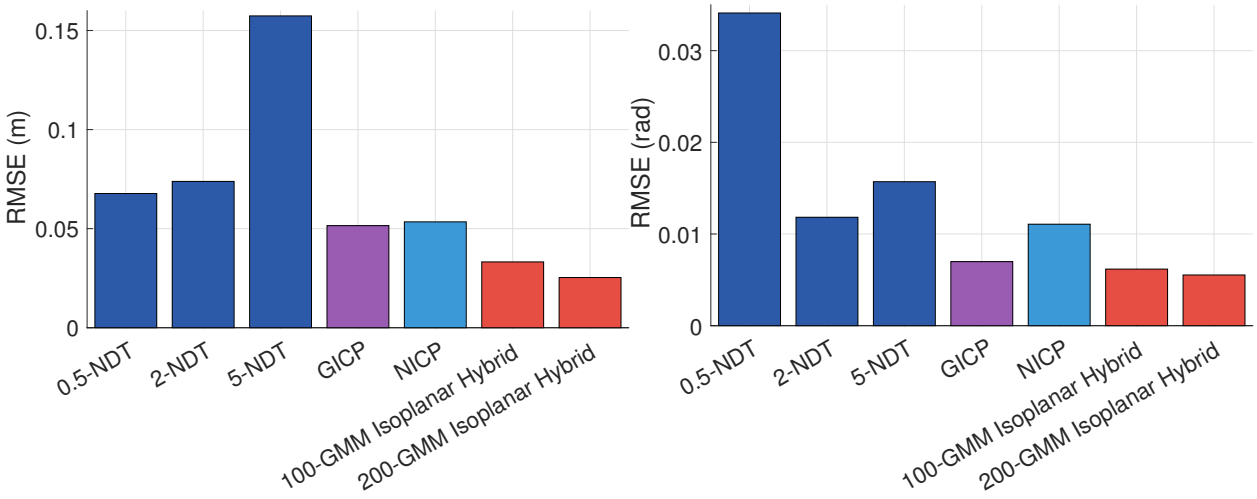
Figure 5.12: Two views of the estimated trajectories for the Mine dataset. The trajectories begin at (0,0,0) and end at the colored dots. Note the 100-GMM Isoplanar Hybrid approach closely matches the ground truth trajectory.

Cave Dataset

The last dataset is the Cave dataset consisting of more than 350 laser scans from Rapps Cave in Greenbrier County, WV. The terrain exhibits complex concavities and disjoint objects, which are visible in the bottom image of Fig. 5.4c. An aerial robot equipped with a VLP-16 laser scanner was used to obtain data in the cave. A FARO[4] laser scanner was used to collect ground truth.

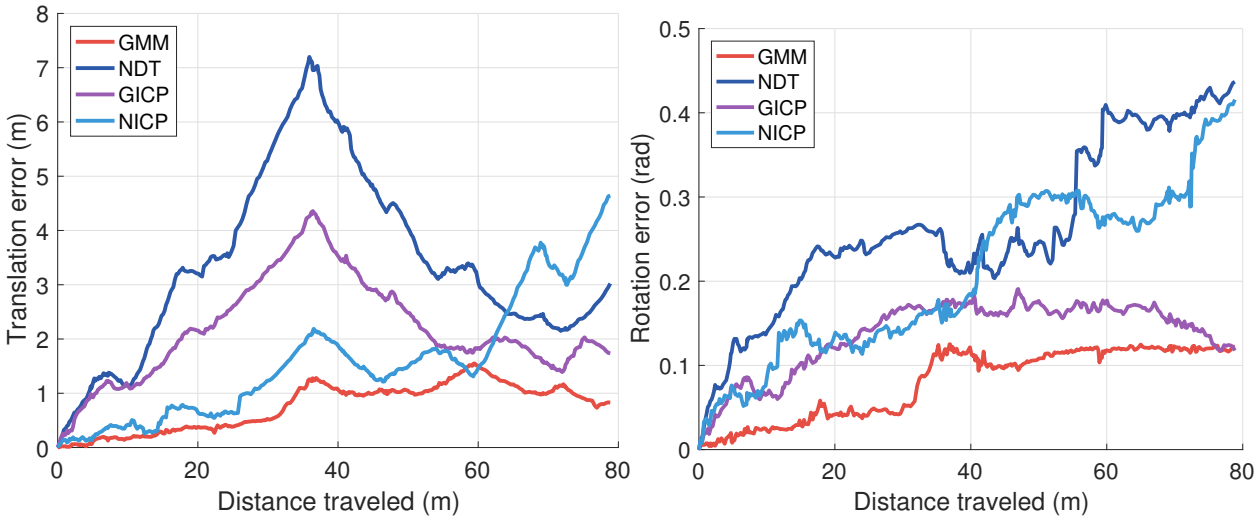
The translation and rotation RMS errors may be found in Fig. 5.13, which illustrates that the 100-Isoplanar Hybrid GMM approach is sufficient for this dataset and NDT D2D performs best with a cell size of 2 m. The odometric errors are shown in Fig. 5.14. While the NICP approach initially does well, the performance deteriorates towards the end of the trajectory. The 100-GMM Isoplanar Hybrid approach most closely follows the ground truth trajectory for this dataset. The trajectories of each of the approaches is shown in Fig. 5.15.

[4]https://www.faro.com



(a) Cave Translation RMSE (b) Cave Rotation RMSE

Figure 5.13: The (a) translation and (b) rotation RMSE for each approach with the Cave dataset.



(a) Cave Translation Error (b) Cave Rotation Error

Figure 5.14: The (a) translation and (b) rotation error statistics demonstrate that the 100-GMM Isoplanar Hybrid approach outperforms the other approaches.

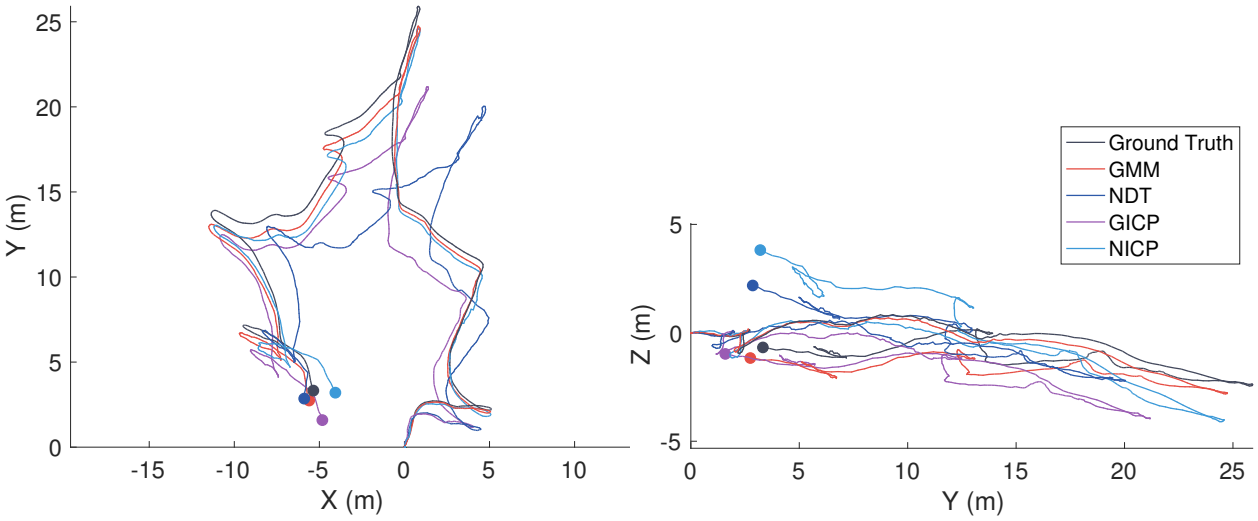


Figure 5.15: Two views of the estimated trajectories for the Cave dataset. The trajectories begin at (0,0,0) and end at the colored dots. Note the 100-GMM Isoplanar Hybrid approach outperforms the NICP, 2 m NDT Map and GICP approaches, particularly towards the end of the trajectory.

5.3 Conclusions

This chapter demonstrated a real-time viable method for registering GMMs in feature-deprived and dark environments such as mines and caves. The approach minimizes the squared L2 norm between the two distributions and considers all possible correspondences between mixture components. The compactness of the GMM representation is leveraged to decrease the runtime of the algorithm. Superior results compared to the state of the art are obtained in a degraded mine, unstructured cave, and cluttered office environments.

Chapter 6

Arbitrary Resolution Occupancy Modeling using GMMs

The previous chapter developed a method to determine the rotation and translation parameters that would transform one GMM into the frame of another. To enable information-theoretic planning, occupancy must be derived from these models. Occupancy modeling enables reasoning about known and unknown regions of the environment and forms a fundamental capability for active perception systems. Most occupancy mapping approaches model the environment as a discrete set of cells of fixed resolution that limits the environment representational fidelity. This chapter overcomes these limitations by proposing to learn approximate continuous representations for the evidence of occupied and free space from which the probability of occupancy may be derived at arbitrary resolution. The methods presented enable efficient and accurate derivation of occupancy within a local area and at multiple resolutions.

6.1 Approach

This section presents a method to model occupancy at arbitrary resolution from GMMs (see Section 3.1 for a mathematical description of the GMM) learned from pointcloud sensor observations.

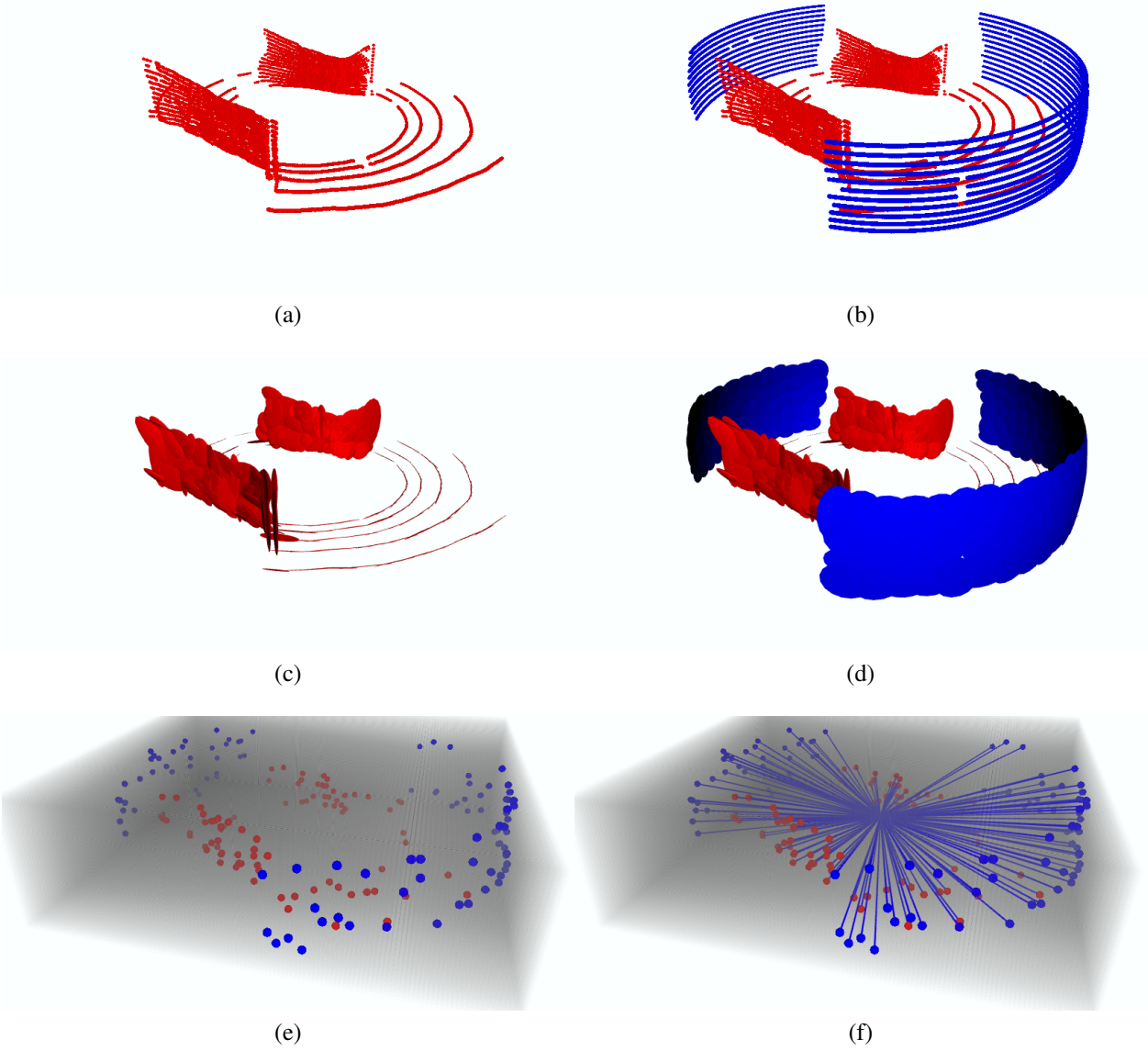


Figure 6.1: Overview of the proposed methodology via a representative example of pointcloud data from a mine environment. (a) Occupied points within a $15\,\mathrm{m}$ *range are shown in red and (b) points outside this range are projected to* $15\,\mathrm{m}$ *and shown in blue. (c) illustrates a 200-component GMM created from the occupied points and (d) illustrates a 200-component GMM created from the projected free space points.(e) 100 points are sampled from the occupied and free-space GMMs, where the number 100 is chosen for illustration purposes, only. In practice, a larger number of points would need to be used (see Fig. 6.2d for an example of the effect of varying the number of sampled points). These points are used to update the occupancy values in the occupancy grid shown in grey. (f) The points are projected to the sensor origin and all voxels along the beam are updated to incorporate the observed free space.*

Given points representing the occupied and free space in the environment (representative example shown in Fig. 6.1b), two GMMs may be separately learned from the poinclouds (shown in Fig. 6.1d). Occupancy may be recovered by Monte Carlo sampling from the distributions and raytracing through a grid map to the sensor origin (shown in Fig. 6.1f).

6.1.1 Occupied Space Modeling

3D LiDAR sensors estimate the distance to objects by illuminating the target with light and measuring the reflected pulses. Because the origin of the beam of light is known, 3D points may be recovered from distances. An example of the returns from a LiDAR sensor taken in a mine environment is shown in Fig. 6.1a. A GMM $\mathcal{G}(\boldsymbol{x})$ encodes the spatial density of points (shown in Fig. 6.1c) and implicitly encodes sensor noise within the surface model. These points sample a 2-manifold in 3D space so the resulting GMM will typically have near degenerate dimension directed along the normal vector to the surface. The variance of the data along this dimension is captured in the component covariance and describes the sensor noise. As each component corresponds to an individual measurement and each measurement is invariant to pose uncertainty, corrections may be applied on a component-wise basis via the following equations that transform the density through a rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ and translation vector $\boldsymbol{t} \in \mathbb{R}^{3\times1}$.

$$\boldsymbol{\mu}' = \boldsymbol{R}\boldsymbol{\mu} + \boldsymbol{t} \tag{6.1}$$

$$\boldsymbol{\Lambda}' = \boldsymbol{R}\boldsymbol{\Lambda}\boldsymbol{R}^T \tag{6.2}$$

6.1.2 Free Space Modeling

The GMM $\mathcal{G}(\boldsymbol{x})$ does not explicitly model the information contained in the beam between the sensor and endpoint. To overcome this limitation, a separate GMM can be learned over the free space points by projecting observations that exceed the sensor's max range to either the sensor's

max range or some other pre-determined max range. Figure 6.1b illustrates the points that exceed a $15\,\mathrm{m}$ range projected to $15\,\mathrm{m}$ and shown in blue. Figure 6.1d illustrates the corresponding GMM for free space $\mathcal{F}(\boldsymbol{x})$.

6.1.3 Occupancy Modeling

It is possible to determine the number of observed points in any volume by solving the integral

$$N_V^O = N_s \sum_{m=1}^{M} \pi_m \int_V \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m) \tag{6.3}$$

where N_s is the support size, or number of points the GMM is trained from. However, this integral cannot be computed in closed form, so Eq. (6.3) is evaluated for $\mathcal{G}(\boldsymbol{x})$ using Monte Carlo sampling. The free space evidence N_V^H is then determined by performing ray casting using the resampled points as described in Section 6.1.2. The probability of occupancy is modeled as a Bernoulli distribution over every volume V in space. The maximum *a posteriori* estimate of the distribution parameters is given by

$$\theta^{MAP} = \frac{N_V^H + N_p}{N_V^H + N_V^M + 2N_p} \tag{6.4}$$

where N_p is the contribution due to the prior and is user-defined to control the sensitivity of the distribution to new observations. The prior model forces all unobserved points to have a probability of occupancy of 0.5.

Algorithm 6.1 provides pseudocode for deriving occupancy from resampled points in an occupied GMM. Line 1 samples N points from the GMM $\mathcal{G}(\boldsymbol{x})$. The list of cells between the sensor origin $\boldsymbol{s}$ and the endpoint p_n are obtained as C_M, where $m \in [1, \ldots, M]$ in line 3. If $m < M$, the number of miss points, N_V^M, is incremented as shown on line 6. c_M is the cell corresponding to the occupied point p_N, so the contents of the cell are updated by incrementing N_V^H shown on line 8. Removing lines 5 and 7-9 yields the algorithm for updating the cells via a max-range GMM. In

this work, the getRayCells function is implemented via the approach proposed by Amanatides and Woo [2].

Algorithm 6.1 Pseudocode to update the probability of occupancy of occupied GMM $\mathcal{G}(\boldsymbol{x})$ given sensor location $\boldsymbol{s} \in$ SE(3) and number of points to sample N

```
P_N = samplePoints(G(x), N)
for p_n ∈ P_N do
    C_M = getRayCells(s, p_n)
    for c_m ∈ C_M do
        if m != M then
            updateMiss(c_m)          ▷ increment N_V^M for cell c_m
        else
            updateHit(c_m)           ▷ increment N_V^H for cell c_m
```

It is important to note that the voxelization presented here is not the same as standard voxel grid mapping, since the voxel grid is used as a convenient interpretation of the underlying, continuous GMM. The occupancy in an arbitrary discretization space could be computed; however, this would likely preclude the use of the proposed efficient Monte Carlo ray tracing strategy and would instead require computation of the intersection between the discretized shapes and the sampled beams.

6.2 Results

6.2.1 3D Occupancy Modeling

The GMM implementation used to evaluate this work is derived from the scikit-learn[1] toolkit and ported to C++ to decrease the runtime. The runtimes are collected on a late 2013 15" Macbook Pro with 2.60GHz Intel Core i7-4960HQ and 16GB RAM. All approaches are tested single-threaded.

The GMM occupancy model is evaluated with two datasets: a mine dataset (reconstruction shown in Fig. 6.2c) and the Freiburg campus dataset (raw pointcloud shown in Fig. 6.3a). For each pose and pointcloud, a free and occupied GMM are computed to represent the environment. 10^6 points are sampled from the distribution and raytraced to the sensor origin and the proba-

[1] http://scikit-learn.org/stable/modules/mixture.html

bility of occupancy is updated for each 25 cm voxel along the ray. To determine the accuracy of reconstruction, the Area under ROC (AUROC) is computed while varying the percentage of observations removed. The proposed approach is compared against the standard occupancy grid map and Bayesian Generalized Kernel Inference for occupancy map prediction [24] (BGK-OM) approaches in Fig. 6.2a, and demonstrates superior results in terms of inferring probability as the number of training points decreases. Timing results shown in Fig. 6.2b demonstrate the superior performance of the proposed approach. While the times reported in [24] are significantly lower, than what is reported here, the discrepancy can be accounted for by the following: 1. BGK-OM is highly parallelized with 8 threads so for a fair comparison the multi-threading is disabled; 2. the max range of the sensor in these tests is higher (15 m) than what is reported in [24] (4 and 8 m); 3. the environment is larger (100m$\times$100m$\times$8m) as compared to the largest environment reported in [24] (43.8m$\times$18.2m$\times$3.3m); 4. the maximum number of points used per scan reported in [24] is 7601 but the tests in Fig. 6.2b contain between 15,000-20,000 points when 0% of the observations are removed and after downsampling with the voxel grid filter. The dense voxel filter resolution is left unchanged from the default parameters (0.1 m) but when the sensor observations are sparse and the environment is large, few points can be removed. The priors are set to 0.5, block size is set to 1, and the resolution is 0.25 in keeping with the other approaches. The other parameters are unchanged.

The key advantages of the GMM occupancy model over state-of-the-art occupancy mapping approaches like the occupancy grid map, Octomap, or BGK-OM is the ability to correct for small or large changes in pose without the need to regenerate the entire map via Eqs. (6.1) and (6.2). BGK-OM requires significantly more data storage to obtain the same level of accuracy as can be seen in the results from the Freiburg Campus Dataset in Table 6.1. Because the standard occupancy grid map under performs as compared to both the proposed and BGK-OM approaches in Fig. 6.2, the memory requirements would be larger to achieve the same AUC. Furthermore, the timing needs

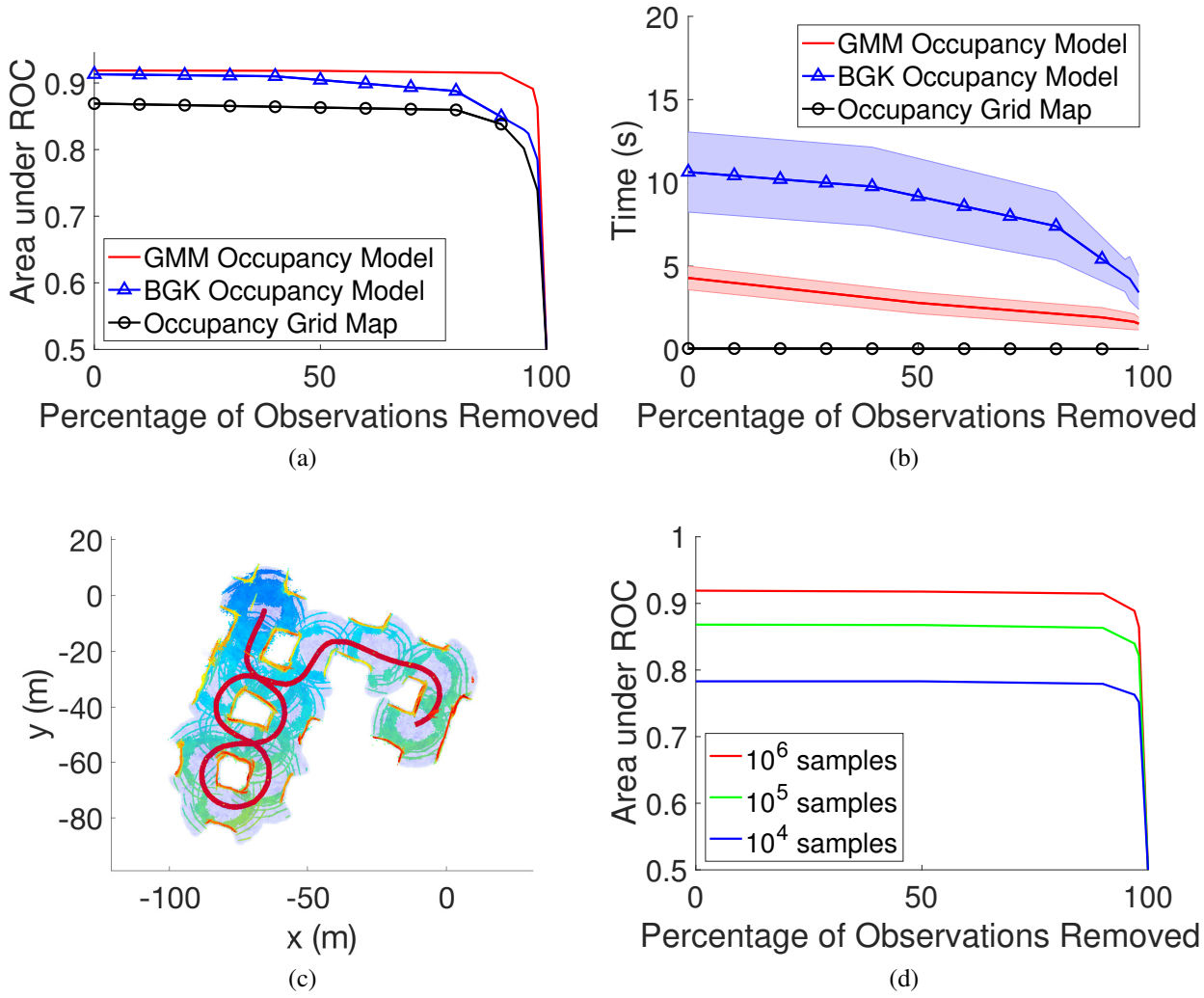
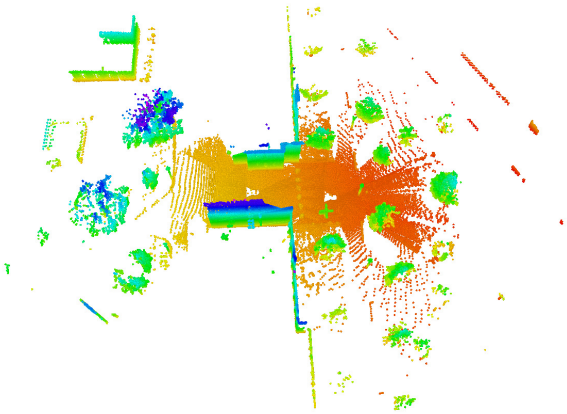
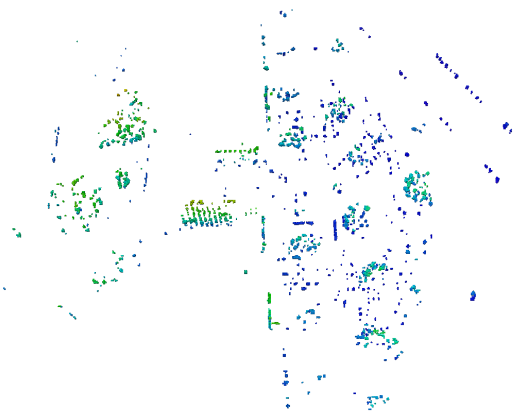


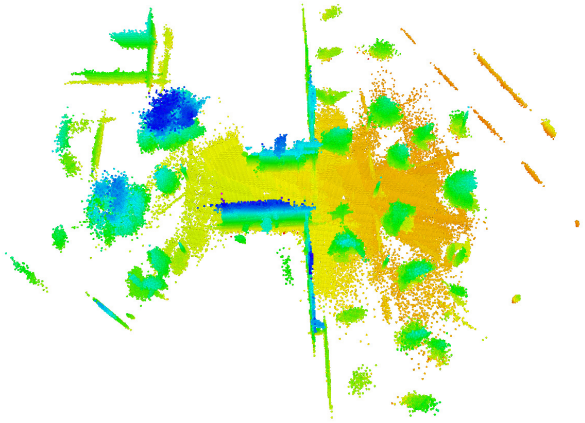
Figure 6.2: (a) The effect of data sparsity on the fidelity of the map is measured by computing the area under curve for several models constructed by increasingly reducing the percentage of observations removed from the original data. (b) provides timing analysis for each method along with variance in the lighter shaded region. (c) illustrates the reconstruction of the map after sampling from the GMM. 100-component GMMs were used to generate these results. (d) illustrates the effect of varying the number of sampled points for the GMM Occupancy Modeling approach. As the number of samples increases, the accuracy of the resulting occupancy model increases.



(a)

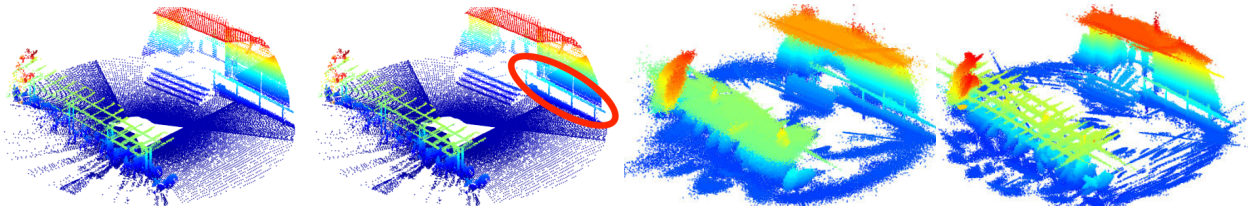


(b)



(c)

Figure 6.3: (a) Raw pointcloud from Freiburg campus dataset with coloring according to location along the z-axis (total is 447,528 points and approximately 5.12MB of data assuming 32-bit floats). (b) Reconstruction of occupied regions with BGK inference for occupancy maps using 9472 points (111 KB data storage requirement). The voxels displayed are those with a probability of occupancy greater than or equal to 0.65. The AUC is 0.72. (c) Reconstruction of GMM map using 1000 components (40 KB data storage requirement). The AUC is 0.82.



(a) (b) (c) (d)

Figure 6.4: (a) A pointcloud of a courtyard from the Freiburg campus dataset with (b) a railing outlined in red. (c) The occupied points from the grid map derived by Monte Carlo sampling a 200-component GMM and (d) 850-component GMM are shown after applying a threshold using variance = 0.06. The AUC for (c) is 0.79 and the AUC for (d) is 0.82. In both cases the railing is properly reconstructed.

	AUC	Memory Usage (Bytes)	
GMM			Number of Components
	0.8179	40,000	1000
	0.8072	20,000	500
	0.8055	10,000	250
	0.7849	4,000	100
BGK			Number of Points
	0.7603	204,132	17,011
	0.7182	113,664	9,472
	0.6338	60,120	5,010
	0.5845	37,140	3,095
	0.5428	18,660	1,555

Table 6.1: AUC for given memory usage of GMM occupancy model compared to the performance of BGK inference occupancy maps. The GMM is better able to reconstruct the occupancy information in the environment with 50× less data than the Bayesian Generalized Kernel inference approach for occupancy mapping.

for regenerating the entire map are prohibitive for real-time applications. Individual voxels cannot be meaningfully rotated and translated; however, individual components of a GMM are easily translated and rotated to correct for errors in pose [72].

The GMM infers the presence of occupied space in the neighborhood of observed points. However, the inferred occupied space may be unobservable from the sensor location due to occlusions from other surfaces. As a result, the inferred density may contribute erroneous free space evidence. For example, Fig. 6.4a depicts a laser scan from the Freiburg campus dataset that exhibits clutter and challenging surfaces to model in the form of a railing outlined in Fig. 6.4b. The inferred, but unobserved, portion of the wall contributes free space evidence to the voxels containing the railing, resulting in the probability of occupancy dropping for the railing voxels. To account for such situations, the variance of the voxels is leveraged to determine the occupancy classification levels. The variance of a voxel is the variance of a Bernoulli random variable

$$\sigma^2 = p(x_i)(1 - p(x_i)) \tag{6.5}$$

The variance quantifies the uncertainty in the occupancy value of the cell and introduces a deadband region in which voxels remain unclassified. The occupied and free thresholds are given by $\frac{1}{2}(1 \pm \sqrt{1 - 4\sigma^2})$. Appropriate selection of $\sigma^2 \in [0, 0.25]$ enables a threshold to be applied

depending on desired conservativeness for a given application. Fig. 6.4c illustrates the occupied cells when a variance thresholding is applied when reconstructing a GMM with 200 components. The railing is well represented in the reconstruction and the AUC is 0.79. Fig. 6.4d illustrates the same result with 850 GMM components and an AUC of 0.82.

6.2.2 Component Selection

Selecting the right number of components to model a GMM is an open area of research. A hierarchical, top-down strategy is developed by Eckart et al. [28] that produces a GMM by successively partitioning the data into J leaf nodes such that each leaf is a Gaussian density and $J \ll N$, where N is the number of points in the pointcloud. The downside of this approach is that user-defined convergence criterion must be set to determine when the point cloud has been sufficiently segmented. Srivastava and Michael [78] propose a bottom-up strategy that successively merges components until a knee-point is achieved. In addition, information-theoretic criterion such as the Bayesian Information Criterion (BIC) and Akaike Information Criterion (AIC) have also been used to determine the elbow-point where adding an additional component does not significantly add information; however, the elbow-point cannot always be unambiguously identified. The plots of BIC and AIC for a pointcloud shown in Fig. 6.5e are shown in Fig. 6.5a and Fig. 6.5b, respectively, for numbers of components between 1 and 200.

This work proposes another method to determine the number of components to ensure maximum resolution inspired by the work of Isler et al. [41] that measures surface coverage by discretizing the space into a voxel grid with very small cell size and counting the number of rays that collide with a surface in the voxel. For a given desired maximum resolution, the number of components needed to achieve that resolution may be determined by computing a GMM, resampling from the distribution a number of points equal to the support size, and raytracing through an occupancy grid map (note: not a Bernoulli grid). The AUC is computed by comparing the resulting

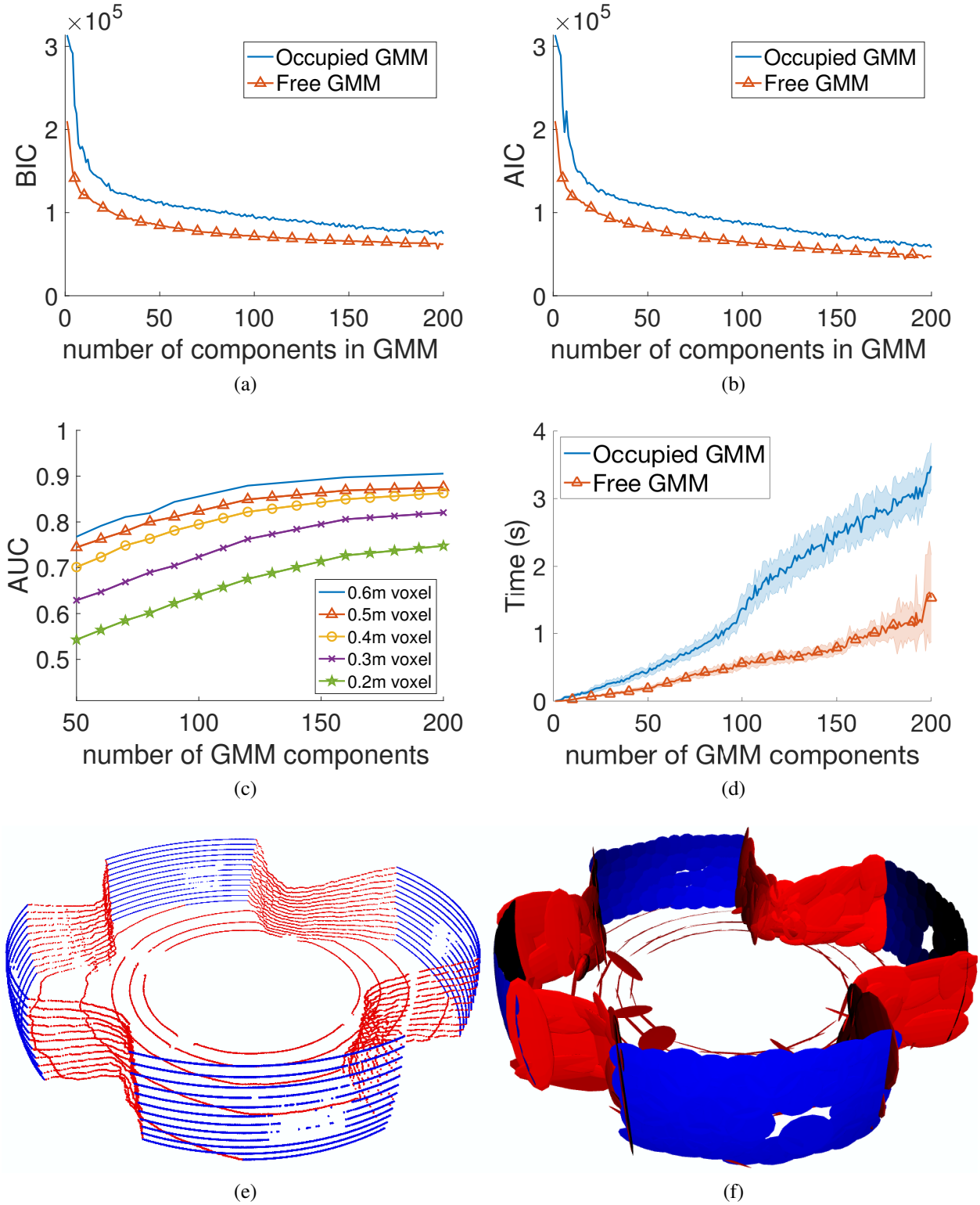


Figure 6.5: (a) The BIC and (b) AIC are often used to determine the number of components to represent a distribution. The proposed approach (c) ensures max resolution through comparison to an occupancy grid map. (d) Timing results for the occupied and free space GMMs as the number of components increases. A visualization of the pointcloud may be seen in (e) and the 200-component GMMs in (f).

occupancy grid map with a ground truth occupancy grid map whose values are determined by ray-tracing the original point cloud. Figure 6.5c illustrates the AUC scores when varying the number of components in the GMMs for free and occupied points.

The intuition behind this approach is that as the number of components in the model approaches the number of points in the sensor observation, the resulting AUC score will approach 1. It will be 1 only when the number of components exactly matches the number of points in the sensor observation. This representation also gives a more intuitive understanding of how selecting the number of components will affect the reconstruction. For example, when the desired cell size is $0.6\,\mathrm{m}$, 50 components may be sufficient, but when increasing the resolution to 0.3m cell size, 200 components would be needed to obtain similar AUC accuracy. To obtain good performance a trade-off must occur between the time it takes to calculate the GMM and the desired maximum resolution. Figure 6.5d illustrates the time to compute a GMM as the number of components increases.The times may be significantly reduced by employing the hierarchical strategy of [28]. This is left as future work discussed in the next section.

6.3 Conclusions

This chapter presented a continuous occupancy modeling methodology to enable efficient storage of a high-fidelity model of observed occupied and free space while remaining amenable to local or global updates in pose. The Monte Carlo sampling and ray-tracing approach is quantitatively evaluated against state-of-the-art occupancy modeling approaches which demonstrates superior modeling accuracy while substantially reducing the complexity of the representation.

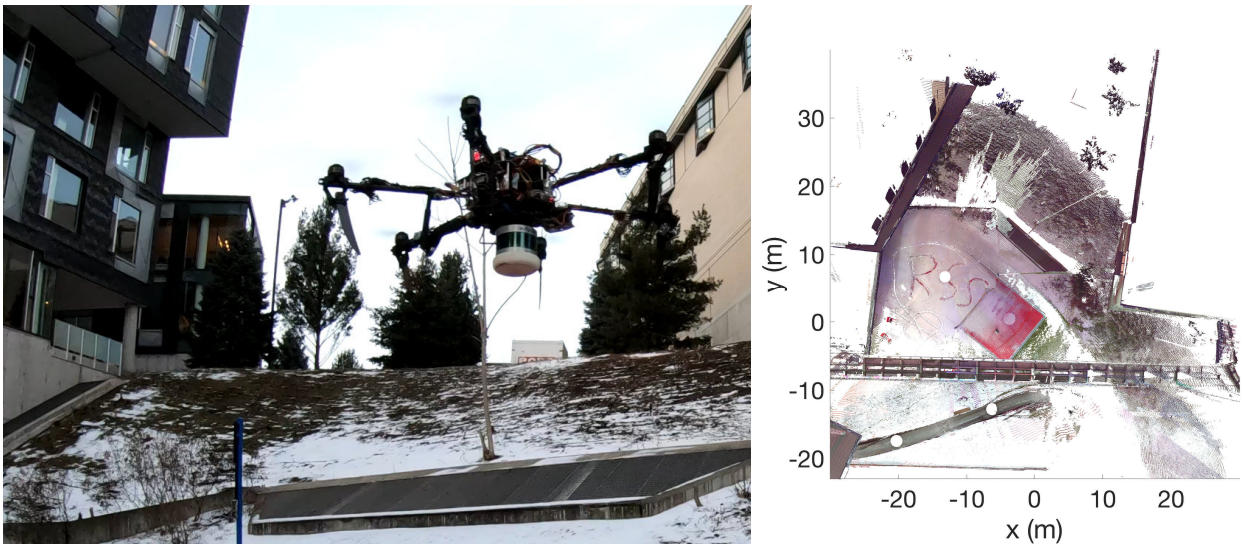
Chapter 7

Local Occupancy Mapping using GMMs for Exploration

To enable information-theoretic exploration, occupancy must be derived from GMMs in real-time. This chapter leverages and extends the occupancy modeling techniques of the previous chapter to generate GMMs and update an occupancy grid map in real-time to enable information-theoretic exploration on constrained aerial systems. The method is evaluated in simulation, deployed to an aerial system, and tested in a complex, real-world environment.

7.1 Approach

The exploration system consists of mapping, information-theoretic planning, and a monocular visual-inertial navigation system (Fig. 7.2). Section 7.1.1 develops a local grid mapping strategy that leverages GMMs and Section 7.1.2 details the planning approach that generates continuous trajectories that maximize the mutual information between potential sensor observations and the map. The visual-inertial navigation and control subsystems are detailed in Section 7.2.2 and results are presented in Section 7.2.



(a) (b)

Figure 7.1: (a) An autonomous aerial robot equipped with a 3D LiDAR explores an outdoor environment. (b) The exploration environment (top-down view) is a red volleyball court covered in snow.

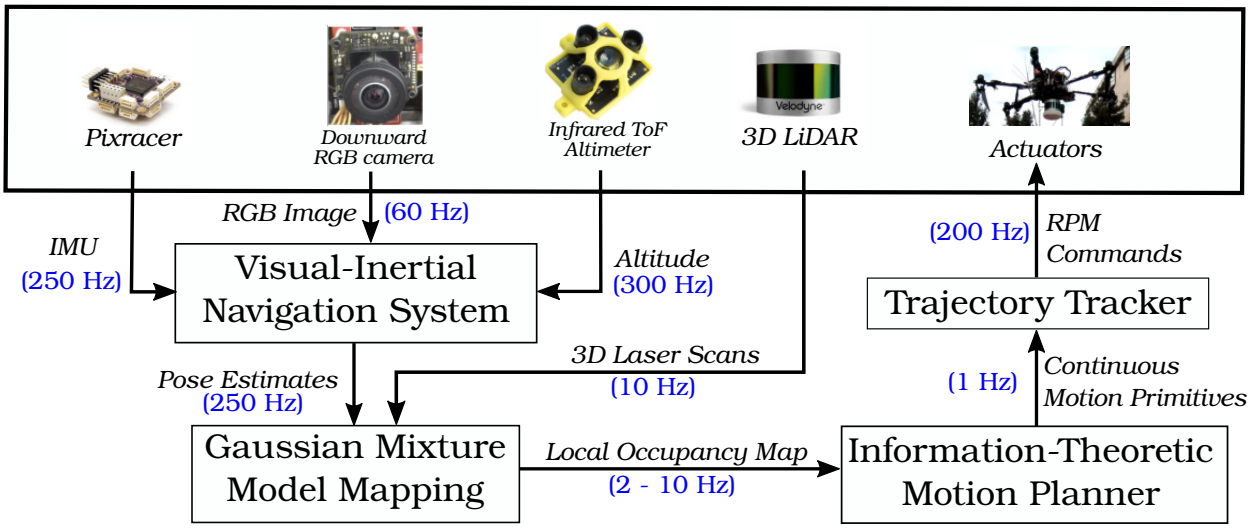


Figure 7.2: Overview of the autonomous exploration system presented in this work. Using pose estimates from a visual-inertial navigation system (Section 7.2.2) and 3D laser scans, the proposed mapping method (Section 7.1.1) builds a memory-efficient approximate continuous belief representation of the environment and generates local occupancy grid maps in real-time. A motion primitives-based information-theoretic planner (Section 7.1.2) uses the local occupancy map to generate snap-continuous forward-arc motion primitive trajectories that maximize the information gain over time.

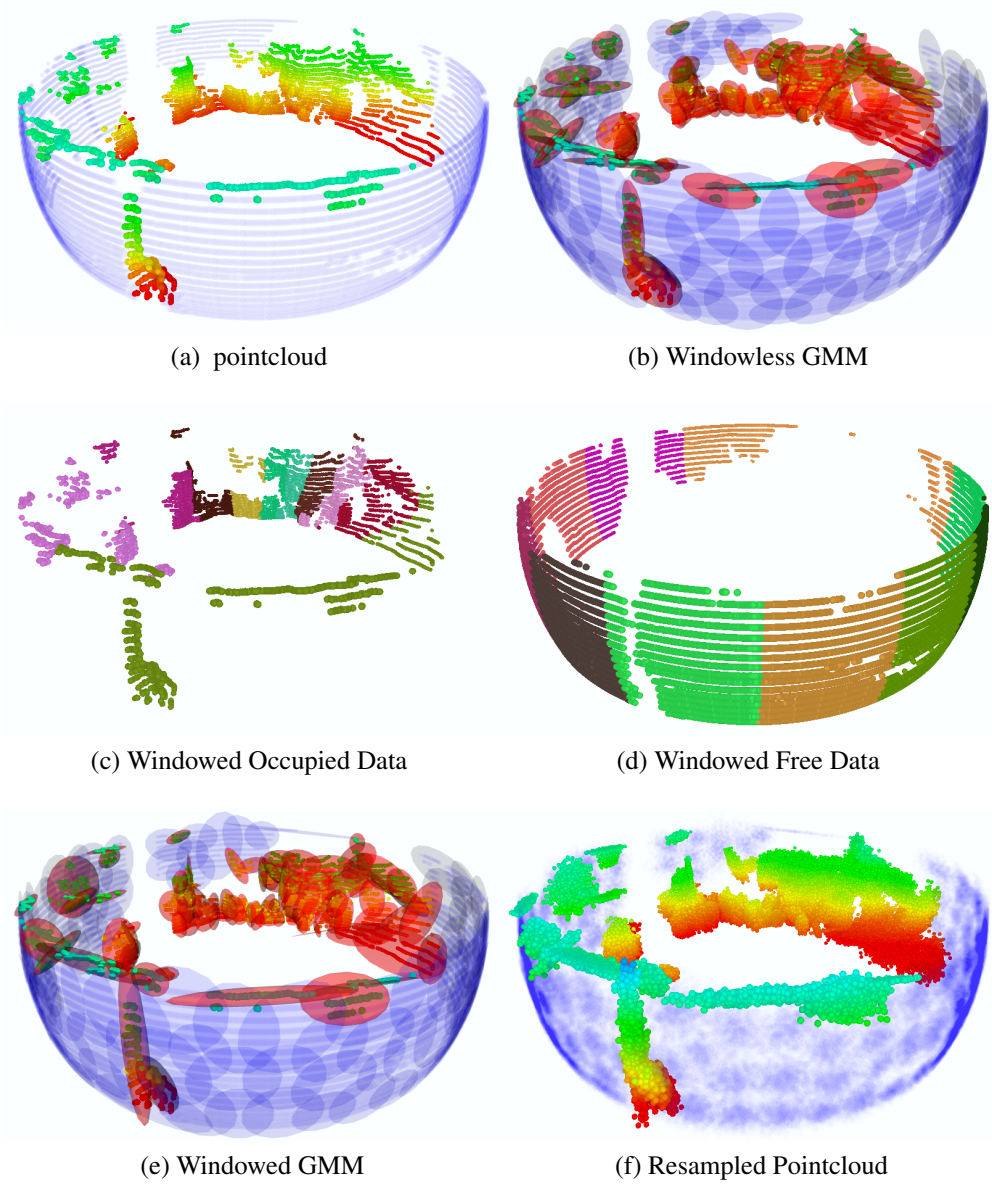
7.1.1 Mapping

Gaussian Mixture Models for Perception

The proposed approach leverages GMMs to compactly encode sensor observations for transmission over low-bandwidth communications channels. The GMM provides a generative model of the sensor observations from which occupancy may be reconstructed by resampling from the distribution and raytracing through a local occupancy grid map. Please see Section 3.1 for a mathematical description of the GMM.

The performance analysis of Table 3.1 highlights the computational challenges associated with learning a Gaussian Mixture Model from pointcloud observations for real-time active perception. As noted in Section 3.1.4, the Expectation step is computationally expensive because the responsibility matrix is of size $\mathbb{R}^{N\times M}$, where N denotes the number of points in the data set and M denotes the number of desired mixture components. The Maximization step is expensive because every point in the data set is used to update the parameters. The time required to initialize clusters via the KMeans++ [4] algorithm also scales with the size of the dataset (see Table 3.1).

To enable real-time performance, a hard partitioning of the data into W windows (see Figs. 7.3c and 7.3d) is performed. The windowing reduces computational cost by reducing the size of the responsibility matrix, where the entries of the responsibility matrix are denoted as γ_{nm} in Eq. (3.1). The effect of this windowing strategy is discussed in greater detail in Section 7.2 and, in particular, Fig. 7.7. Let $\mathcal{G}_i(\boldsymbol{x})$ be a GMM trained from N_i points in window i and let $\mathcal{G}_j(\boldsymbol{x})$ be a GMM trained from N_j points in window j, where $\sum_{w=1}^{W} N_w = N$ and N is the total number of points in sensor observation $\mathcal{Z}_t$ taken at time t. $\mathcal{G}_j(\boldsymbol{x}) = \sum_{k=1}^{K} \tau_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\nu}_k, \boldsymbol{\Omega}_k)$ may be merged into $\mathcal{G}_i(\boldsymbol{x}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)$ by concatenating the means, covariances, and weights. However, care must be taken when merging the weights as they must be renormalized to sum to 1 [77]. The



(a) pointcloud (b) Windowless GMM

(c) Windowed Occupied Data (d) Windowed Free Data

(e) Windowed GMM (f) Resampled Pointcloud

Figure 7.3: Overview of the windowing technique for learning GMMs from a sensor observation. (a) 3D LiDAR scan from Rapps Cave in Greenbrier, WV. Points at a distance larger than $5\,\mathrm{m}$ *are normalized to a unit vector and projected to* $5\,\mathrm{m}$ *(dark blue) to represent free space. Points with a norm less than* $5\,\mathrm{m}$ *represent occupied space. (b) A 100-component GMM* $\mathcal{F}(\boldsymbol{x})$ *is learned over the free space points (dark blue) and 100-component GMM* $\mathcal{G}(\boldsymbol{x})$ *is learned over the occupied points (red). (c) and (d) illustrate the windowing technique on the occupied and free space points, respectively, by coloring each window in a different color. The GMM learned with windowed pointcloud data is shown in (e). The windowless GMM produces a tighter fit to the data than the windowed GMM which is exemplified in the occupied point component closest to the viewer. The resampled model produced by sampling* 1×10^6 *points from (e) is shown in (f). The number of points to resample is selected for illustration purposes and to highlight that the resampling process yields a map reconstruction with an arbitrary number of points.*

weights are renormalized via Eqs. (7.1) and (7.2):

$$N^* = N_i + N_j \tag{7.1}$$

$$\boldsymbol{\pi}^* = \begin{bmatrix} \frac{N_i\pi_1}{N^*} & \frac{N_i\pi_2}{N^*} & \cdots & \frac{N_i\pi_m}{N^*} & \frac{N_j\tau_1}{N^*} & \frac{N_j\tau_2}{N^*} & \cdots & \frac{N_j\tau_k}{N^*} \end{bmatrix}^T \tag{7.2}$$

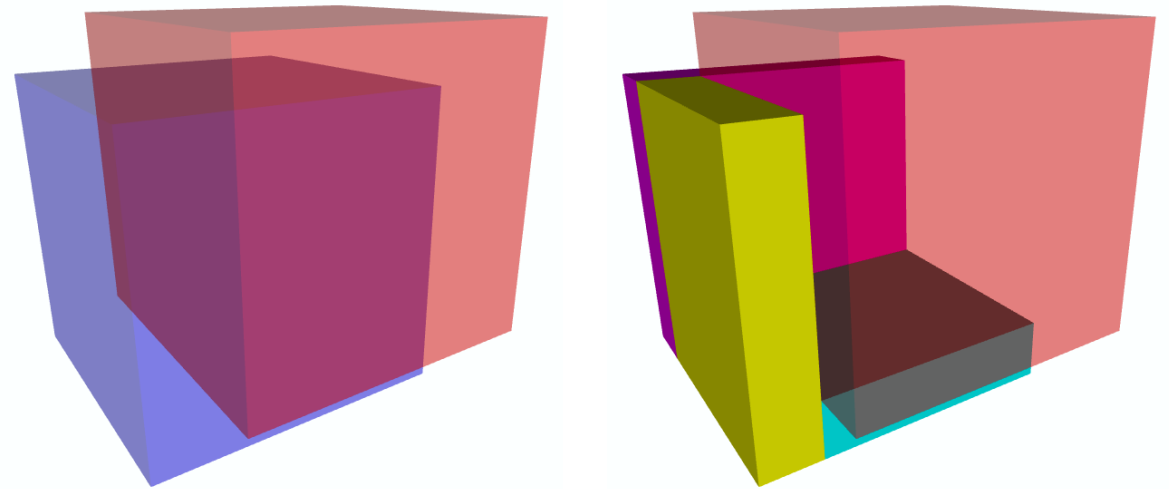
where $m \in [1, \ldots, M]$ and $k \in [1, \ldots, K]$ denote the mixture component in GMMs $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$, respectively. $N^* \in \mathbb{R}^1$ is the sum of the support sizes of $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$. $\boldsymbol{\pi}^* \in \mathbb{R}^{M+K}$ are the renormalized weights. The means and covariances are merged by concatenation.

Distinct free $\mathcal{F}(\boldsymbol{x})$ and occupied $\mathcal{G}(\boldsymbol{x})$ GMMs are maintained to compactly represent the density of points observed in the environment. The process by which $\mathcal{F}(\boldsymbol{x})$ and $\mathcal{G}(\boldsymbol{x})$ are created is illustrated in Figs. 7.3a and 7.3e. Because the GMM is a generative model, one may sample from the surface model to generate points and reconstruct occupancy. A point $\hat{\boldsymbol{p}} \in \mathbb{R}^3$ is sampled from component m of GMM $\mathcal{G}(\boldsymbol{x}) \propto \pi_m$ by decomposing the covariance $\boldsymbol{\Lambda}_m$ into the eigenvectors $\boldsymbol{U}$ and a diagonal matrix consisting of the eigenvalues $\boldsymbol{D}$ (shown in Eq. (7.3)). The entries of a point $\boldsymbol{p} \in \mathbb{R}^3$ are drawn from the univariate Gaussian probability density function with mean $\mu = 0$ and unit variance $\sigma^2 = 1$, $\mathcal{N}(0, 1)$. All entries larger than three-sigma are discarded so that the sensor observation accurately represents 97% of the distribution. The point $\boldsymbol{p}$ is then transformed via Eq. (7.4).

$$\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T = \boldsymbol{\Lambda}_m \tag{7.3}$$

$$\hat{\boldsymbol{p}} = \boldsymbol{U}\boldsymbol{D}^{1/2}\boldsymbol{p} + \boldsymbol{\mu}_m \tag{7.4}$$

Figure 7.3 illustrates the model used to compactly represent the sensor observation.



(a) Overlapping bounding boxes (b) Set difference $b_{t+1} \setminus b_t$

Figure 7.4: (a) The blue bounding box b_{t+1} is centered around $\mathcal{X}_{t+1}$ and red bounding box b_t is centered at $\mathcal{X}_t$. (b) illustrates the bounding boxes contained in b_{t+1} and not in b_t in solid magenta, teal, and yellow colors. The magenta, teal, and yellow bounding boxes represent the set difference $b_{t+1} \setminus b_t$.

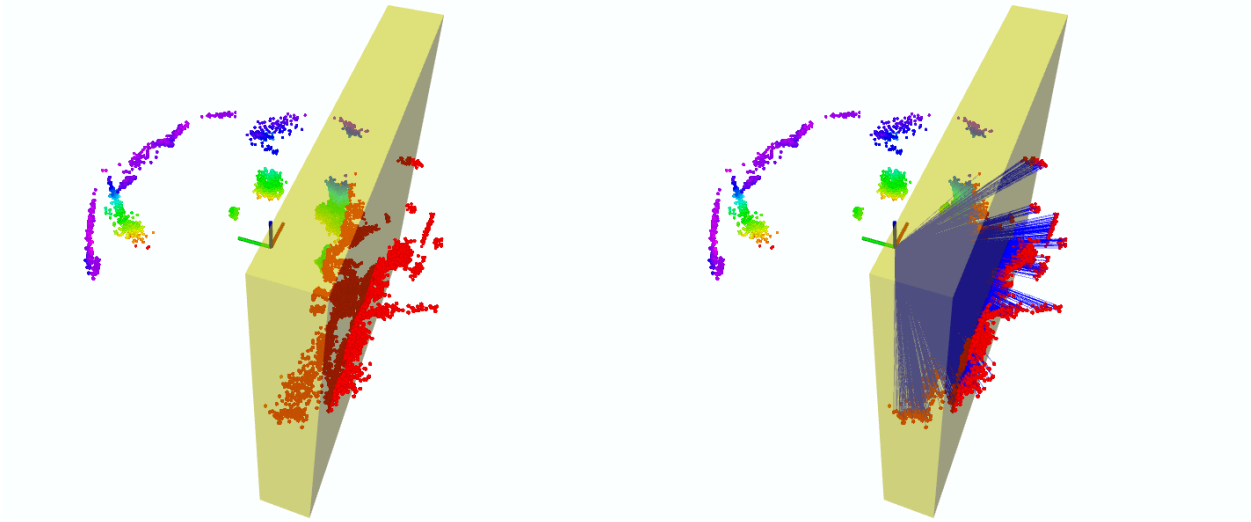
Local Occupancy Grid Map

The occupancy grid map [85] is a probabilistic representation that discretizes 3D space into finitely many grid cells $\mathbf{o} = \{o_1, ..., o_{|\mathbf{o}|}\}$. Each cell is assumed to be independent and the probability of occupancy for an individual cell is denoted as $p(o_i|\mathcal{X}_{1:t}, \mathcal{Z}_{1:t})$ where $\mathcal{X}_{1:t}$ denotes all vehicle states up to and including time t and $\mathcal{Z}_{1:t}$ denotes the corresponding observations. Unobserved grid cells are assigned a uniform prior of 0.5 and the occupancy value of the grid cell o_i at time t is expressed using log odds notation for numerical stability.

$$l_{t,i} \triangleq \log\left(\frac{p(o_i|\mathcal{Z}_{1:t}, \mathcal{X}_{1:t})}{1 - p(o_i|\mathcal{Z}_{1:t}, \mathcal{X}_{1:t})}\right)$$

When a new measurement $\mathcal{Z}_t$ is obtained, the occupancy value of cell o_i is updated as

$$l_{t,i} \triangleq l_{t-1,i} + L(o_i|\mathcal{Z}_t) - l_0$$



(a) (b)

Figure 7.5: Given a sensor observation at the origin, the resampled pointcloud shown in Fig. 7.3f, and the novel bounding box shown in yellow, each ray from the sensor origin to the point is tested to determine if it intersects the bounding box. (a) illustrates the endpoints of rays that intersect the bounding box in red. (b) illustrates how the bounding box occupancy values are updated. Endpoints inside the yellow volume update cells with an occupied value. All other cells along the ray are updated to be free.

where $L(o_i|\mathcal{Z}_t)$ denotes the inverse sensor model of the robot and l_0 is the prior of occupancy [85].

Let $\bar{\boldsymbol{o}}_t$ denote a local occupancy grid map centered at the robot's pose, $\mathcal{X}_t$. $\bar{\mathbf{o}}_t$ may be represented as a vector of fixed size such that every entry corresponds to a voxel within a 3D bounding box. $\bar{\boldsymbol{o}}$ represents a significantly smaller area than the environment. If the robot moves a small amount between times t and $t+1$, the maps at times $\bar{\boldsymbol{o}}_t$ and $\bar{\boldsymbol{o}}_{t+1}$ will not cover exactly the same region but may overlap. Given bounding boxes b_t and b_{t+1}, the set difference $b_{t+1} \setminus b_t$ is used to compute at most three non-overlapping bounding boxes (see Fig. 7.4). The intersection of the bounding boxes remains up-to-date, but the occupancy of the novel bounding boxes must be reconstructed using the surface models $\mathcal{G}(\boldsymbol{x})$ and $\mathcal{F}(\boldsymbol{x})$. Raytracing is an expensive operation [2], so eliminating the voxels at the intersection of b_t and b_{t+1} by using three novel bounding boxes avoids unnecessary calculations and saves time.

The local occupancy grid map at time $t+1$, $\bar{\boldsymbol{o}}_{t+1}$, is initialized by copying the voxels from b_t that lie at the intersection of b_{t+1} and b_t. In practice, the time to copy the local occupancy

grid map is very low (on the order of a few tens of milliseconds) as compared to the cost of raytracing through the grid. In order to identify the GMM components that intersect the bounding boxes, a KDTree [11] stores the means of the densities. A radius equal to twice the sensor's max range is used to identify the components that could affect the occupancy value of the cells in the bounding box. A ray-bounding box intersection algorithm [88] checks for intersections between the bounding box and the ray from the sensor origin to density mean. Densities that intersect the bounding box are extracted into local submaps $\bar{\mathcal{G}}(\boldsymbol{x})$ and $\bar{\mathcal{F}}(\boldsymbol{x})$ via Eqs. (7.1) and (7.2). Points are sampled from each distribution by following Eqs. (7.3) and (7.4) and raytraced to their corresponding sensor origin to update the local grid map.

As the number of mixture components in the distribution increases over time in one region, updating the occupancy becomes increasingly expensive as the number of points needed to resample and raytrace increases. To limit this potentially unbounded number of points, a small, fixed-size bounding box around the current pose with half-lengths h_x, h_y and h_z are used to determine if a prior observation was made within the confines of the bounding box. This bounding box approach works for sensors that have a 360° field of view such as the 3D LiDAR used in this work, but does not readily extend to depth sensors with smaller fields of view. If a prior observation was made within the bounding box, the current observation, $\mathcal{Z}_t$ is not stored as a GMM.

7.1.2 Planning for Exploration

The planning framework consists of action generation and action selection. Action generation refers to the design of candidate actions for the planner, while action selection in the exploration context refers to the planning policy that selects safe, feasible trajectories to minimize the uncertainty of the map over time.

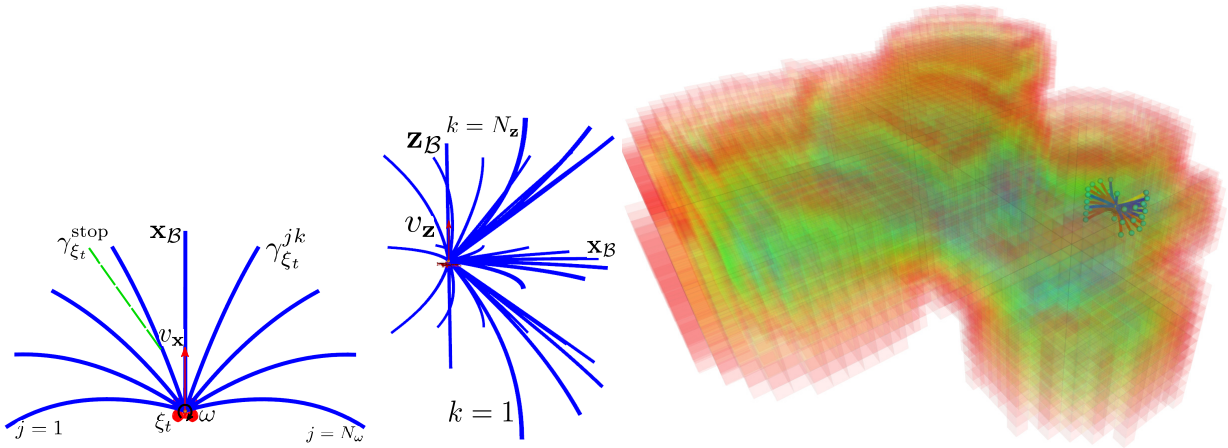
Action Generation

Accurate position control of multirotors presumes continuity in supplied references up to high-order derivatives of position [54]. Actions that satisfy continuity requirements must be computable in real-time. Forward arc motion primitives [90], generated via forward propagation of unicycle kinematics with higher-order endpoint constraints, have been successfully demonstrated in high-speed multirotor exploration and teleoperation scenarios [34, 76]. An action representation similar to the one proposed in [34] is employed in this work. For the differentially-flat multirotor state at time t, $\boldsymbol{\xi}_t$, denote the action parameterization as $\mathbf{a} = [v_{\mathbf{x}}, v_{\mathbf{z}}, \omega]$ where $v_{\mathbf{x}}$ and $v_{\mathbf{z}}$ are velocities in the body frame $\mathbf{x}_{\mathcal{B}}$ and $\mathbf{z}_{\mathcal{B}}$ directions, and ω is the body frame angular rate about $\mathbf{z}_{\mathcal{B}}$ axis. Actions are discretized using the user-specified maximum velocity bounds in $\mathbf{x}_{\mathcal{B}} - \mathbf{y}_{\mathcal{B}}$ plane (ω variation, N_ω primitives) and $\mathbf{z}_{\mathcal{B}}$ plane ($v_{\mathbf{z}}$ variation, $N_{\mathbf{z}}$ primitives) to obtain a motion primitive library (MPL) $\Gamma_{\boldsymbol{\xi}_t}$ given by (Figs. 7.6a and 7.6b):

$$\Gamma_{\boldsymbol{\xi}_t} = \{\gamma_{\boldsymbol{\xi}_t}^{jk} \mid j \in [1, N_\omega], k \in [1, N_{\mathbf{z}}], |\mathbf{v}| \leq V_{\max}, |\omega| \leq \Omega_{\max}\} \tag{7.5}$$

where $|\mathbf{v}|$ is the norm of $v_{\mathbf{x}}$ and $v_{\mathbf{z}}$, and $V_{\max}$ and $\Omega_{\max}$ are user-specified bounds on linear and angular velocities respectively.

For a given action discretization, the motion primitive $\gamma_{\boldsymbol{\xi}_t}^{jk}$ is generated as an 8^{th} order polynomial in time using start- and end-point velocities, keeping position unconstrained. End-point velocity is obtained by forward propagating a unicycle kinematics model using the current state, maximum duration of the motion primitive (τ), and the available action parameterization. Higher-order derivatives from acceleration to snap are constrained to zero at endpoints:



(a) ω variation (b) $v_\mathbf{z}$ variation (c) Frontier Distance Field

Figure 7.6: Forward-arc motion primitives from the multirotor state $\boldsymbol{\xi}_t$ [76] are obtained after varying yaw rate (a) and vertical velocity (b). Stopping trajectories $\gamma^{stop}_{\boldsymbol{\xi}_t}$ are always computed for each primitive (green, dashed) to ensure safety. An explored local occupancy map is used to construct a distance field for frontiers (c) [19], which enables computation of global reward as detailed in Section 7.1.2.

$$\begin{aligned} \dot{\boldsymbol{\xi}}_\tau &= [v_\mathbf{x}\cos\theta, v_\mathbf{x}\sin\theta, v_\mathbf{z}, \omega] \\ \boldsymbol{\xi}^{(n)}_\tau &= \mathbf{0} \text{ for } n = 2, 3, 4 \end{aligned} \tag{7.6}$$

where $\{.\}^{(n)}$ denotes the n^{th} time derivative.

For each MPL $\Gamma_{\boldsymbol{\xi}_t}$, an additional MPL containing stopping trajectories at any $\boldsymbol{\xi}_t$ can be sampled by fixing $\dot{\boldsymbol{\xi}}_\tau = \mathbf{0}$ ($\Gamma^{\text{stop}}_{\boldsymbol{\xi}_t}$, Fig. 7.6a). These stopping trajectories are appended to the motion primitives in $\Gamma_{\boldsymbol{\xi}_t}$ at a timestep one planning round away from the starting time. These actions ensure safety if the planner fails to compute an optimal action. The final action space χ_{act} available for the motion planner is composed of the MPLs shown in Table 7.1 [1]. The total number of actions available in this configuration of MPLs is chosen such that computation of rewards is real-time feasible.

[1]This design is dependent on the sensor model and SWaP constraints of the aerial robot. We follow the analysis of [34] for the selection of these design parameters.

MPL ID	Velocity and Duration	Direction	N_ω	$N_\mathbf{z}$	N_{prim}
1	$v_\mathbf{x}, \tau$	$\mathbf{x}_\mathcal{B}$	3	5	15
2	$v_\mathbf{x}, 2\tau$	$\mathbf{x}_\mathcal{B}$	3	5	15
3	$v_\mathbf{x}, \tau$	$-\mathbf{x}_\mathcal{B}$	3	5	15
4	$v_\mathbf{x}, 2\tau$	$-\mathbf{x}_\mathcal{B}$	3	5	15
5	$v_\mathbf{z}$	$\mathbf{z}_\mathcal{B}$	1	5	5

Table 7.1: Discretization used to construct action space χ_{act} for our simulation and hardware experiments. Total number of primitives for a MPL are denoted by $N_{prim} = N_\omega \cdot N_\mathbf{z}$.

Information-Theoretic Objective

The action selection policy uses an information-theoretic objective to maximize the information gain over time. The choice of this objective is motivated by computational feasibility onboard a compute-constrained platform. This work leverages the Cauchy-Schwarz Quadratic Mutual Information (the CSQMI is defined in Eq. (4.8)) [16] to calculate the information reward, $\mathcal{I}_\gamma$. To minimize redundant computation, $\mathcal{I}_\gamma$ is computed only at the end point of the primitive $\gamma_{\boldsymbol{\xi}_t}$. The planner uses the CSQMI to measure the information gain a candidate action will return locally, however, this design may result in myopic decision-making and only reason locally. Therefore, an additional global distribution of information is incorporated via frontiers [89]. This global reward, denoted by $\mathcal{V}_\gamma$, is calculated based on the change in distance towards a frontier along a candidate action. Using the node state $\boldsymbol{\xi}_0$, endpoint state $\boldsymbol{\xi}_\tau$, and a distance field constructed based on the position of the frontiers (see Fig. 7.6c), this reward can be calculated as $\mathcal{V}_\gamma = d(\boldsymbol{\xi}_0) - d(\boldsymbol{\xi}_\tau)$, where $d(\boldsymbol{\xi}_t)$ denotes the distance to the nearest voxel in the distance field from state $\boldsymbol{\xi}_t$ [19].

Action Selection

Using the rewards described in the preceding section, the objective for the motion planner is defined as follows [19, 34]:

Algorithm 7.1 Overview of Action Selection for Exploration

input: χ_{act}, χ_{free}
output: $\gamma^*_{\boldsymbol{\xi}_t}$ ▷ best action
for $\Gamma_{\boldsymbol{\xi}_t} \in \chi_{\text{act}}$ **do**
  for $\gamma_{\boldsymbol{\xi}_t} \in \Gamma_{\boldsymbol{\xi}_t}$ **do**
    feasible $\leftarrow$ SAFETYCHECK($\gamma_{\boldsymbol{\xi}_t}$, $\gamma^{\text{stop}}_{\boldsymbol{\xi}_t}$, χ_{free})
    if *feasible* **then**
      $\mathcal{I}_\gamma \leftarrow$ INFORMATIONREWARD($\gamma_{\boldsymbol{\xi}_t}$)
      $\mathcal{V}_\gamma \leftarrow$ FRONTIERDISTANCEREWARD($\gamma_{\boldsymbol{\xi}_t}$)
    else
      $\mathcal{I}_\gamma \leftarrow 0.0$, $\mathcal{V}_\gamma \leftarrow 0.0$
return $\gamma^*_{\boldsymbol{\xi}_t} \leftarrow \underset{\gamma_{\boldsymbol{\xi}_t} \in \chi_{\text{act}}}{\arg\max} [\mathcal{I}_\gamma + \mathcal{V}_\gamma]$

$$\begin{aligned} &\underset{\gamma_{\boldsymbol{\xi}_t}}{\arg\max}\ \mathcal{I}_\gamma + \alpha\, \mathcal{V}_\gamma \\ &\text{s.t. } \gamma_{\boldsymbol{\xi}_t} \in \chi_{\text{act}} \end{aligned} \tag{7.7}$$

where α is a positive factor used to scale the frontier distance reward. As stated earlier, the goal is to maximize this reward function in real-time on a compute-constrained aerial platform. Previous information-theoretic approaches that construct a tree and use a finite-horizon planner either do not use a global heuristic [82] or are not known to be amenable for operation on compute-constrained platforms [19]. Keeping real-time operation as a key goal in this work, a single-step planner is used with the action space χ_{act} consisting of motion primitives of varying duration (see Table 7.1). Due to this design of candidate actions, the planner is able to compute rewards over candidate actions further into the explored map from the current position. This way, even in a single-step planning formulation, longer duration candidate actions provide a longer lookahead than the case when all candidate actions are of the same duration. For both simulation and hardware experiments in this work, $\tau = 3\,\text{s}$ and $2\tau = 6\,\text{s}$ are used as candidate action durations (see Table 7.1).

The action selection procedure is detailed in Algorithm 7.1. For every candidate action $\gamma_{\boldsymbol{\xi}_t}$ in the action space χ_{act}, a safety check procedure is performed to ensure that this candidate and the

associated stopping action ($\gamma^{\text{stop}}_{\boldsymbol{\xi}_t}$) are dynamically feasible and lie within free space χ_{free} (Line 5). The free space check is performed using a Euclidean distance field created from locations of occupied and unknown spaces in the robot's local map given a fixed collision radius [18]. Checking that the stopping action is also feasible ensures that the planner never visits an inevitable collision state, which is essential for safe operation, as shown in [44]. If the action is feasible, the local information reward ($\mathcal{I}_\gamma$, Line 7) and frontier distance reward ($\mathcal{V}_\gamma$, Line 8) are calculated. The planner returns the action with the best overall reward (Line 11).

7.2 Results

This section details the experimental design to validate the proposed approach. Results are shown for 25 hours of real-time simulation trials in unstructured, 3D rich environments, as well as for field test experiments where the approach is deployed on hardware. The following shorthand is introduced for this section only: MCG will refer to the Monte Carlo GMM mapping approach and OG mapping will refer to the Occupancy Grid mapping approach. The mapping and planning software is run on an embedded Gigabyte Brix 6500U with four cores. The simulation computer has 8GB RAM and the hardware computer has 16GB RAM. All other technical specifications are identical for the simulation and hardware setups.

All simulation and hardware experiments use 10 windows each with 10 mixture components for the occupied- and free- space GMMs (200 mixture components total) with $h_x = h_y = 2\,\text{m}$ and $h_z = 1\,\text{m}$. The LiDAR has a max range of $5.0\,\text{m}$ and operates at $10\,\text{Hz}$ for both simulation and hardware. The local occupancy grid maps for both MCG and OG mapping span $20\,\text{m} \times 20\,\text{m} \times 12\,\text{m}$ at a $0.2\,\text{m}$ resolution. To calculate memory requirements for the OG mapping approach, the incremental OG map is transmitted as a changeset pointcloud where each point consists of 4 floating point numbers: $\{x, y, z, \text{logodds}\}$. The changeset is computed after incorporating each pointcloud in the occupancy grid map. A floating point number is assumed to be 4 bytes, or 32

bits. For the MCG approach, the cumulative data transferred is computed by summing the cost of transmitted GMMs. Each mixture component is transmitted as 10 floating point numbers: 6 numbers for the covariance matrix (because the covariance matrix is symmetric), three numbers for the mean, and one number for the mixture component weight. One additional number is also stored per GMM that represents the number of points from which the GMM was learned.

Before discussing the results of the exploration approaches, the choice of the windowing strategy presented in Section 7.1.1 is analyzed for reconstruction accuracy. Fig. 7.7 employs the Area under the ROC Curve (AUROC) to evaluate the accuracy of the occupancy reconstruction using the windowed and windowless GMM approaches. Ground truth probabilities for the occupancy grid map are computed by raytracing the sensor observation through a traditional occupancy grid map and updating the probabilities of occupancy via the inverse sensor model.

Given the pointcloud in Fig. 7.3a consisting of N points, the N_o occupied space points shown in colors varying from red to green are separated into 10 windows of points (shown in Fig. 7.3c). For each window $i \in [1, \ldots, 10]$, a GMM $\mathcal{G}_i(\boldsymbol{x})$ consisting of 10 mixture components is learned and merged into a single GMM $\mathcal{G}(\boldsymbol{x})$ consisting of a total of 100 mixture components (shown in red in Fig. 7.3e). The same process is repeated to generate a free space GMM $\mathcal{F}(\boldsymbol{x})$ with free space points (illustrated in blue in Fig. 7.3e). The occupancy is reconstructed by sampling and raytracing N_o points from $\mathcal{G}(\boldsymbol{x})$ and N_f points from $\mathcal{F}(\boldsymbol{x})$ such that $N = N_o + N_f$ through an occupancy grid map. Figure 7.7 illustrates the occupancy reconstruction accuracy for varying cell sizes ($0.1\,\mathrm{m}$, $0.2\,\mathrm{m}$, and $0.4\,\mathrm{m}$). All simulation and hardware experiments in this chapter use $0.2\,\mathrm{m}$ voxel sizes. The windowing approach results in a 1.4% performance decrease and a $10\times$ speed up in calculation ($1.22\,\mathrm{s}$ for the windowless approach vs. $0.12\,\mathrm{s}$ for the windowed approach) for 100-component $\mathcal{G}(\boldsymbol{x})$ and 100-component $\mathcal{F}(\boldsymbol{x})$, reducing computation time from seconds to milliseconds. Exploiting the spatial locality through windowing reduces the computational complexity associated with learning the distribution. For example, the responsibility matrix learned on all of

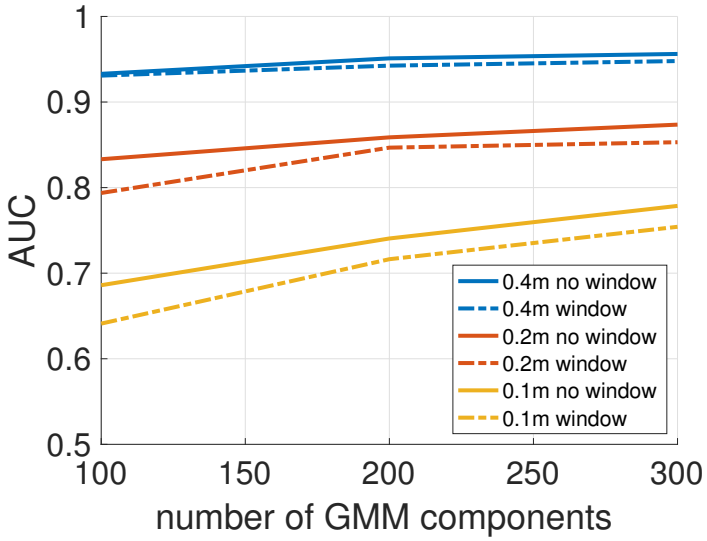


Figure 7.7: The quality of reconstruction when opting for the windowed strategy (dashed lines) shown in Fig. 7.3e as opposed to learning a distribution on all of the data (solid lines) as in Fig. 7.3b is presented. The effect of partitioning data into 10 windows, learning a distribution on each of the windows, and merging the result does not significantly reduce the quality of the reconstruction. The gains in speed are significant enough to motivate deploying the windowing strategy for real-time operations. The legend illustrates the size of the voxel used in the occupancy grid map. As expected, the reconstruction accuracy increases as the size of the voxel increases.

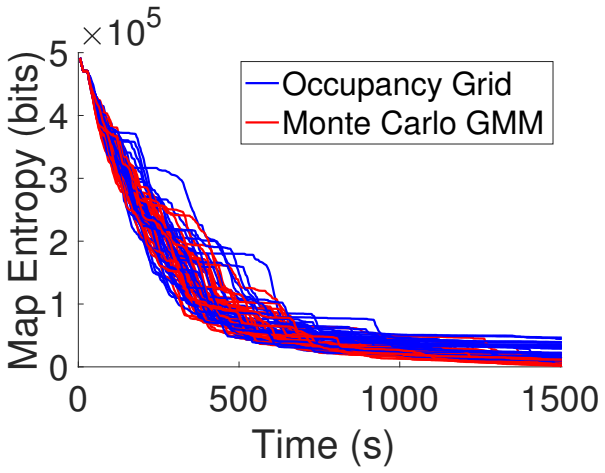
the occupied data is of size $N_o \times M$ but each windowed GMM has a responsibility matrix of size $\frac{N_o}{10} \times \frac{M}{10}$.

7.2.1 Simulation Design and Experiments

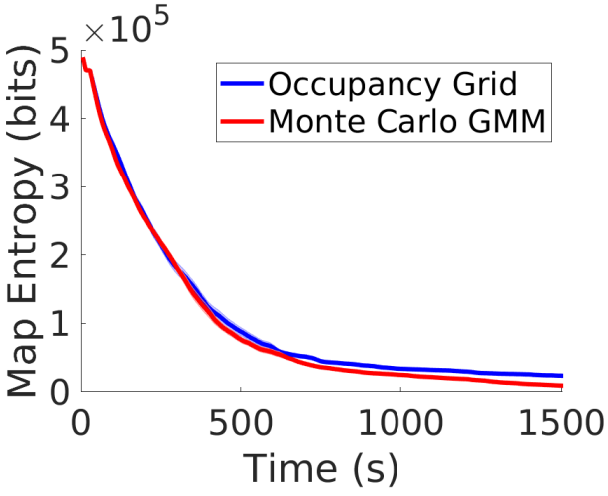
Simulation Setup

The proposed exploration strategy is evaluated with 60 real-time simulation trials over approximately 25 hours in a $30\,\mathrm{m} \times 40\,\mathrm{m} \times 6\,\mathrm{m}$ environment constructed from colorized FARO[2] pointclouds of Rapps Cave, located in Greenbrier, West Virginia (see Fig. 7.9a). In each simulation, the multirotor robot begins exploration from one of three pre-determined starting positions and explores for $1500\,\mathrm{s}$.

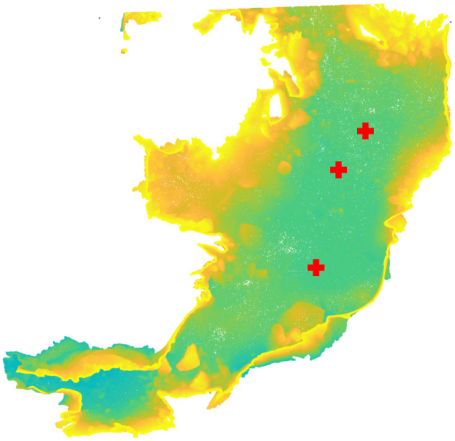
[2]A FARO is a survey grade 3D laser that emits pointclouds with color. https://www.faro.com/



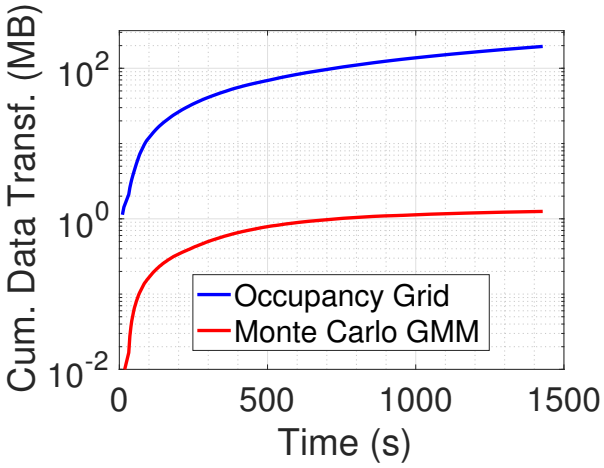
(a) Map Entropy



(b) Map Entropy

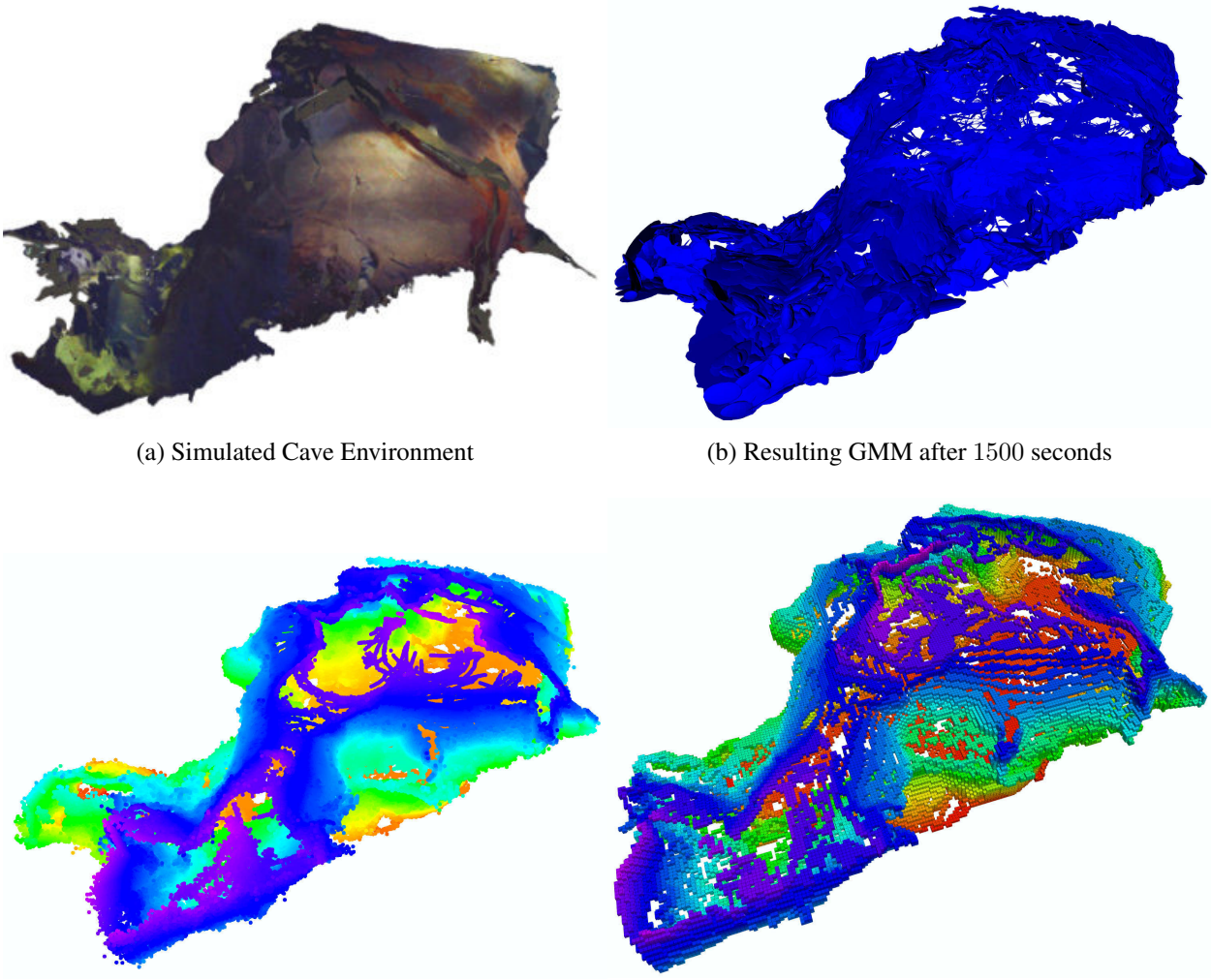


(c) Starting Positions



(d)

Figure 7.8: Exploration statistics for simulation experiments. (a) Map entropy over time for 30 trials, (b) mean map entropy over time for each method, and (c) three different starting positions (10 trials per starting position per method). Although both methods achieve similar entropy reduction, MCG uses significantly less memory according to the average cumulative data transferred shown in (d). The total amount of transmitted map data after $1500\,\mathrm{s}$ *is* $1.3\,\mathrm{MB}$ *for MCG,* $204\,\mathrm{MB}$ *for OG, and* $4.5\,\mathrm{GB}$ *(not shown) for raw pointclouds. The maximum variance for the MCG approach is* $4.2 \times 10^{-3}\,\mathrm{MB}^2$ *and* $1.2 \times 10^{2}\,\mathrm{MB}^2$ *for the OG approach. The proposed MCG method represents a decrease of two orders of magnitude as compared to the OG method.*



(a) Simulated Cave Environment

(b) Resulting GMM after 1500 seconds

(c) Resampled map from GMM

(d) Dense voxel map after 1500 seconds

Figure 7.9: Environment used for simulation experiments, shown in (a), is based on a real cave environment in West Virginia. After 1500 *seconds of autonomous exploration, the resulting MCG map (b) is densely resampled to* 10 *million points to obtain the reconstruction shown in (c). The output of the OG mapping strategy consists of 50,398 voxels and is shown in (d). Both mapping strategies leverage voxels with* 20 cm *resolution. (c) and (d) are colored according to z height.*

Results

Map entropy reduction over time is used to compare exploration performance of the MCG and OG approaches over 30 simulation trials (see Fig. 7.8a). Map entropy is measured from a global occupancy grid map that is maintained separately from the simulated robot's onboard map [21]. The simulation trials demonstrate that MCG achieves similar performance as OG, which indicates that the approximations made by the former enables real-time performance without compromising exploration or map reconstruction quality.

Figure 7.8d depicts the cumulative amount of data that must be transferred to reproduce the OG and MCG maps remotely. After $1500\,\mathrm{s}$, transferring the MCG map requires 1.3MB as compared to 204MB to incrementally transfer the OG map and 4.5GB to transfer the raw pointcloud data. The MCG approach significantly outperforms the other approaches in terms of cumulative data transfer requirements.

A representative example of the reconstructed GMM map for one trial from Fig. 7.8b is shown in Fig. 7.9b. Resampling one million points from this distribution yields the map shown in Fig. 7.9c. Note that the MCG reconstruction is smoother and has a higher fidelity than the OG reconstruction because the generative model used by the former does not assume independence between voxels.

7.2.2 Hardware Design and Experiments

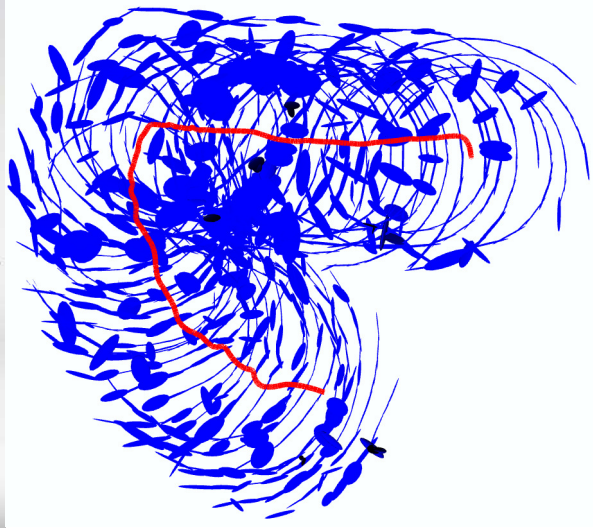
A $6.7\,\mathrm{kg}$ aerial robot (see Fig. 7.10b) equipped with a downward-facing color camera, IMU, and 3D LiDAR aerial is used to conduct field experiments.

Visual-Inertial Navigation and Control

State estimates are computed from IMU and downward facing camera observations via VINS-Mono [70], a tightly-coupled visual-inertial odometry framework that jointly optimizes vehicle motion, feature locations, camera time delay, and IMU biases over a sliding window of monocular images and pre-integrated IMU measurements. An auxiliary state estimator runs during takeoff to

(a) Robot trajectory

(b) Aerial robot with 3D LiDAR



(c) GMM Map constructed during flight

Figure 7.10: Experimental setup for hardware trials. (a) Test site for outdoor exploration experiments with robot trajectory highlighted in red. (b) Hexrotor aerial robot used in field tests. (c) Top-down view of the GMM map (blue) constructed during the 60 s *flight and the estimated trajectory (red).*

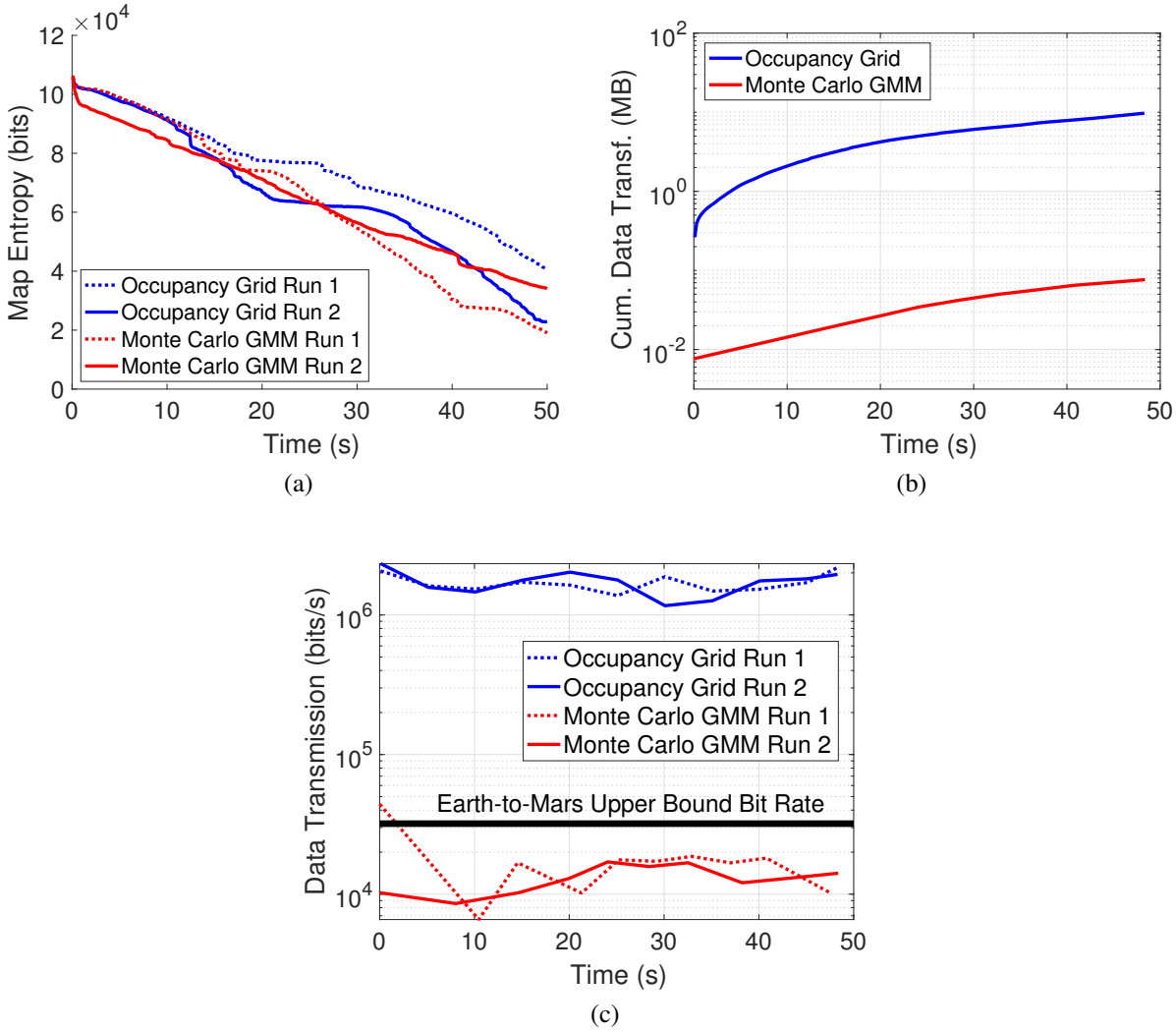


Figure 7.11: Exploration statistics for hardware experiments. (a) illustrates the map entropy reduction for two trials of each approach. (b) represents the average cumulative data transferred at a given time in a semi-logarithmic plot where the vertical axis is logarithmic labeled with successive powers of 10. The maximum variance is $2.9 \times 10^{-5}\,\mathrm{MB}^2$ *and* $7.2\,\mathrm{MB}^2$ *for the MCG and OG approaches, respectively. The total amount of data transferred at the end of each hardware trial on average is approximately* $78\,\mathrm{kB}$ *for the MCG mapping approach,* $9.7\,\mathrm{MB}$ *for the OG approach, and* $154\,\mathrm{MB}$ *when transmitting raw pointclouds (not shown). (c) illustrates the bit rate for each approach in a semi-logarithmic plot where the vertical axis is logarithmic labeled with successive powers of 10. The communications bandwidth required for the MCG approach is just under the upper bound on the available Earth-to-Mars bit rate.*

provide smooth odometry when the vehicle has not yet experienced sufficient motion excitation for VINS-Mono to initialize. The auxiliary state estimator is an unscented Kalman filter that fuses downward rangefinder altitude observations, downward optical flow, and IMU measurements to estimate vehicle velocity, position, and attitude. The loop closure functionality of VINS-Mono is disabled to avoid having relocalization-induced discontinuities in the trajectory estimate, which would have significant implications for occupancy mapping and is left as future work.

For accurate trajectory tracking, a cascaded Proportional-Derivative (PD) controller is used with a nonlinear Luenberger observer to compensate for external acceleration and torque disturbances acting on the system [55]. To improve trajectory tracking, the controller uses angular feedforward velocity and acceleration terms computed from jerk and snap references sampled from the reference trajectory's 8^{th} order polynomial (Fig. 7.2). The multirotor achieves a mean absolute position tracking error of less than $3\,\mathrm{cm}$ when flying outdoors with wind speeds of up to $29\,\mathrm{km/h}$.

Additionally, a state machine moderates the control architecture, and, by providing access to future controller references, enables seamless transitions between modes of flight operation. For example, during the hardware experiments conducted in this work, the vehicle can be in one of the five modes: (1) take-off, (2) hover, (3) tele-operation, (4) autonomous exploration, and (5) landing. The state machine also allows for an intuitive joystick interface for the operator to switch between these modes. The results presented in the next section are for the autonomous exploration mode.

Results

Four experimental trials are conducted and evaluated outdoors in inclement weather with duration of $50\,\mathrm{s}$. Given that re-localization is not enabled in state estimation, the pose estimates drift over time. To compensate for this in the analysis of exploration quality, raw sensor observations are logged onboard and post-processed to estimate ground truth poses using a FARO ground truth map (shown in Fig. 7.1b). Ground truth estimates are computed by manually aligning the first LiDAR sensor observation with the FARO map to obtain an initial pose estimate $\mathbf{T}_{fz_1} \in \mathrm{SE}(3)$,

where f denotes the FARO frame and z_1 denotes the sensor frame of the first LiDAR observation. Successive LiDAR observations are registered using GICP [74] to estimate $\mathbf{T}_{z_t z_{t+1}} \in \text{SE(3)}$ and the transform $\mathbf{T}_{f z_t}\mathbf{T}_{z_t z_{t+1}}$ is used to seed the GICP registration between the sensor observation at time $t+1$ and the FARO map. The ground truth pose estimates are used to provide a fair comparison between the two exploration approaches and generate the map entropy reduction plots shown in Fig. 7.11a. These plots illustrate almost equivalent performance between the MCG and OG mapping approaches, however, Fig. 7.11b demonstrates the MCG approach requires $100\times$ less memory as compared to the OG approach. Furthermore, Fig. 7.11c shows the reduction in the communication bandwidth required by the MCG approach compared to the OG approach, which brings the former just under the upper bound on Earth-to-Mars bit rate [46].

7.3 Conclusions

This chapter presented a method of high-fidelity perceptual modeling that is amenable to transmission across low-bandwidth communications channels. In analyzing the results of the hardware trials in the context of the Mars-to-Earth lowest communications bit rate (500 bits per second), it would take almost 30 days to transmit the raw pointcloud data, 45 hours to transmit the the occupancy grid map incrementally, and 21 minutes to transmit the GMM map [46]. At the highest bit rate of 32,000 bits per second, those numbers are changed to 0.5 days, 42 minutes, and 20 seconds, respectively. This mapping capability is an enabling technology for search and rescue, planetary exploration, and tactical operations where humans and robots must share information in real-time.

Chapter 8

Simultaneous Localization and Mapping using GMMs

The previous chapter demonstrates that there are significant advantages in representational fidelity and memory efficiency when using GMMs for perceptual and occupancy modeling without losing exploration performance. However, to enable consistent mapping in substantially unstructured and 3D environments, a method to close the loop and enable global pose updates is critical for real-time exploration. This chapter develops a GMM-based pose graph simultaneous localization and mapping framework to bridge this gap in the state of the art. The on-manifold distribution to distribution registration approach from Chapter 5 is used to estimate pose between consecutive GMMs and the Cauchy-Schwarz divergence is employed to calculate the similarity between the distributions and identify loop closures.

To enable accurate pose estimate, this chapter also presents a real-time viable method to learn GMMs by exploiting the insight that although Gaussian distributions have infinite support, a substantial amount of the support is contained within a finite region.

8.1 Approach

The chapter proceeds with a description of a reformulation of the EM algorithm to enable real-time viable calculation of GMMs. Section 8.1.2 details the distribution to distribution registration, SLAM, and loop closure approaches. Section 8.2 provides an analysis of the proposed work as compared to the state of the art and Section 8.3 concludes the chapter with a discussion of the limitations of the method.

8.1.1 Expectation Maximization

Recall the discussion of Section 3.1.2 that the K-Means algorithm is similar to the EM algorithm except that it performs a hard assignment of data points to clusters whereas the EM algorithm makes a soft assignment based on the posterior probabilities. The intuition behind the soft assignment yields one of the contributions of this chapter which is that because Gaussian densities fall off quickly, points far away from an initialized density will have a small effect on the updated parameters for that density. The responsibility matrix $\mathbf{\Gamma} \in \mathbb{R}^{N \times M}$ scales with the number of samples N and the number of components M, so to reduce the computation time, an approximation is made to ignore points that lie outside a Mahalanobis distance greater than λ for the initialized density. EM is modified to reduce the size of the responsibility matrix:

1. Initialize $\boldsymbol{\mu}_m^1$, $\mathbf{\Lambda}_m^1$ and π_m^1 with the method detailed in Section 3.1.1.

2. For a given component m, evaluate only the γ_{nm} (from Eq. (3.1)) that satisfy Mahalanobis-bound:

$$\lambda < \sqrt{(\boldsymbol{x}_n - \boldsymbol{\mu}_m^1)^T (\mathbf{\Lambda}_m^1)^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_m^1)} \tag{8.1}$$

3. Estimate the updated parameters $\boldsymbol{\mu}_m^{i+1}$, $\mathbf{\Lambda}_m^{i+1}$, and π_m^{i+1} with the current responsibilities γ_{nm} and Eqs. (3.2) to (3.4).

4. Evaluate the log likelihood (Eq. (3.5)) and iterate again from step 2 if convergence is not achieved.

8.1.2 Pose Graph SLAM via GMM Registration

Each sensor observation is represented as a GMM and successive GMMs are registered together using one of the approaches detailed in Chapter 5. Loop closures are detected by aligning observations within a given radius r of the current pose until a match is found that achieves a predetermined fitness threshold.

The pose graph is formulated as a factor graph [47] where factors represent constraints between poses, or nodes. The factor graph $G = (\mathcal{F}, \mathcal{V}, \mathcal{E})$ is composed of factor nodes $f_i \in \mathcal{F}$ and variable nodes $v_j \in \mathcal{V}$ with edges $e_{ij} \in \mathcal{E}$ connecting the factor nodes and variable nodes. The factor graph finds the variable assignment $\mathcal{V}^*$ that maximizes

$$\mathcal{V}^* = \arg\max_{V} \prod_i f_i(V_i) \tag{8.2}$$

where V_i is the set of variables adjacent to the factor f_i and independence relationships are encoded by the edges e_{ij} such that each factor f_i is a function of the variables in V_i. The pose graph uses relative constraints from GMM registration with a covariance based on the depth sensor model.

Registration

Let $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$ denote GMMs trained from sensor observations $\mathcal{Z}_i$ and $\mathcal{Z}_j$, respectively,

$$\mathcal{G}_i(\boldsymbol{x}) = \sum_m^M \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)$$
$$\mathcal{G}_j(\boldsymbol{x}) = \sum_k^K \tau_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\nu}_k, \boldsymbol{\Omega}_k)$$

and let $T(\cdot, \boldsymbol{\theta})$ denote the rigid transformation consisting of a rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$. To register $\mathcal{G}_j(\boldsymbol{x})$ into the frame of $\mathcal{G}_i(\boldsymbol{x})$, optimal rotation and translation parameters must be found such that the squared L2 norm between the distributions $\mathcal{G}_i(\boldsymbol{x})$ and $T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})$ is minimized. A rigid transformation may be applied to the GMM $\mathcal{G}_j(\boldsymbol{x})$ by the following equation

$$T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta}) = \sum_{k=1}^{K} \tau_k \mathcal{N}(\boldsymbol{x} | \boldsymbol{R}\boldsymbol{\nu} + \boldsymbol{t}, \boldsymbol{R}\boldsymbol{\Omega}_k \boldsymbol{R}^T).$$

The cost function is

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \int ||\mathcal{G}_i(\boldsymbol{x}) - T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})||_2^2 \, d\boldsymbol{x} \tag{8.3}$$

$$= \arg\min_{\boldsymbol{\theta}} - \int 2\mathcal{G}_i(\boldsymbol{x}) T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta}) \, d\boldsymbol{x} \tag{8.4}$$

which has a closed form solution as shown in Eq. (5.4) and reproduced below:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} - \sum_{m=1}^{M} \sum_{k=1}^{K} \pi_m \tau_k \mathcal{N}(\boldsymbol{\mu}_m | \boldsymbol{R}\boldsymbol{\nu}_k + \boldsymbol{t}, \boldsymbol{\Lambda}_m + \boldsymbol{R}\boldsymbol{\Omega}_k \boldsymbol{R}^T)$$

Following the approach of Section 5.1.3, the optimal rigid transformation parameters may be solved for by employing a Riemannian trust-region method with conjugate gradients [13]. The registration method used in this work is the Isoplanar variant that modifies the covariances prior to registration to smooth the cost function as sensor observations are assumed to be taken far apart from one another. The modified covariance is computed as the eigen decomposition $\boldsymbol{\Lambda}_m = \boldsymbol{U}_m \boldsymbol{D}_m \boldsymbol{U}_m^T$ and $\boldsymbol{\Lambda}_m$ is replaced with

$$\boldsymbol{\Lambda}_m = \boldsymbol{U}_m \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \epsilon \end{bmatrix} \boldsymbol{U}_m^T \tag{8.5}$$

where ϵ is a small constant representing the covariance along the normal.

The compactness of the GMM enables fast and robust registration. Because the sensor observation is represented by a small number of mixture components (several tens or hundreds) as opposed to several tens of thousands or hundreds of thousands of points, every mixture component in the source distribution may be compared to all other mixture components in the target distribution. This is in contrast to methods like ICP [7] and GICP [74] that consider the sum of squared distances between corresponding points, only.

Loop Closure

When the current estimated pose j is within a fixed radius r away from a previously visited pose i represented in the factor graph, the poses are considered candidates for loop closure. The estimated pose difference is used to seed the registration between GMMs $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$ associated with poses i and j in the factor graph. After registration, the updated pose $\boldsymbol{\theta}$ is used to transform $\mathcal{G}_j(\boldsymbol{x})$ into the frame of $\mathcal{G}_i(\boldsymbol{x})$ by applying the transform $T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})$. To determine if the GMMs partially overlap the same scene, the Cauchy-Schwarz divergence [48] is used to measure the similarity between $T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})$ and $\mathcal{G}_i(\boldsymbol{x})$. This measure is developed by Kampa et al. [48] as an entropic measure for classification for distributions. The loop closure problem may be viewed as a classification problem where the goal is to determine whether the scene under consideration by the current view has been previously observed. Kampa et al. [48] derive a closed-form expression for the Cauchy-Schwarz divergence (shown in Eq. (8.6)).

$$D_{CS}\Big(\mathcal{G}_i(\boldsymbol{x}), T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})\Big) = -\log\left(\frac{\int \mathcal{G}_i(\boldsymbol{x}) T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta}) d\boldsymbol{x}}{\sqrt{\int \mathcal{G}_i(\boldsymbol{x})^2 d\boldsymbol{x} \int T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})^2 d\boldsymbol{x}}}\right) \tag{8.6}$$

Equation (8.6) may be efficiently calculated by distributing the integral into the summation and using the Gaussian identity:

$$
\begin{aligned}
&D_{CS}(\mathcal{G}_i(\boldsymbol{x}), T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta})) = \\
&-\log\left(\sum_{m=1}^{M}\sum_{k=1}^{K}\pi_m\tau_k\mathcal{N}(\boldsymbol{\mu}_m|\boldsymbol{R\nu}_m+\boldsymbol{t}, \boldsymbol{\Lambda}_m+\boldsymbol{R\Omega}_k\boldsymbol{R}^T)\right) \\
&+\frac{1}{2}\log\left(\sum_{m=1}^{M}\frac{\pi_m^2|2\boldsymbol{\Lambda}_m|^{-1/2}}{(2\pi)^{D/2}}+2\sum_{m=1}^{M}\sum_{m'<m}\pi_m\pi_{m'}\mathcal{N}(\boldsymbol{\mu}_m|\boldsymbol{\mu}_{m'}, \boldsymbol{\Lambda_m}+\boldsymbol{\Lambda_{m'}})\right) \\
&+\frac{1}{2}\log\left(\sum_{k=1}^{K}\frac{\tau_k^2|2\boldsymbol{R\Omega}_k\boldsymbol{R}^T|^{-1/2}}{(2\tau)^{D/2}}+2\sum_{k=1}^{K}\sum_{k'<k}\tau_k\tau_{k'}\mathcal{N}(\boldsymbol{R\nu}_k+\boldsymbol{t}|\boldsymbol{R\nu}_{k'}+\boldsymbol{t}, \boldsymbol{R\Omega_k R^T}+\boldsymbol{R\Omega_{k'}R^T})\right)
\end{aligned}
$$

This measure of similarity is not a metric because it does not satisfy the triangle inequality, but $0 \leq D_{CS} \leq \infty$ and the measure satisfies the symmetry property, meaning that $D_{CS}(\mathcal{G}_i(\boldsymbol{x}), \mathcal{G}_j(\boldsymbol{x})) = D_{CS}(\mathcal{G}_j(\boldsymbol{x}), \mathcal{G}_i(\boldsymbol{x}))$. Furthermore, the distributions $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$ are the same only when $D_{CS} = 0$. The Cauchy-Schwarz divergence has the same extrema as the cost function used for registration under optimization, but has some nice additional properties useful for closing the loop. Note that because the score is always bounded to be between 0 and ∞, it must be the case that the value inside the logarithm is always between 0 and 1. Thresholding on this value is used for determining when a location has been previously visited.

If $D_{CS}(\mathcal{G}_i(\boldsymbol{x}), T(\mathcal{G}_j(\boldsymbol{x}), \boldsymbol{\theta}))$ is less than a pre-defined threshold, an edge is added between the poses i and j in the factor graph. Measuring the similarity of the distributions provides a robust threshold for determining the existence of loop closures because the distribution represents the spread of the samples in the environment. Points sample the 3D world but there is no guarantee that exactly the same point is observed from consecutive observations. Because observations are composed of thousands of points, point-based registration techniques like ICP typically only

consider distances between nearest points to remain real-time viable. By exploiting the compactness of the GMM, the similarity between all pairs of components is considered, which increases robustness.

8.2 Results

8.2.1 Experimental Design

The proposed approach is compared to the Surfel Mapping (SuMa) approach developed by [6], which is a highly optimized, parallelized implementation for GPU. The GMM formulation is run single-threaded so the timing performance is not one-for-one comparable. 450 LiDAR[1] observations are collected from a ground vehicle in a mine and 420 observations are collected from an aerial system (shown in Fig. 8.1b) in a cave.

All of the experiments are run on a low-power, embedded Gigabyte Brix with an Intel i7 8550U CPU, four cores (8 hyperthreads), and 32GB of RAM suitable for use on a SWaP-constrained robotic system. The Cauchy-Schwarz divergence loop closure threshold is set to $-\log(1 \times 10^{-6})$. Both SLAM implementations employ the GTSAM framework [23]. 100-component GMMs are used in both experiments and the parameters are kept constant for both experiments.

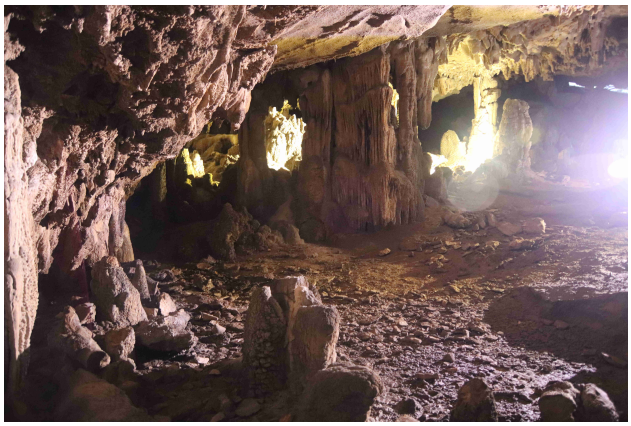
SuMa is originally developed for the Velodyne HDL-64E[2] so the following parameters are updated to work with the Velodyne VLP-16. The data width and height are changed to 500 and 16, respectively. The model width and model height are changed to 500 and 16, respectively. The fields of view up and down are changed to 15 and -15. The map max. angle is changed to 30, sigma angle to 2, and sigma distance to 2. The loop closure distance is also changed to match the setting of the GMM approach.

Two measures are used to evaluate the SLAM results. The Root Mean Square Error (RMSE)

[1] https://velodynelidar.com/vlp-16.html
[2] https://velodynelidar.com/hdl-64e.html

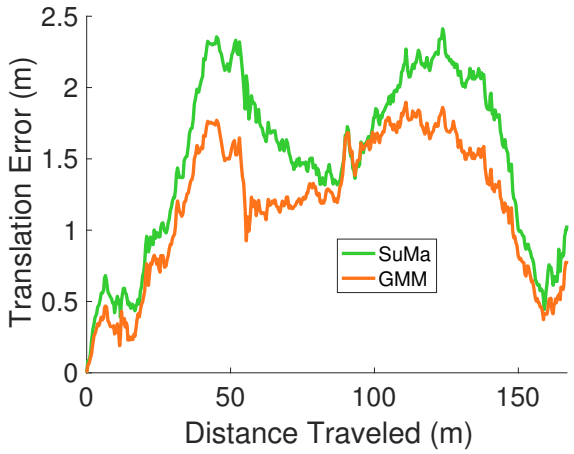
(a) Aerial system flies in Rapps Cave

(b) Aerial system

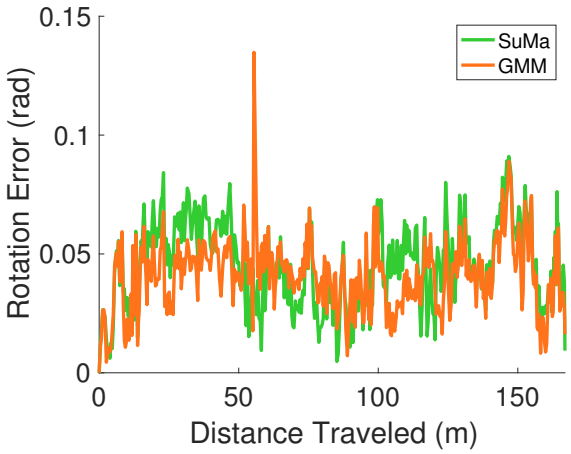


(c) Narrow passageway

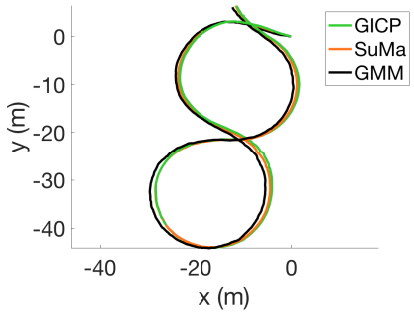
Figure 8.1: (a) A custom-built aerial system collects data in an expansive cavern of Rapps Cave in Greenbrier, WV. (b) The aerial system is equipped with a VLP-16 laser scanner. (c) The cave exhibits both expansive caverns and narrow passageways with unstructured 3D features.



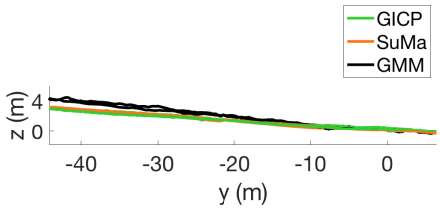
(a) Trans. Error



(b) Rot. Error



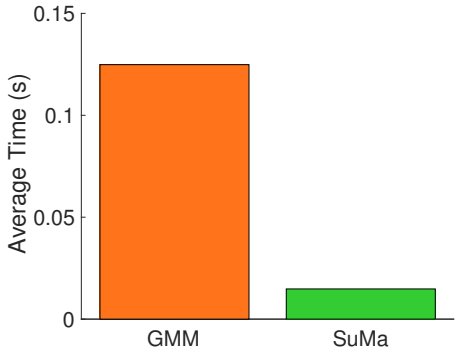
(c) Trajectory (XY View)



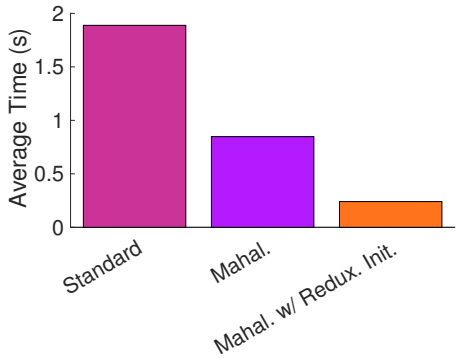
(d) Trajectory (YZ View)

	RMSE (m)
SuMa	9.88×10^{-2}
GMM	9.48×10^{-2}
	RMSE (rad)
SuMa	2.07×10^{-2}
GMM	2.11×10^{-2}

(e) RMSE



(f) Registration Times



(g) EM Variant Times

Figure 8.2: Results for Mine dataset. (a) and (b) illustrate the odometric error as a function of distance traveled. (c) and (d) present two views of the ground truth trajectory in black with each approach overlaid in different colors. (e) presents the RMSE values to evaluate the relative pose error between consecutive sensor observations. (f) presents the registration times and (g) illustrates the timing comparison from the second column of Table 8.1 as a bar chart.

as detailed in [81] and the odometric error. The RMSE is defined as the relative pose error at time step i:

$$\boldsymbol{E}_i := \left(\boldsymbol{Q}_i^{-1}\boldsymbol{Q}_{i+1}\right)^{-1}\left(\boldsymbol{S}_i^{-1}\boldsymbol{S}_{i+1}\right) \tag{8.7}$$

$$RMSE(\boldsymbol{E}_{1:n}) := \left(\frac{1}{n-1}\sum_{i=1}^{n-1}\|trans(\boldsymbol{E}_i)\|^2\right)^{1/2} \tag{8.8}$$

where $trans(\boldsymbol{E}_i)$ refers to the translational components of the relative pose error $\boldsymbol{E}_i$, the estimated trajectory $\boldsymbol{S}_1, \dots \boldsymbol{S}_n \in SE(3)$ and the ground truth trajectory $\boldsymbol{Q}_1, \dots, \boldsymbol{Q}_n \in SE(3)$. The odometric error is computed as the translation and rotation error between frames 1 and j where $j \in [1, n]$:

$$\boldsymbol{E}_j := \left(\boldsymbol{Q}_1^{-1}\boldsymbol{Q}_j\right)^{-1}\left(\boldsymbol{S}_1^{-1}\boldsymbol{S}_j\right) \tag{8.9}$$

$$OE(\boldsymbol{E}_{1:n}) := \|trans(\boldsymbol{E}_j)\| \tag{8.10}$$

For both relative pose and odometric errors, the rotation errors are similarly computed.

Ground truth for the mine dataset is provided by Near Earth Autonomy[3]. GPS is unavailable in the cave so ground truth estimates are obtained using a map generated from a survey-grade, high accuracy FARO scanner[4].

A timing analysis for training the GMM is provided in Table 8.1. The table illustrates that the Mahalanobis variant of EM from Eq. (8.1) reduces the timing by a factor of approximately 2.25. The remaining time for the Mahalanobis EM is due in large part to initialization. In order to decrease the time even further, points outside of a $15\,\mathrm{m}$ range are removed for both GMM initialization and training. This is not done for the SuMA approaches because it negatively impacts the quality of the pose estimates. The time for GMM initialization is further reduced by using a

[3] https://www.nearearth.aero/
[4] https://www.faro.com/

EM Variant	Mine Avg. Train. Time (s)	Cave Avg. Train. Time (s)
Standard EM	1.888	2.365
Mahalanobis EM	0.848	1.054
Mahalanobis EM & KMeans++ Redux	0.2406	0.322

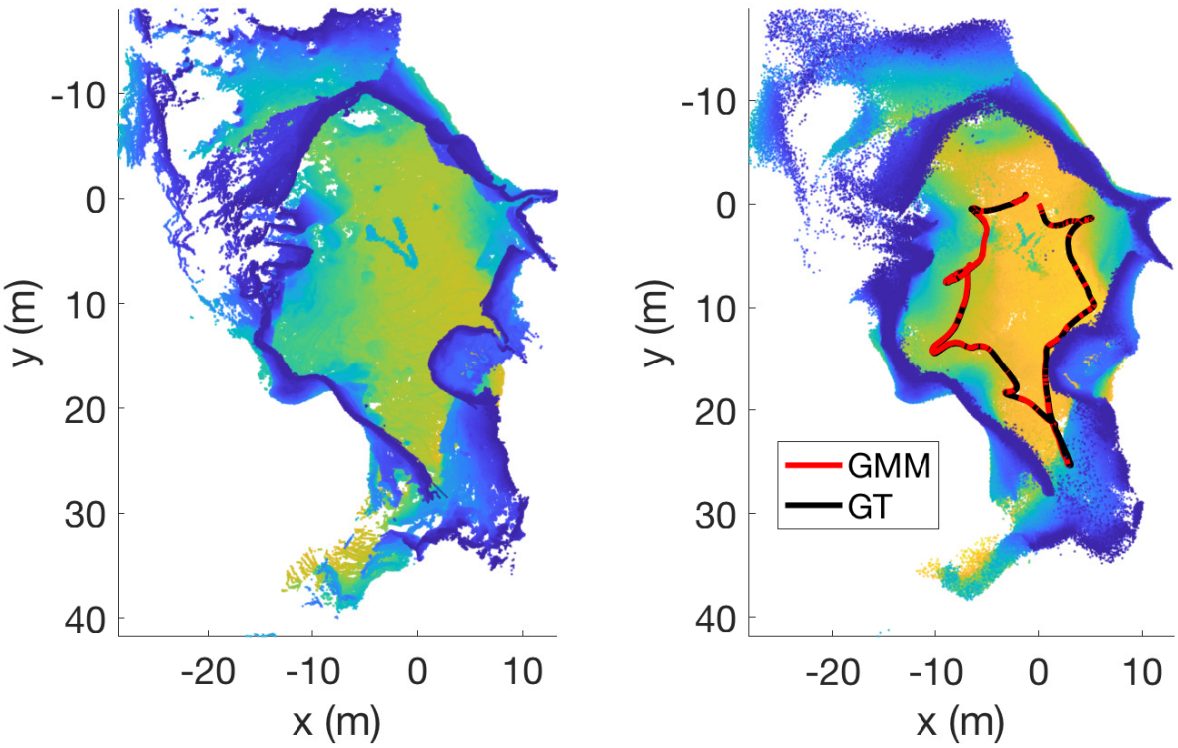
Table 8.1: Timing analysis for the Mine (also shown in the bar graph in Fig. 8.2g) and Cave (also shown in the bar graph in Fig. 8.4g) datasets. The Mahalanobis EM variant is approximately 2.25 times faster than the standard EM approach. With the K-Means++ reduction and point filtering detailed in Section 8.2.1, the runtime reduces approximately by a factor of 7.

downsampled set of points (every fifth point) for initialization and then assigning all remaining points to the closest cluster. Processing the data in this way leads to the training times labeled *Mahalanobis EM & KMeans++ Redux* in Table 8.1, which is approximately 7 times faster than the standard EM. This initialization and EM approach are used for training GMMs in the mine and cave experiments. Registration times are also provided in Figs. 8.2f and 8.4f. The frame-to-frame registration times for SuMa are reported for timing to compare fairly; however, the more accurate frame-to-model SLAM approach is used in all other plots and tables.

8.2.2 Mine

The experiment consists of 450 LiDAR observations in a mine environment taken from a ground vehicle. Figures 8.2a and 8.2b illustrate the error between the estimated pose as a function of distance traveled for the path taken by the vehicle shown in Figs. 8.2c and 8.2d. Figure 8.2e presents the RMS errors for consecutive pose estimates as a table. The translation RMSE values are slightly lower for the GMM approach and the rotation RMSE values are slightly lower for the SuMa approach. The odometric errors approximately follow the trends of the RMSE values.

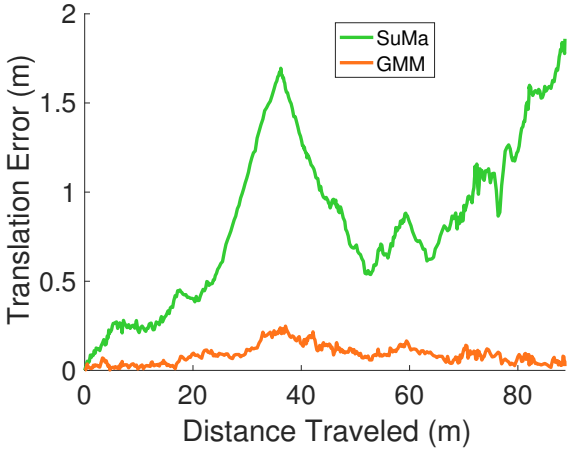
The timing results in Figs. 8.2f and 8.2g demonstrate that the Mahalanobis EM with the KMeans++ reduction approach is able to significantly reduce the time to create a GMM. SuMa takes the least time but this approach is parallelized on the integrated GPU in the 8550U.



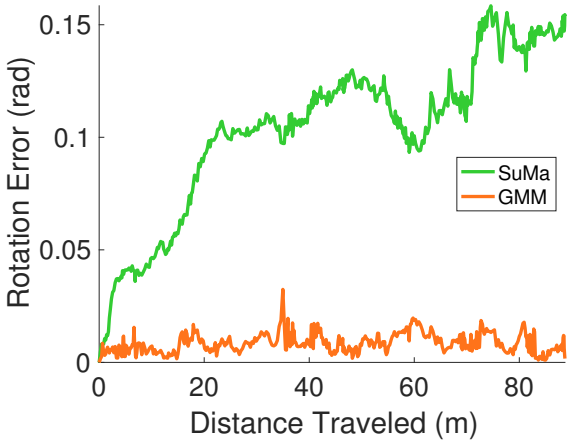
(a) Ground Truth Reconstruction (Top-Down)

(b) GMM Reconstruction (Top-Down)

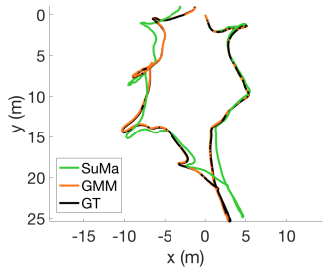
Figure 8.3: Reconstruction for the cave dataset. (a) and (b) illustrate the ground truth and GMM reconstruction of the environment model.



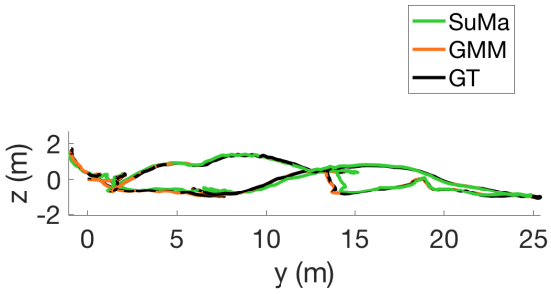
(a) Trans. error



(b) Rot. error



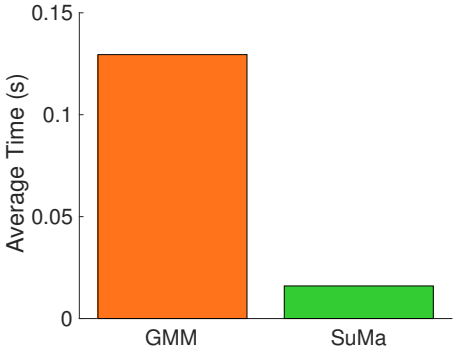
(c) Trajectory (XY View)



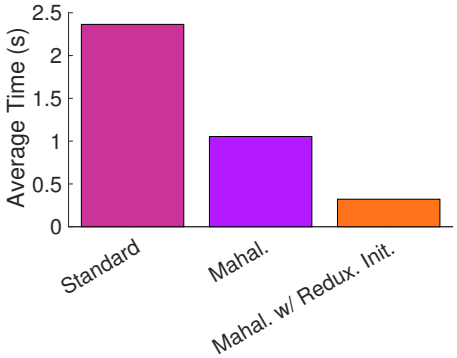
(d) Trajectory (YZ View)

	RMSE (m)
SuMa	3.78×10^{-2}
GMM	2.67×10^{-2}
	RMSE (rad)
SuMa	4.57×10^{-3}
GMM	5.05×10^{-3}

(e) RMSE



(f) Registration Times



(g) EM Variant Times

Figure 8.4: Results for Cave dataset. (a) and (b) illustrate the odometric error as a function of distance traveled. (c) and (d) present two views of the ground truth trajectory in black with each approach overlaid in different colors. (e) presents the RMSE values for consecutive sensor observations for each approach after running SLAM. (f) and (g) present a timing comparison for the registration times and EM variants, respectively.

8.2.3 Cave

Figure 8.3a illustrates a cross section of the ground truth environment model created from LiDAR observations taken from an aerial system (shown in Fig. 8.1). The path taken by the vehicle is shown in Figs. 8.4c and 8.4d. In this trial, the distance to close the loop is too large for SuMa and the error over the length of the trajectory is not decreased. This is due to one of the limitations of ICP where the cost function exhibits many local minima that are difficult to overcome when registering point-based sensor observations. Behley and Stachniss [6] attempt to overcome this limitation by trying multiple different initializations for frame-to-model ICP, but the GMM approach is better able to overcome this problem by modifying the covariances to be isoplanar (as opposed to anisotropic) which smooths the cost function and enables registration tolerant to larger changes in translation and rotation [74, 83].

Figures 8.4f and 8.4g illustrates the almost $7\times$ reduction in training time for the GMM as opposed to the standard EM method. SuMa runs on the integrated GPU in a highly optimized and parallelized way to achieve the reported runtimes. Figure 8.3b illustrates the GMM reconstruction of the environment by sampling points from the distribution.

8.3 Conclusions

While the results for the GMM approach presented in Section 8.2 cannot be run in real-time on a single thread, there is promise for a multi-threaded implementation. The distance calculations in K-Means++, responsibilities in EM, and the correspondence between pairs of mixture components in the GMM registration are all readily parallelizable. Offloading these calculations to a GPU is left as future work to enable real-time performance. This chapter demonstrated a real-time viable method for GMM SLAM for compute-constrained systems by formulating an EM algorithm that exploits sparsity to significantly reduce GMM calculation time. GMMs are trained from pointcloud

data, used to estimate pose, and build a map of the environment. A method to close the loop is presented and the approach is evaluated with real-world data of challenging mine and unstructured cave environments.

Chapter 9

Conclusion

This thesis addresses the problem of efficient exploration in subterranean environments with respect to a high-fidelity perceptual model that is compact, encodes occupancy, and is amenable to local and global pose updates. Gaussian Mixture Models are well-suited to address the challenges of operating in domains with limited communications, but are computationally intensive to learn aboard SWaP-constrained micro aerial vehicles. Theoretical limitations have prevented their use in fundamental perception tasks such as occupancy modeling and real-time pose estimation. Therefore, this thesis introduces a series of perception techniques to

1. Enable efficient exploration with respect to a high-fidelity, compact perceptual model
2. Enable accurate, real-time local and global pose updates
3. Model occupancy at arbitrary resolution

9.1 Summary of Contributions

Chapters 4 to 8 present perception techniques that achieve these objectives. Thus, the five contributions of this thesis are:

- **Information-theoretic Exploration:** A real-time, baseline information-theoretic exploration

formulation that selects actions represented on a 3D state space lattices by maximizing the mutual information between potential sensor observations and the map. (Chapter 4).

- **On-Manifold GMM Registration:** A distribution to distribution registration method to enable computationally efficient, real-time pose estimation from successive sensor observations. The approach minimizes the squared L2 norm between distributions and considers all pairs of components. For sensor observations taken far apart from one another, methods are presented to modify and smooth the cost function to enable robust registration for both LiDAR and RGB-D sensors. (Chapter 5).
- **Arbitrary Resolution Occupancy Modeling using GMMs:** An approach to derive occupancy from the perceptual model at arbitrary resolution by Monte Carlo sampling from the distribution and rayracing through a grid map to the sensor origin. Methods are presented to select the number of components for a given application and evaluate the occupancy model reconstruction performance. (Chapter 6).
- **Local Occupancy Mapping using GMMs for Exploration:** An information-theoretic exploration approach that leverages local occupancy grid maps derived from GMMs. Spatial locality is exploited to enable real-time GMM learning through a windowing strategy. Environment representation with GMMs yields substantial improvements in memory efficiency over state-of-the-art techniques. (Chapter 7).
- **Simultaneous Localization and Mapping using GMMs:** Exploiting spatial locality through the windowing technique is insufficient to enable accurate pose estimation so a method is devised to exploit the compact support during the GMM learning phase. By reducing the number of calculations during learning, a real-time viable method for simultaneous localization and mapping is realized that is amenable to global updates in pose by measuring the similarity of two distributions via the Cauchy-Schwarz divergence for GMMs. (Chapter 8).

9.2 Future Work

Several new avenues for future work have opened as a result of this work. While this work primarily develops methods for spatial and geometric information, there is promise for incorporating additional modalities not just for perceptual modeling, but also to ameliorate pose estimates. Extending the techniques proposed in this thesis to multi-robot scenarios or human-robot collaborative mapping could significantly increase performance for active perception tasks.

Appendices

Appendix A

Rotation Parameterization

To remove the need for the explicit $\mathrm{SO}(3)$ constraint, the minimal axis-angle parameterization is employed that uniquely defines a rotation through the exponential map [10]. In the axis-angle formulation, a vector $\boldsymbol{u}$ is constructed from a rotation angle $\alpha = |\boldsymbol{u}|$ and a rotation axis $\bar{\boldsymbol{u}} = \alpha^{-1}\boldsymbol{u}$. Care must be taken with this representation as α approaches π, since the cost-function wraps around at this point. In the applications discussed in this paper, the rotation angles observed are typically sufficiently small so as to avoid this issue, but in other applications explicit reasoning about this edge case may be necessary [35].

The exponential map utilized to reconstruct a rotation matrix from the axis-angle representation matrix is defined in terms of the matrix exponential as

$$\boldsymbol{R}(\boldsymbol{u}) = e^{[\boldsymbol{u}]_\times}, \quad [\boldsymbol{u}]_\times = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \tag{A.1}$$

The compact form of the Jacobian of the exponential map is given by [33] from which the Hessian

is derived as

$$\boldsymbol{R}_{u_i} = \frac{u_i[\boldsymbol{u}]_\times + [\boldsymbol{u} \times (\boldsymbol{I_3} - \boldsymbol{R})\boldsymbol{e}_i]_\times}{\|\boldsymbol{u}\|^2}\boldsymbol{R} \tag{A.2}$$

$$\begin{aligned}\boldsymbol{R}_{u_j u_i} = \Bigg[&\delta_{ij}\boldsymbol{u} + u_i\boldsymbol{e}_j + [\boldsymbol{e}_j]_\times(\boldsymbol{I_3} - \boldsymbol{R})\boldsymbol{e}_i \\ &- [\boldsymbol{u}]_\times \boldsymbol{R}_{u_j}\boldsymbol{e}_i\Bigg]_\times \frac{\boldsymbol{R}}{\|\boldsymbol{u}\|^2} + \boldsymbol{R}_{u_i}\left(\boldsymbol{R}^T\boldsymbol{R}_{u_j} - \frac{2u_j\boldsymbol{I_3}}{\|\boldsymbol{u}\|^2}\right)\end{aligned} \tag{A.3}$$

Appendix B

Derivation of Gradient and Hessian

Starting from the cost function $F = \sum_m \sum_k f_{mk}$, we note that the terms $d_{mk}^{(a)}$ and $q_{mk}^{(a)}$ are derived from $d_{mk}^{(a)} = |\boldsymbol{\Sigma}_{mk}|^{1/2} \frac{\partial}{\partial \xi_a} |\boldsymbol{\Sigma}_{mk}^{-1}|^{1/2}$, which is the partial derivative of the determinant of the precision matrix and $q_{mk}^{(a)} = -\frac{1}{2} \frac{\partial}{\partial \xi_a} \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk}$, which is the partial derivative of the scaled Mahalanobis distance in the Gaussian.

In order to expand these expressions further, we make use of the determinant identity $det(\boldsymbol{A}^n) = det(\boldsymbol{A})^n$ (Eqn. (23) [67]), the determinant derivative $\frac{\partial \det(\boldsymbol{X}^k)}{\partial \boldsymbol{X}} = k \det(\boldsymbol{X}^k) \boldsymbol{X}^{-T}$ (Eqn. (58), [67]), and the matrix chain rule $\frac{\partial g(\boldsymbol{U})}{\partial v_i} = Tr \left[\left(\frac{\partial g(\boldsymbol{U})}{\partial \boldsymbol{U}} \right)^T \frac{\partial \boldsymbol{U}}{\partial v_i} \right]$ (Eqn. (137), [67]). Beginning with $d_{mk}^{(a)}$

$$
\begin{aligned}
d_{mk}^{(a)} &= |\boldsymbol{\Sigma}_{mk}|^{1/2} Tr \left\{ \left(\frac{\partial}{\partial \boldsymbol{\Sigma}_{mk}} |\boldsymbol{\Sigma}_{mk}^{-1/2}| \right)^T \frac{\partial \boldsymbol{\Sigma}_{mk}}{\partial \xi_a} \right\} \\
&= -\frac{1}{2} Tr \left\{ |\boldsymbol{\Sigma}_{mk}|^{-1/2} \boldsymbol{\Sigma}_{mk}^{-1/2} (\boldsymbol{R}_{\xi_a} \boldsymbol{\Omega}_k \boldsymbol{R}^T + \boldsymbol{R} \boldsymbol{\Omega}_k \boldsymbol{R}_{\xi_a}^T) \right\} \\
&= -Tr \left\{ \boldsymbol{\Sigma}_{mk}^{-1/2} \boldsymbol{R}_{\xi_a} \boldsymbol{\Omega}_k \boldsymbol{R}^T \right\}
\end{aligned}
$$

For $q_{mk}^{(a)}$ we need the derivative of the matrix inverse $\frac{\partial \boldsymbol{A}^{-1}(x)}{\partial x} = -\boldsymbol{A}^{-1} \frac{\partial \boldsymbol{A}}{\partial x} \boldsymbol{A}^{-1}$ (Eqn. (59), [67]).

$$
q_{mk}^{(a)} = \frac{\partial}{\partial \xi_a} \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk}
$$

$$
\begin{aligned}
&= -\boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \frac{\partial \boldsymbol{y}_{mk}}{\partial \xi_a} - \frac{1}{2} \boldsymbol{y}_{mk}^T \frac{\partial \boldsymbol{\Sigma}_{mk}^{-1}}{\partial \xi_a} \boldsymbol{y}_{mk} \\
&= -\boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \frac{\partial \boldsymbol{y}_{mk}}{\partial \xi_a} - \frac{1}{2} \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{Z}_a \boldsymbol{\Sigma}_{mk}^{-1} \boldsymbol{y}_{mk}
\end{aligned}
$$

Where $\boldsymbol{j}_{mk}^{(a)} = -\frac{\partial R}{\partial \xi_a}\nu_k - \frac{\partial t}{\partial \xi_a}$ and $Z_{mk}^{(a)} = \frac{\partial}{\partial \xi_a}\boldsymbol{\Sigma}_{mk} = \frac{\partial}{\partial \xi_a}\boldsymbol{R}\boldsymbol{\Omega}_k\boldsymbol{R}^T = \boldsymbol{R}_{\xi_a}\boldsymbol{\Omega}_k\boldsymbol{R}^T + \boldsymbol{R}\boldsymbol{\Omega}_k\boldsymbol{R}_{\xi_a}^T$. Using these identities and noting that the second term in $q_{mk}^{(a)}$ is of the form $\boldsymbol{x}^T(\boldsymbol{P} + \boldsymbol{P}^T)\boldsymbol{x} = 2\boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x}$, we arrive at (5.9)

$$
q_{mk}^{(a)} = \boldsymbol{y}_{mk}^T \boldsymbol{\Sigma}_{mk}^{-1}(\boldsymbol{R}_{\xi_a}\boldsymbol{\nu}_k + \boldsymbol{t}_{\xi_a}) + \boldsymbol{y}_{mk}\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}\boldsymbol{\Omega}_k\boldsymbol{R}_{\xi_a}^T\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{y}_{mk}
$$

In order to derive $D_{mk}^{(ba)}$, we require the trace identity $\frac{\partial Tr(\boldsymbol{X}\boldsymbol{A})}{\partial \boldsymbol{X}} = \boldsymbol{A}^T$ (Eqn. (100), [67])

$$
\begin{aligned}
D_{mk}^{(ba)} &= -\frac{\partial}{\partial \xi_b} Tr\left\{\boldsymbol{\Omega}_k\boldsymbol{R}^T\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_a}\right\} \\
&= -Tr\left\{\left(\frac{\partial Tr\{\boldsymbol{\Omega}_k\boldsymbol{R}^T\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_a}\}}{\partial \boldsymbol{R}^T\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_a}}\right)^T \frac{\partial \boldsymbol{R}^T\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_a}}{\partial \xi_b}\right\} \\
&= -Tr\big\{\boldsymbol{\Omega}_k\big(\boldsymbol{R}_{\xi_b}^T\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_a} - \boldsymbol{R}\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{Z}_b\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_a} \\
&\qquad + \boldsymbol{R}\boldsymbol{\Sigma}_{mk}^{-1}\boldsymbol{R}_{\xi_b\xi_a}\big)\big\}
\end{aligned}
$$

$Q_{mk}^{(ba)}$ follows from $q_{mk}^{(a)}$ using the same rules employed to derive $q_{mk}^{(a)}$.

Bibliography

[1] H. Alismail, B. Browning, and S. Lucey. Direct visual odometry using bit-planes. *CoRR*, abs/1604.00990, 2016. URL http://arxiv.org/abs/1604.00990.

[2] J. Amanatides and A. Woo. A fast voxel traversal algorithm for ray tracing. In *Eurographics*, volume 87, pages 3–10, 1987.

[3] A. Arora, P. M. Furlong, R. Fitch, S. Sukkarieh, and T. Fong. Multi-modal active perception for information gathering in science missions. *Autonomous Robots*, Feb 2019. ISSN 1573-7527. doi: 10.1007/s10514-019-09836-5. URL https://doi.org/10.1007/s10514-019-09836-5.

[4] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proc. of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[5] M. Bangura, Hyon Lim, H. J. Kim, and R. Mahony. Aerodynamic power control for multirotor aerial vehicles. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, pages 529–536, Hong Kong, May 2014.

[6] J. Behley and C. Stachniss. Efficient surfel-based slam using 3d laser range data in urban environments. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018. doi: 10.15607/RSS.2018.XIV.016.

[7] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. *IEEE Trans. Pattern*

Analysis and Machine Intell., 14:239–259, February 1992.

[8] J. A. Bilmes. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

[9] C Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, New York, 2 edition, 2007.

[10] J. Blanco. A tutorial on se (3) transformation parameterizations and on-manifold optimization. (012010), October 2010.

[11] J. L. Blanco and P. K. Rai. nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees. https://github.com/jlblancoc/nanoflann, 2014.

[12] S. Bouabdallah, P. Murrieri, and R. Siegwart. Design and control of an indoor micro quadrotor. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, volume 5, pages 4393–4398, New Orleans, LA, April 2004.

[13] N. Boumal, B. Mishra, P. A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(1):1455–1459, 2014.

[14] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[15] B. Charrow, V. Kumar, and N. Michael. Approximate representations for multi-robot control policies that maximize mutual information. *Auton. Robots*, 37(4):383–400, Dec 2014. ISSN 1573-7527. doi: 10.1007/s10514-014-9411-2. URL https://doi.org/10.1007/s10514-014-9411-2.

[16] B. Charrow, S. Liu, V. Kumar, and N. Michael. Information-theoretic mapping using Cauchy-Schwarz quadratic mutual information. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Seattle, WA, May 2015.

[17] J. Conca. Environmental monitoring programs and public engagement for siting and operation of geological repository systems: Experience at the waste isolation pilot plant. In *Geological Repository Systems for Safe Disposal of Spent Nuclear Fuels and Radioactive Waste (Second Edition)*, pages 667–709. Elsevier, 2017.

[18] M. Corah and N. Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: theory and practice. *Auton. Robots*, 2018.

[19] M. Corah, C. OMeadhra, K. Goel, and N. Michael. Communication-efficient planning and mapping for multi-robot exploration in large environments. *IEEE Robotics and Automation Letters*, 4(2):1715–1721, April 2019. ISSN 2377-3766. doi: 10.1109/LRA.2019.2897368.

[20] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press and McGrawHill, Cambridge, MA, 2001.

[21] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, NY, 2012.

[22] R. R. Curtin. A dual-tree algorithm for fast k-means clustering with large k. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 300–308. SIAM, 2017.

[23] F. Dellaert. Factor graphs and gtsam: A hands-on introduction. Technical report, Georgia Institute of Technology, 2012.

[24] K. Doherty, J. Wang, and B. Englot. Bayesian generalized kernel inference for occupancy map prediction. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3118–3124. IEEE, 2017.

[25] S. Du, N. Zheng, S. Ying, and J. Liu. Affine iterative closest point algorithm for point set registration. *Pattern Recognition Letters*, 31(9):791–799, 2010.

[26] B. Eckart and A. Kelly. Rem-seg: A robust em algorithm for parallel segmentation and registration of point clouds. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, pages 4355–4362, Tokyo, Japan, November 2013.

[27] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz. Mlmd: Maximum likelihood mixture decoupling for fast and accurate point cloud registration. In *2015 International Conference on 3D Vision (3DV)*, pages 241–249. IEEE, 2015.

[28] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz. Accelerated generative models for 3d point cloud data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5497–5505, 2016.

[29] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer Society*, 22(6):46–57, June 1989. doi: 10.1109/2.30720.

[30] C. Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 147–153, 2003.

[31] G. D Evangelidis and R. Horaud. Joint alignment of multiple point sets with batch and incremental expectation-maximization. *IEEE Trans. Pattern Analysis and Machine Intell.*, 40(6):1397–1410, June 2017.

[32] N. Faiz, S. Agrawal, and R. Murray. Differentially flat systems with inequality constraints: An approach to real-time feasible trajectory generation. *J. Guid. Control Dynam.*, 24(2): 219–227, 2001.

[33] G. Gallego and A. Yezzi. A compact formula for the derivative of a 3-d rotation in exponential coordinates. *Journal of Mathematical Imaging and Vision*, 51(3):378–384, 2015.

[34] K. Goel, M. Corah, and N. Michael. Fast exploration using multirotors: Analysis, planning, and experimentation. Technical Report CMU-RI-TR-19-03, The Robotics Institute, Carnegie Mellon University, 2019.

[35] F. S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of graphics tools*, 3:29–48, March 1998.

[36] R. Hahn, D. Lang, M. Häselich, and D. Paulus. Heat mapping for improved victim detection. In *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*, pages 116–121. IEEE, 2011.

[37] G. Hamerly. Making k-means even faster. In *Proceedings of the 2010 SIAM international conference on data mining*, pages 130–140. SIAM, 2010.

[38] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3): 189–206, 2013.

[39] R. Hosseini and S. Sra. An alternative to em for gaussian mixture models: Batch and stochastic riemannian optimization. *arXiv preprint arXiv:1706.03267*, 2017.

[40] M. Irani and P. Anandan. About direct methods. In Bill Triggs, Andrew Zisserman, and Richard Szeliski, editors, *Vision Algorithms: Theory and Practice*, pages 267–277, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[41] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza. An information gain formulation for active volumetric 3d reconstruction. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 3477–3484. IEEE, 2016.

[42] M. G. Jadidi, J. V. Miro, and G. Dissanayake. Gaussian processes autonomous mapping and exploration for range-sensing mobile robots. *Autonomous Robots*, 42(2):273–290, Feb 2018. ISSN 1573-7527. doi: 10.1007/s10514-017-9668-3. URL https://doi.org/10.1007/s10514-017-9668-3.

[43] M. Jaimez and J. Gonzalez-Jimenez. Fast visual odometry for 3-d range sensors. *IEEE Transactions on Robotics*, 31(4):809–822, 2015.

[44] L. Janson, T. Hu, and M. Pavone. Safe motion planning in unknown environments: Optimal-

ity benchmarks and tractable policies. In *Proc. of Robot.: Sci. and Syst.*, Pittsburgh, PA, July 2018.

[45] B. Jian and B. C. Vemuri. Robust point set registration using gaussian mixture models. *IEEE transactions on pattern analysis and machine intelligence*, 33(8):1633–1645, 2011.

[46] NASA JPL. Mars science laboratory data rates/returns, 2019. URL https://marsmobile.jpl.nasa.gov/msl/mission/communicationwithearth/data/. (Accessed April 14, 2019).

[47] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. isam2: Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

[48] K. Kampa, E. Hasanbelliu, and J. C. Principe. Closed-form cauchy-schwarz pdf divergence for mixture of gaussians. In *The 2011 International Joint Conference on Neural Networks*, pages 2578–2585. IEEE, 2011.

[49] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, pages 2100–2106, Tokyo, Japan, November 2013.

[50] S. Kim and J. Kim. Building occupancy maps with a mixture of gaussian processes. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4756–4761. IEEE, 2012.

[51] S. Kim and J. Kim. *GPmap: A Unified Framework for Robotic Mapping Based on Sparse Gaussian Processes*. Springer International Publishing, Cham, 2015. ISBN 978-3-319-07488-7. doi: 10.1007/978-3-319-07488-7_22. URL https://doi.org/10.1007/978-3-319-07488-7_22.

[52] S. Kolouri, G. K. Rohde, and H. Hoffmann. Sliced wasserstein distance for learning gaus-

sian mixture models. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3427–3436, June 2018.

[53] R. Mahony, V. Kumar, and P. Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robot. Autom. Mag.*, 19(3):20–32, September 2012.

[54] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Shanghai, China, May 2011.

[55] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar. The grasp multiple micro-uav testbed. In *IEEE Robot. Autom. Mag.*, September 2010.

[56] J. Minguez, L. Montesano, and F. Lamiraux. Metric-based iterative closest point scan matching for sensor displacement estimation. *IEEE Trans. on Robotics*, 22(5):1047–1054, October 2006.

[57] F. Morbidi, R. Cano, and D. Lara. Minimum-energy path generation for a quadrotor UAV. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, pages 1492–1498, Stockholm, Sweden, May 2016. doi: 10.1109/ICRA.2016.7487285.

[58] M. W. Mueller, M. Hehn, and R. D'Andrea. A computationally efficient motion primitive for quadrocopter trajectory generation. *Robotics, IEEE Transactions on*, 31(6):1294–1310, December 2015.

[59] E. Nelson and N. Michael. Information-theoretic occupancy grid compression for high-speed information-based exploration. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Hamburg, Germany, September 2015.

[60] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, Basel, Switzerland, October 2011.

[61] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3d

euclidean signed distance fields for on-board mav planning. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 1366–1373. IEEE, 2017.

[62] C. O'Meadhra, W. Tabib, and N. Michael. Variable resolution occupancy mapping using Gaussian mixture models. *IEEE Robotics and Automation Letters*, page 1, 2018. doi: 10.1109/LRA.2018.2889348. Early access.

[63] U. A. Osman, M. J. Black, and A. Geiger. Patches, planes and probabilities: A non-local prior for volumetric 3d reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3280–3289, 2016.

[64] S. T. OCallaghan and F. T. Ramos. Gaussian process occupancy maps. *The International Journal of Robotics Research*, 31(1):42–62, 2012.

[65] C. Papachristos, S. Khattak, and K. Alexis. Autonomous exploration of visually-degraded environments using aerial robots. In *Unmanned Aircraft Systems (ICUAS), 2017 International Conference on*, pages 775–780. IEEE, 2017.

[66] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[67] K. B. Petersen, M. S. Pedersen, et al. The matrix cookbook, November 2012. URL `http://www2.imm.dtu.dk/pubdb/p.php?3274`. Version 20121115.

[68] M. Pivtoraiko, D. Mellinger, and V. Kumar. Incremental micro-UAV motion replanning for exploring unknown environments. In *Proc of the IEEE Int. Conf. on Robot. and Autom.*, pages 2452–2458, Karlsruhe, Germany, May 2013.

[69] J. C. Príncipe. *Information Theoretic Learning: Rényi's Entropy and Kernel Perspectives*. Springer Science & Business Media, New York, NY, 2010.

[70] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state

estimator. *IEEE Trans. Robotics*, 34(4):1004–1020, August 2018. ISSN 1552-3098. doi: 10.1109/TRO.2018.2853729.

[71] F. Ramos and L. Ott. Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. *The International Journal of Robotics Research*, 35(14):1717–1730, 2016.

[72] G. Rigas, C. Nikou, Y. Goletsis, and D. I. Fotiadis. Hierarchical similarity transformations between gaussian mixtures. *IEEE transactions on neural networks and learning systems*, 24 (11):1824–1835, 2013.

[73] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal. Normal distributions transform occupancy maps: Application to large-scale online 3d mapping. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2233–2238. IEEE, 2013.

[74] A. Segal, D. Haehnel, and S. Thrun. Generalized-icp. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009. doi: 10.15607/RSS.2009.V.021.

[75] J. Serafin and G. Grisetti. Using extended measurements and scene merging for efficient and robust point cloud registration. *Robot. Auton. Syst.*, 92:91–106, June 2017.

[76] A. Spitzer, X. Yang, J. Yao, A. Dhawale, K. Goel, M. Dabhi, M. Collins, C. Boirum, and N. Michael. Fast and agile vision-based flight with teleoperation and collision avoidance on a multirotor. In *Proc. of the Intl. Sym. on Exp. Robot.*, Buenos Aires, Argentina, 2018. Springer.

[77] S. Srivastava. Efficient, multi-fidelity perceptual representations via hierarchical gaussian mixture models. Technical Report CMU-RI-TR-17-44, Pittsburgh, PA, August 2017.

[78] S. Srivastava and N. Michael. Approximate continuous belief distributions for precise autonomous inspection. In *Safety, Security, and Rescue Robotics (SSRR), 2016 IEEE International Symposium on*, pages 74–80. IEEE, 2016.

[79] A. C. Stella-Watts, C. P. Holstege, J. K. Lee, and N. P. Charlton. The epidemiology of caving injuries in the united states. *Wilderness & environmental medicine*, 23(3):215–222, 2012.

[80] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal. Fast and accurate scan registration through minimization of the distance between compact 3d ndt representations. *The International Journal of Robotics Research*, 31(12):1377–1393, 2012.

[81] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Oct. 2012.

[82] W. Tabib, M. Corah, N. Michael, and R. Whittaker. Computationally efficient information-theoretic exploration of pits and caves. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Daejeon, Korea, October 2016.

[83] W. Tabib, C. OMeadhra, and N. Michael. On-manifold gmm registration. *IEEE Robotics and Automation Letters*, 3(4):3805–3812, 2018.

[84] T. Tamaki, M. Abe, B. Raytchev, and K. Kaneda. Softassign and em-icp on gpu. In *Proc. of the First Intl. Conf. on Networking and Computing*, pages 179–183, Higashi-Hiroshima, Japan, November 2010.

[85] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, Cambridge, MA, 2005. ISBN 0262201623.

[86] T. Waltham. Sinkhole hazard case histories in karst terrains. *Quarterly Journal of Engineering Geology and Hydrogeology*, 41(3):291–300, 2008.

[87] T. Whelan, S. Leutenegger, R. S. Moreno, B. Glocker, and A. Davison. Elasticfusion: Dense slam without a pose graph. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015. doi: 10.15607/RSS.2015.XI.001.

[88] A. Williams, S. Barrus, R. K. Morley, and P. Shirley. An efficient and robust ray-box intersection algorithm. In *ACM SIGGRAPH 2005 Courses*, page 9. ACM, 2005.

[89] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proc. of the Intl. Sym. on Comput. Intell. in Robot. and Autom.*, Monterey, CA, July 1997.

[90] X. Yang, K. Sreenath, and N. Michael. A framework for efficient teleoperation via online adaptation. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 5948–5953. IEEE, 2017.