Task-driven Navigation and Mapping with Resource Constraints

by

Beipeng Mu

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Aerospace Engineering
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
July 2016

Author .....
Department of Aeronautics and Astronautics
August 5, 2016

Certified by .....
Jonathan P. How
Richard C. Maclaurin Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by .....
John J. Leonard
Samuel C. Collins Professor, Mechanical and Ocean Engineering

Certified by .....
Sertac Karaman
Assistant Professor, Aeronautics and Astronautics

Certified by .....
John W. Fisher III
Senior Research Scientist, CSAIL

Accepted by .....
Paulo C. Lozano
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Task-driven Navigation and Mapping

with Resource Constraints

by

Beipeng Mu

Submitted to the Department of Aeronautics and Astronautics
on August 5, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Aerospace Engineering

Abstract

Breakthroughs in sensing technology in the past decade have greatly improved the capability of robots to sense complicated, partially-known environments. For example, RGB-D cameras and Velodyne scanners allow for the collection of massive amounts of sensor data in real time. These new technologies enable many new possibilities for mobile robots, such as driverless cars, drones, delivery robots, and autonomous marines vehicles. While advances in sensing technology have enabled robots to obtain data quickly and cheaply, robots are typically resource-constrained in storing and processing all the of the data. New algorithmic challenges arise that how to process data selectively to be directly useful for the robot tasks, and use sparse models to meet resource constraints.

In many of the applications, a fundamental problem for autonomous systems is the ability to simultaneously map the environment and localize within, especially when there is no global reference. This problem is often referred to as Simultaneous Localization and Mapping (SLAM). This thesis particularly studies three related key technologies in SLAM, sparse mapping, autonomous path planning and interacting with natural objects, but in the content of being task-driven and resource-constrained. In part one, given a pre-collected dataset, only a subset of landmarks and measurements of landmarks are carefully selected to build a sparse map, such that the robot still achieves good navigation performance (minimal collision) with this sparse map.

Part two extends the robot's capability to plan its own trajectories while autonomously exploring an unknown environment to build maps. A *Topological Feature Graph* is developed to maintain sparsity of the map but still enable collision check for path planning. The new approach uses a unified information metric to explicitly balance exploration of new environment and exploitation of mapped environments.

Part three uses deep neural networks to detect real-world objects as landmarks for map building. The new algorithm explicitly takes into account false positives in object detection, and performs object data association and SLAM simultaneously.

The proposed approaches are compared with existing methods using both detailed simulations as well as real-world experiments. The results show that the new

approaches have good navigation and mapping performance with significantly less memory and computation resources.

Thesis Supervisor: Jonathan P. How
Title: Richard C. Maclaurin Professor of Aeronautics and Astronautics

Committee Member: John J. Leonard
Title: Samuel C. Collins Professor, Mechanical and Ocean Engineering

Committee Member: Sertac Karaman
Title: Assistant Professor, Aeronautics and Astronautics

Committee Member: John W. Fisher III
Title: Senior Research Scientist, CSAIL

Acknowledgments

It has been an amazing journey to study at MIT. While I still feel new to MIT, it has been five years and I am still learning new things every single day. Looking back, I am always surprised by the changes that happened through the PhD study.

Five years ago, I sat on an airplane for the first time, and landed in a totally different place, knowing absolutely nobody, and didn't even speak English well. Five years later, I am getting a PhD degree from this prestigious school. I'm confident enough to speak, present and argue in public professional venues on the latest technical issues.

I appreciate so much the support from my advisors, committee members and other faculty/staff at MIT. I would first thank my supervisor Jonathan P. How. I cherish both bad and good meetings with you. While I definitely struggled many times during the study in various ways, you have always been by my the side to walk me through. It's also the difficulties that led me into deep insights into the field, and tightened my skins to confront any academic/technical challenges in future careers. I would thank my co-advisor John J. Leonard. You led me to a very different research area with has become the theme topic of my thesis. I would always remember your warm-hearted phone call when the paper got rejected and your valuable advice on my career. I would also thank my committee members Sertac Karaman and John W. Fisher, who have given tremendous support on the development of the thesis. Finally, I would also thank many other researchers and professors in LIDS, CSAIL and other laboratories, who I learned classes from, had insightful meetings, or confusing conversations but may end up in research ideas.

I would thank all the labmates and friends in Aerospace Controls Laboratory, Marine Robotics Group, LIDS and CSAIL. Thanks for bearing with all my complaints about experiments and research. Especially I would like to thank my collaborators Liam Paull, Ali-akbar Agha-mohammadi, Mathew Graham, Matthew Giamou and Shih-Yuan Liu. There were many times that I walked to you complaining how I got stuck on problems and needed help immediately. You always kindly gave help and advices, and by the time I walked out of the office, I already had plans to work on. I

would alway remembers the days fighting together to catch the deadline 3am in the morning, the days messing up the entire stata second floor with AprilTags, the days of dancing with chess boards to calibrate cameras. Thank you for sharing the fun of working on robots even at all those weekends, holidays and midnights.

I also left footprints at places like ADT, GSC, CSSA, ACF, Robotcon and CHIEF at MIT. I am proud to be with these organizations and feel lucky to meet the people there. It is also all of you that made me who I am today. Grad school is never about sitting in the office and writing computer codes, the interactions with my peers has been the true hallmark of my MIT experience. It is all of you that taught me that life could, and should be colorful, multi-dimension. It gives me a pair of glasses that see things as: we do it as we enjoy it, and we enjoy it as it makes a difference to the world to some extent.

Finally I would like to thank my funding agent ONR MURI grant W911NF-11-1-0391 and N00014-11-1-0688, without which none of the research could have happened.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The past decade has seen a boom in robotics and related applications. The development in smartphones and the gaming industry have significantly enhanced some the sensing capabilities of everyday technology. Examples include high-resolution, lightweight cameras for more accurate robot vision, RGB-D cameras that can be used to reconstruct RGB 3D models, and cheap GPS/IMUs for navigation and localization. On the other hand, robot-specific sensors also made great progress, such as Velodynes that can make wide-angle 360^o laser scans. With these devices, robots now can easily obtain data at high speed and high resolution. These new sensing abilities give the robots new potential to perform complex tasks in complicated environments. We have seen many world-changing projects emerging, such as cars drive themselves on the road, light aircrafts deliver packages, marine robots exploring remote sites, and ground robots serving food and towels in hotels. However, many of these robots are constrained on resources, such as computation, memory, communication and battery. The fast growth in sensing technology also puts new challenges on big data storage, fast processing and decision making, and real-time communication.

Among various applications, one of the core enabling capabilities for mobile robots is the ability to operate in uncertain and GPS-denied environments. In such cases, the robot would need to recognize landmarks, build a map of them, and localize itself within. If the robot position was known, for example from accurate GPS, it can use its own location information to infer landmark positions. This problem is referred

to as mapping. On the other hand, if the landmarks were easily recognizable and already mapped, the robot can use landmark information to infer its own location. This problem is referred as localization. When both robot and landmark positions are unknown, the problem of autonomous map building and localizing is often referred to as Simultaneous Localization and Mapping (SLAM). This thesis will visit three related topics in SLAM: sparse mapping, autonomous path planning, and natural landmark recognition. But we put these three problems in the content of task-driven perspective and resource constraints.

Building a map would involve using variables to represent the environment, then use measurements to infer the value of these variables. As explored space grows, the number of variables used to represent the map grows, the memory requirement will grow. As exploration time grows, the number of measurements grows, the computation requirement for inference grows. Both of memory and computation scalability issues will be problematic for robots with finite resources. To achieve scalability with finite resources, *both* the variables and measurements must be reduced. When the robot has specific tasks, building a detailed map of every inch of an environment would quickly exhaust the robot's resources without contributing to the robot tasks. We argue that the variables and measurements to be retained should be those which are most important to achieve the desired tasks. This thesis first proposes a "focused mapping" technique that tackles *both* variable and measurement selection to build sparse maps, under the task of minimal collision navigation. Given a dataset, the new approach is conducted in two stages: the first stage selects a subset of the "focused variables" for the task, which is minimizing the uncertainty of navigation. The second stage uses a task-agnostic "focused inference" method to select a subset of the measurements that maximizes the information over the focused variables. To avoid resource expensive batch selection on big datasets, an incremental approach is further developed where the two-stage procedure is conducted in a stream of batches. The variable and measurement section on the current batch is based on prior result computed from previous batch.

The developed focused mapping technique requires the availability of a pre-collected

dataset to build sparse maps, which is collected by manually operating the robot in the environment. However, in fully autonomous scenarios, the robot must generate path plans without a prior map. Active SLAM is the task of actively generating and following paths while simultaneously building a map and localizing within it. Planning while learning the map is challenging in that it is difficult to check the feasibility and optimality of a path without knowing a model of the obstacles and landmarks. The robot will also need to balance between exploring unknown regions to expand maps and exploiting visited regions for map refinement. The second part of the thesis proposes a *Topological Feature Graph* (TFG) map representation that maintains the sparsity of maps developed in part one, but enables obstacle representation for path feasibility check. An *active focused mapping* technique is developed to plan paths with TFG. The proposed approach explicitly accounts for the fact that when the robot moves to the frontiers of previously mapped regions, it can observe new landmarks from unknown regions, thus gaining new information about the environment. The information gain in observed landmarks and new landmarks is quantified with a unified metric to facilitate a natural and principled balance between exploration and exploitation.

SLAM with landmarks heavily relies on the identification of objects as unique landmarks to localize the robot. SLAM with natural objects and landmarks is challenging, in that it further requires the robot to detect the objects first, and then assign a unique identifier that can be recognized even when viewed from different perspectives and in different images. Data association refers to the problem of associating object detections to a unique identifier across different perspectives and images. The data association and SLAM problems are, individually, well-studied in the literature. But these two problems are inherently tightly coupled: Without accurate SLAM, the number of possible data associations is combinatorial and becomes intractable quickly. Without accurate data association, the error of SLAM algorithms diverges quickly. The third part of the thesis proposes a novel nonparametric graph that models data association and SLAM in a single framework. An algorithm is further introduced to alternate between inferring data association and performing SLAM.

1.1 Literature Review

This work is at the intersection of a number of robotics sub-domains. The thesis will start with representations of SLAM problems, then review some of recent results in related areas of map reduction, active SLAM, robust SLAM, object-based SLAM. This section also includes a broad review of existing techniques in landmark selection, planning with uncertainty with known maps.

1.1.1 Map Representation for SLAM

SLAM with various representations of the world and different sensors has been thoroughly studied in the literature. Occupancy grid map with LiDAR or laser range finders is among the early successes that dates back to the 1980s [1–6]. In occupancy based approaches, the world is represented by 2D/3D grids composed of free spaces and occupied spaces. To build such occupancy grid maps, new scans from the LiDAR or laser range finders are compared and matched with existing grids and the new observed parts are incrementally built into the map. This simplified representation of the world facilitates efficient computation, and thus real-time performance can be achieved on relatively large scenes with a single CPU. However, the successful matching of two scans relies on geometric features such as corners. In places that lack such features, like long hallways, SLAM using occupancy grid maps tends to fail [4]. In recent years, SLAM with 3D dense mapping and RGB-D cameras has become more and more popular [7–10]. This line of work is able to utilize both the geometric information from depth cameras and the color information from RGB cameras to reconstruct environments in 3D centimeter resolution. Incoming depth and color images are converted into volumes [7] or deformation surfaces [8], then matched with previously constructed volumes or surfaces to incrementally build the map. 3D dense maps constructed from such procedures provide photographic details of the environment with millions of volumes or surfaces. However, they rely heavily on parallel computation enabled by GPUs, and does not scale well to large size environments, such as a building.

Factor graphs are a different representations of the SLAM problem [11–14]. Instead of using small units as grids, volumes, or surfaces, factor graphs encode the poses of the robot and the observed landmarks at different poses. In a factor graph, The robot poses and landmark positions are modeled as random variables. Each factor represents a constraint on the relative poses either between two consecutive robot poses or between a robot pose and a landmark. The variables are optimized by maximizing the joint likelihood of the observed factors. To facilitate concise representation, mechanisms can be designed such that new factors are only added when there are significant pose updates or new object measurements. As a result, factor graph SLAM scales much better on bigger scenes than SLAM with occupancy grid maps or 3D dense maps. However, the convergence of factor graph SLAM algorithms relies heavily on correct data association of the landmarks. Even a single false association can cause the algorithm to diverge [15, 16].

This thesis uses factor graphs as the map representation, which enables sparsity. Furthermore, it explicitly considers data association while building maps.

1.1.2 Map Reduction

Recent work on map reduction has focused on minimizing the impact on the overall quality of the map and robot trajectory.

Original filtering-based approaches for SLAM marginalize old poses at every time step resulting in a dense information matrix, which would significantly delay inference on large scale problems. Different approaches to mitigate this have been proposed. For example, the sparse extended information filter (SEIF) is introduced to speed up the SLAM solution [17] by breaking the weak links in the graph to improve the sparsity of the information matrix.

An alternative, the exactly sparse extended information filter (ESEIF) developed in [18] selectively discards data during the measurement update step. A less conservative is proposed in [19], where the sparsification process is formulated as a constrained convex optimization to minimize the Kullback-Leibler (KL) divergence between the sparse information matrix and full information matrix subject to the accuracy and

sparsity constraints.

Graph-based optimization approaches [20] have become popular for SLAM problems. These methods provide a naturally sparse representation of the SLAM problem that can be solved efficiently [21]. Nevertheless, these methods do not scale constantly with time of exploration and space traveled, and ultimately require some form of graph reduction to enable prolonged operation.

Given the variables to be removed, blindly marginalizing them out induces a fully connected subgraph over the Markov blanket of the marginalized variable, which leads to dense graphs and significantly slows down inference on it. To avoid dense information matrix, a convex optimization can be employed to maintain sparse graphs. For example, similar to the method introduced in [19], the KL divergence between the dense subgraph and a sparse approximation is minimized subject to a consistency constraint. Carlevaris and Eustice [22] present the generic linear constraints (GLC) method that sparsifies the dense subgraph using a Chow-Liu tree (CLT) approximation. Alternately, sparsity can be enforced through an ℓ_1-regularization term in the KL divergence minimization [23], which is appealing because it does not impose a specific graph structure on the sparse approximation (e.g., a CLT). More recently, Mazuran et al. [24] improved upon previous methods by allowing non-linear measurements to approximate the dense subgraph and then minimizing the KL divergence with respect to the *measurement*, rather than *variables* or information matrix. This method works better on factors with strong nonlinearity. However, these graph reduction techniques are not concerned with selecting the nodes to be removed from the graph. Performance can degrade if the wrong landmarks are removed through marginalization since they are no longer available for subsequent loop-closures.

Another set of work on map reduction does not remove any variables, but discards measurements before they are processed by the SLAM optimizer. Kretzschmar et al. [25] propose a pose-graph compression method where laser scans are selectively removed by utilizing an approximate marginalization based on a CLT, and Ila et al. [26] used an information criterion to remove uninformative loop closures.

Instead of reducing variables or measurements individually, The method proposed

in this thesis reduces *both* variables and measurements. Furthermore, a task-focused view is adopted, where both the variables and measurements are selected based on how useful they are to support a specific task. When the robot has specific tasks, such as minimal collision navigation in this work, we show that by focusing on improving map quality in narrow passages, a significant higher compression rate could be achieved without losing navigation performance.

1.1.3 Landmark Selection

When all landmarks are perfectly mapped and given as prior information, the notion of selecting landmarks to support localization and/or mapping has been proposed to accomplish a number of different objectives. For example, actively place sensors to maximize coverage [27] and reduce navigation uncertainty [28]. A popular application in vision-based systems is to downsample landmarks based on some measure of visual saliency in the hope of improving loop closure detection. Specific applications include map compression [29], active gaze control [30], area coverage [31], and lifelong operation of service robots [32]. More related to our motivation is resource-constrained inference. For a localization objective, previously proposed approaches include uniform landmark selection [33], and entropy-based landmark selection [34].

A small number of previous works have considered resource-constrained selection of landmarks to support navigation. Strasdat et al. [35] proposed a reinforcement learning based landmark selection policy to minimize the robot position error at the goal. Lerner et al. [36] considered single camera frame based landmark selection in terms of a "severity function." Sala et al. [37] chose a minimal set of landmarks such that at least k are viewable from every point in the configuration space. None of these previous works considered the existence of obstacles, obstacle uncertainty, or probability of collision in the landmark selection process.

In contrast, this thesis selects landmarks in a SLAM setting. Landmarks are selected without a prior map and their positions need to be established. Landmarks are selected explicitly for the task of minimal collision navigation. Metric properties of the map, such as constrictions and tight corridors, are taken into accounts in the

selection process.

1.1.4 Planning under Uncertainty

In planning under uncertainty, the map is given a priori, but the robot pose has uncertainties. Much of the current literature addresses the problem of finding collision free paths in the presence of robot pose uncertainty. In the literature, the safety of a path is commonly measured by the probability of collision between the robot and an obstacle. Resulting paths can be chosen that balance the length of the paths and risk of collisions [38, 39]. In [40], an optimal path is found subject to a maximum allowable probability of collision (typically called a "chance constraint"). In [41, 42], measurement uncertainty is taken into account to compute a more accurate estimate of the robot pose and collision probabilities. The path is planned in advance assuming accurate stochastic models for motion dynamics and sensor measurements. Finally, there is a small class of planning algorithms that consider map uncertainty as well (e.g., [43]). However, these approaches are mainly limited to problems with small discrete state, action, or measurement spaces thus do not scale well.

Perhaps the most relevant work to our present approach of navigation is the work of Lambert and Le Fort-Piat [44]. A set-bounding technique is used to ensure that the robot pose estimate's 3σ ellipse never comes into contact with an obstacle. This is one of the few works to explicitly consider the pose, control, and map uncertainty explicitly. The known shortcoming of such an approach is that it tends to be overly conservative. Particularly, in the case of highly cluttered environments or tight corridors, the algorithms will fail to produce a feasible solution.

This thesis presents a focused mapping technology that is specifically designed to be utilized in probabilistic navigation. We provide a rigorous treatment of the coupling between trajectory and landmark map uncertainties, which is achieved within memory and computational constraints, making our approach applicable to low-cost robots operating with limited sensing in realistic environments.

1.1.5 Active SLAM

Existing work on active exploration focuses on designing trajectories to reduce robot pose uncertainty when the map is known or there exists a global position reference [38, 40–42, 45]. There also exists work that maximizes myopic information gain on the next action with partially known maps [46, 47]. However, in this work, the goal is to build a map of an unknown environment. The problem of actively plan path to perform SLAM is referred to as active SLAM. In this thesis, active SLAM would particularly focus on *global* map quality.

Previous work on active SLAM usually involves two types of actions that are categorized by their purposes: *exploration actions* are used to guide the robot towards unexplored regions of the map, and *exploitation actions* are used to drive the robots towards already explored regions for map refinement. The fundamental challenge of active SLAM is selecting the right metric to quantify the benefits of exploration and exploitation. A common choice for such metric is entropy reduction. For example, the work of Bourgault et al. [48] formulates the problem as a trade-off between information gain about the map and entropy reduction over the robot pose:

$$u^* = \max_u w_1 I_{SLAM}(x, u) + w_2 I_{OG}(x, u) \tag{1.1}$$

where I_{OG} is the information gained over the occupancy grid (OG) map (grid of independent binary random variables denoting occupancy) and I_{SLAM} is the information gained over of the robot poses (dependent Gaussian random variables). Similarly, Stachniss et al. [5] use a Rao-Blackwellized particle filter (RBPF) to represent the robot poses and the map, and then consider the informativeness of actions based on the expected resultant information gain. Other information metrics within a similar framework, such as the Cauchy-Schwarz quadratic mutual information [6], the D-optimality criterion [49], and the KL divergence [50] have also been proposed recently. These two information gains are computed separately and maintaining the balance between them often requires careful parameter tuning on the weights w_1 and w_2.

Recently, graph-based optimization approaches to the SLAM problem have become very popular due to their ability to exploit the naturally sparse connectivity between robot poses and landmarks in the map. These approaches have been proven to have better scalability than the RBPF approaches, which ultimately suffer from particle depletion as the size of the environment grows. Within the graph-based approaches there are two main flavors: pose graphs and landmark-based graphs. In the pose-graph approaches, sensor data is used to generate relative transformation constraints between robot poses directly, and an underlying OG map is often required to represent the environment. For example, [51, 52] optimize the robot trajectory by iteratively computing transformations between laser scans, but still maintains an underlying OG map and plans paths using sample-based approaches such as the probabilistic roadmap or the RRT* algorithm. Leung et al. propose optimizing robot trajectory with landmarks in a structured environment and also maintains an OG for collision check [53]. Information quantification over the OG map representation carries over known shortcomings of bad scalability and robustness [54]. The grid map is also an approximation because the conditional dependencies between the grid cells are discarded. For example, if there is significant drift in the robot's pose estimate, this uncertainty is not reflected explicitly in the OG map. As a result, a straight corridor will appear curved, but their relative map entropies will be equivalent. In addition, OG maps also have large memory footprints.

To our knowledge, this thesis proposes the first active SLAM approach that plans robot paths to directly optimize a global landmark-based representation of the map without any underlying OG maps. The new representation is much sparser than an OG map and gains huge advantage on scalability and computation efficiency, which is important for resource-constrained systems.

1.1.6 Object SLAM

When objects are used as landmarks for SLAM, the algorithm would need to detect the existence of objects of certain predefined classes in an image, and provide data association across images. Data association of objects and SLAM are typically solved

as decoupled problems in the literature. Pillai and Leonard [55] showed that when a SLAM solution is known, and thus there is no uncertainty in robot poses, robot poses provide good prior information about object locations and can achieve better recalls than frame by frame detections. Song et al. [56] used a SLAM solver to build a 3D map of a room, and then fixed the map and manually labeled objects in the room. On the other hand, object detection can improve localization as well. Atanasov et al. [57] pre-mapped doors and chairs as landmarks. During the navigation stage, these pre-mapped objects are detected online and their location information is used to localize the robot.

However, in this thesis, neither data association of objects nor robot poses are perfectly known. The algorithm must associate object detections to unique identifiers and perform SLAM simultaneously. Algorithms that solve object detection and SLAM jointly can be categorized into front-end approaches and back-end approaches.

In front-end approaches, objects detected in a new image are compared with previous images. If matches between new and old images are found, then corresponding objects are associated to the same unique identifier. These matches are typically reliable as the disparity between two consecutive images are usually small. When the robot comes back to a previously visited place after traversing a long distance, costly global optimization must be performed to achieve global loop closures.

These data associations by front-end procedures are taken as reliable and true, and then passed to a SLAM solver. SLAM++ [58] is one such front-end approach. Full 3D scans of chairs and tables are created and used as templates. When new point clouds are observed during testing, they are matched to pre-built templates. Successfully matched detections are often of high credibility. A SLAM solver is then run on these reliable detections to optimize object locations and camera poses. In semantic SLAM [59], Civera et al. created a library of six objects, used SURF features to detect these objects, and then ran an EKF to simultaneously localize robot and map objects.

In this thesis, instead of creating exact templates for objects, deep learning is used to detect objects in the environment. Deep learning generalizes much better

than template-based approaches. It can leverage open source software (millions of images already exist online to create models), scales easily to hundreds of object classes instead of a handful of pre-tuned templates, and does not require the objects in the scene to be exactly the same as the templates. However, the detections have a significant ratio of false positives and partial occlusions, thus are very challenging for front-end algorithms to produce reliable data associations.

1.1.7 Robust SLAM

Robust SLAM is a line of research that explicitly use back-end approaches to deal with outliers in the data [16, 60–64]. In robust SLAM, most of the object measurements are already correctly associated to unique identifiers. When some measurement is incorrectly associated, it will be inconsistent with other object measurements of the same identifier. Robust SLAM tries to identify these inconsistent measurements and eliminate them. When the remaining measurements are consistent with each other, standard SLAM solvers can be used to optimize poses.

It was demonstrated in [16] that directly using measurements with wrong identifiers leads to divergence of the pose graph SLAM solutions. The authors instead maximized a set of measurements that are consistent with each other in both identifiers and predicted locations. Only the consistent measurements are plugged into a SLAM solver to recover the robot poses and landmark locations.

By nature robust SLAM relies on the assumption that inlier measurements with unique identifier associations are the majority compared to outlier measurements. Under this assumption, eliminating outliers can still give good SLAM results. However, in object SLAM, it is often the case that there are multiple instances of the same object class. If all object measurements with same class are associated to the same identifier, different object instances will always give inconsistent measurements. In other words, in object SLAM, outliers are pervasive. If only one set of consistent measurements for each object class is kept, the algorithm will eliminate the majority of the data and fail to identify any repetitive instances of the same class.

The algorithm presented in this thesis is a back-end approach where there are

multiple instances of the same object class. The data association of object measurements to unique identifiers are considered unknown and must be established while doing SLAM. We exploit the coupling between data association and SLAM, jointly optimize both, and achieve better performance on both.

The thesis will be structured as follows: chapter 2 introduces graphical models, which is the model to represent SLAM problem thus important for understanding this thesis. Chapter 3 introduces a focused mapping technique that sub-selects both variables and measurements to build sparse maps for the task of minimal-collision navigation. Chapter 4 introduces a novel Topological Feature Graph that exploits sparsity of model built in chapter 3, but further includes obstacle representations. An active SLAM algorithm is developed with the new model to enable autonomous path planning for unknown environment exploration. To interact with natural environments, chapter 5 uses deep learning technique to detect objects as landmarks. A new Nonparametric Graphical Model is proposed to explicitly account for ambiguous data association and false positives. An inference algorithm is developed with this new model to jointly perform SLAM and data association.

The contributions of the thesis are as follows:

- Evaluate information in sensing data by their relevance to the task of navigation, develop a focused mapping technique that can build sparse models [65, 66].
- Proposed Topological Feature Graph that exploits sparsity but also incorporate obstacle representations. The resulting active mapping algorithm is able to autonomously plan robot paths to builds maps of unknown environments [67].
- Create a Nonparametric Graphical model, and develop according inference algorithm on it to associate ambiguous object detections and localize detected objects at the same time [68].
- Conduct extensive simulation as well as real-world experiments to test the proposed approaches. Results show the proposed algorithms has good task performance with significantly less resources than existing technologies.

Chapter 2

Graphical Models and SLAM

This chapter reviews background on graphical models, which is an important tool for modeling high-dimension probabilistic variables. In particular, a factor graph is a widely used model for large scale SLAM problem, which is important for understanding this thesis.

2.1 Graphical Models

2.1.1 Factor Graph

A graphical model is a probabilistic model that expresses the conditional dependence structure between random variables. Graphical models use a graph-based representation to encode a complete distribution over a high-dimensional space [69]. It is also a compact representation of independences between variables. Commonly used graphical models include Bayesian networks, Markov Random Fields, and Factor Graphs. A Bayesian network is represented as a directed acyclic graph, where edges represent conditional probabilities. Marcov Random Field, on the other hand, is represented as a undirected graph, where edges imply conditional dependencies between variables. A factor graph is represented as a bipartite graph connecting variables and factors. Each factor represents a joint function over the variables it is connected to. Figure 2-1 gives an example of such a factor graph. Circles represent random variables and

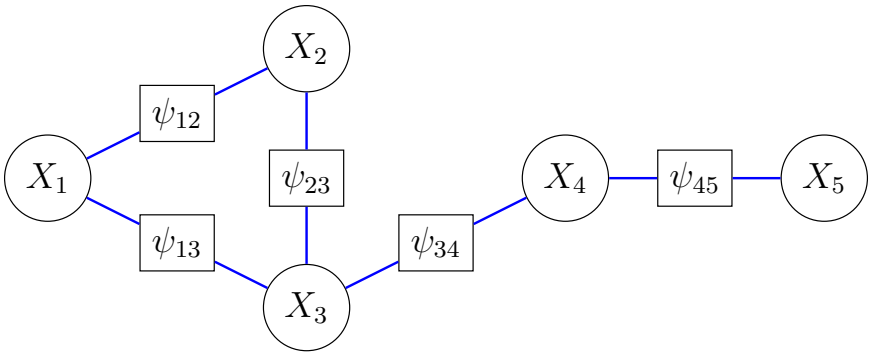


Figure 2-1: Factor graph. Squares denote factors: $\psi_{12}(X_1, X_2)$, $\psi_{23}(X_2, X_3)$, $\psi_{13}(X_1, X_3)$, $\psi_{34}(X_3, X_4)$, and $\psi_{45}(X_4, X_5)$.

squares represent factors.

For a factor graph, denote $\mathbf{X} = \{X_1, \cdots, X_n\}$ as the random variables. Denote $\psi_a(X_{\{a\}})$ as a factor among random variable in set $\{a\}$. The the joint probability can be expressed as a product of factors:

$$p(\mathbf{X} = x) \propto \prod_{a \in \mathcal{A}} \psi_a \left(X_{\{a\}} = x_{\{a\}} \right) \tag{2.1}$$

where $\mathcal{A}$ is the set of all factors. Each factor $\psi_a(x_{\{a\}})$ maps the values of random variables to a strictly positive real number representing the likelihood of the variables. It also represents probabilistic dependences among the variables in the factor.

Factor graph is an efficient representation in that it captures sparsity among variables: two variables are independent of each other given all other variables if and only if they do not belong to the same factor. The log likelihood, $\log p(x)$, can be written equivalently as:

$$\log p(x) \propto \sum_{a \in \mathcal{A}} \phi_a \left(x_{\{a\}} \right) \tag{2.2}$$

where $\phi_a(x_{\{a\}}) = \log \psi_a(x_{\{a\}})$. With graphical models, there exist fast algorithms to compute statistical properties such as marginalization, expectation, maximum likelihood [69].

A Markov blanket of a variable is the set of variables that share factors with it. For example, the Markov blanket of variable x_3 in Figure 2-1 is $\{x_1, x_2, x_4\}$. Typically, marginalizing out a variable will introduce a new factor over its Markov blanket.

2.1.2 Entropy

To quantify uncertainties in random variables, many information metrics have been used, such as entropy, mutual information, and Kullback-Leibler(KL) divergence [70]. In this thesis, Shannon entropy [71] will be used as the information metric for quantifying uncertainties.

Let X denote a random variable, and $\mathcal{X}$ denote the the set of values X can take. The Shannon entropy is defined as:

$$H(X) = \mathbb{E}\left[\log p(x)\right] \tag{2.3}$$

When X is discrete, the Shannon entropy is:

$$H(X) = -\sum_{x_i \in \mathcal{X}} p(x_i) \log(x_i) \tag{2.4}$$

and when X is continuous, the Shannon entropy is as:

$$H(X) = \int_{\mathcal{X}} p(x) \log p(x) dx \tag{2.5}$$

In cases where X is a Gaussian random variable with covariance matrix Λ, Shannon entropy has a closed form solution:

$$H(X) = \frac{1}{2} \ln\left((2\pi e)^n |\Lambda^{-1}|\right) = \frac{n}{2} \ln(2\pi e) - \frac{1}{2} \ln|\Lambda| \tag{2.6}$$

where Λ is the information matrix.

2.1.3 Gaussian Approximation

Quantifying the information gains on general factor graphs is hard, as it involves enumerating all values of all the variables. However, in the case of Gaussian distribution, there is closed form solutions for information quantification. In this way, calculating information gains can be done quickly, which is important for resource constrained

systems. This section approximates the joint probability of all factors $p(x)$ with a Gaussian distribution.

We begin by applying the standard method [70] of approximating a distribution over the variables using a second-order Taylor series expansion of the factors $\phi_{\{a\}}$ at some initial guess x^*, and denote the approximated factors as $\hat{\phi}_{\{a\}}$:[1]

$$\begin{aligned} p(x) \approx \hat{p}(x) &\propto \exp\{\sum_{a\in\mathcal{A}} \hat{\phi}_a(x_{\{a\}})\} \\ &= \exp\{\log p(x^*) + (x-x^*)^T \frac{\partial}{\partial x} \log p(x^*) + \frac{1}{2}(x-x^*)^T \frac{\partial^2}{\partial x^2} \log p(x^*)(x-x^*)\}. \end{aligned} \tag{2.7}$$

where

$$\begin{aligned} \frac{\partial}{\partial x} \log p(x^*) &= \sum_{a\in\mathcal{A}} \frac{\partial}{\partial x} \phi_a(x^*_{\{a\}}) \\ \frac{\partial^2}{\partial x^2} \log p(x^*) &= \left(\sum_{a\in\mathcal{A}} \frac{\partial^2}{\partial x^2} \phi_a(x^*_{\{a\}}) \right) \end{aligned} \tag{2.8}$$

are the gradient and Hessian of the log likelihood at point x^*.

Note that the exponential component in (2.7) is quadratic in x, therefore the approximation is a Gaussian distribution with information matrix, Λ, given by the Hessian:

$$\hat{p}(x) = \mathcal{N}(\zeta, \Lambda^{-1}) \ , \ \Lambda = \sum_{a\in\mathcal{A}} -\frac{\partial^2}{\partial x^2} \phi_a(x^*_{\{a\}}). \tag{2.9}$$

Note that Λ is inherently dependent on the chosen linearization point x^*. Typically x^* is chosen to be the maximum likelihood value of $p(x)$, so the gradient is zero, and $\hat{p}(x)$ is zero mean, more specifically $\zeta = 0$. However, computing the maximum likelihood would require nonlinear optimization over $p(x)$ and might be costly itself.

[1] The $\hat{}$ notation is used throughout to refer to the Gaussian approximation of corresponding terms in the original factor graph.

2.2 Pose Graph for SLAM

Pose graph is a widely used tool for SLAM problems in a probabilistic way. This section introduces how SLAM problems could be modeled with pose graphs. We first start with an assumption that there exist landmarks that the robot can identify to localize itself.

Assumption 1. *There exists a library of static landmarks to localize the robot in the environment. The number and locations of these landmarks is not known a priori.*

With the landmark assumption, when moving in the environment, the robot can obtain measurements of these landmarks. Given a dataset, the robot trajectory is represented as a discrete sequence of poses. Denote T as the total number of time steps, and denote $\mathbf{X}_{0:T} = \{X_0, \cdots, X_T\}$ as the robot's trajectory from the start to the end. Each robot pose consists of a position and an orientation. Denote $SE(2)$ as the space of 2D poses and $SE(3)$ as the space of 3D poses. Then $X_t \in SE(2)$ for 2D cases and $X_t \in SE(3)$ in 3D cases. In GPS-denied environments these poses are not directly observable. However, the robot can always measure the incremental change between two sequential poses via an IMU or wheel encoder, which is referred to as odometry. Denote o_t as the odometry measurement between pose x_t and pose x_{t-1}. Under the standard assumption that o_t is corrupted by additive Gaussian noise, the odometry measurement at time t can be represented as:

$$o_t = X_t \ominus X_{t-1} + v, \quad v \sim \mathcal{N}(0, Q), \tag{2.10}$$

where $\ominus$ represents an operator that takes two pose and return the relative pose between then in $SE(2)$ or $SE(3)$, and Q is the odometry noise covariance matrix. The likelihood of o_t given the two poses is then:

$$p(o_t; X_t, X_{t-1}) \sim \mathcal{N}(X_t \ominus X_{t-1}, Q) \tag{2.11}$$

During navigation, the robot also observes landmarks from the environment. Assuming that there exist M landmarks in the environment, which might be unknown

ahead of time. The positions of the landmarks are denoted as $\mathbf{L} = \{L_1, \cdots, L_N\}$. In the 2D case $L_i \in \mathbb{R}^2$, and in the 3D case $L_i \in \mathbb{R}^3$. At time t, the robot obtains K_t landmark measurements, denoted as $\mathbf{z}_t = \{z_t^1, z_t^2, \cdots, z_t^{K_t}\}$. Each measurement is associated to a unique landmark identifiers, the associations are denoted as $\mathbf{y}_t = \{y_t^1, y_t^2, \cdots y_t^{K_t}\}$, where $y_t^i \in \{1, \cdots, M\}$. For example, at time 0, the robot obtained two measurements, $z_0 = \{z_0^1, z_0^2\}$. And these 2 measurements are from landmark 5 and 7, then $y_0 = \{y_0^1, y_0^2\} = \{5, 7\}$.

Using the standard model that object measurements z_t^k are corrupted by additive Gaussian noise:

$$z_t^k = L_{y_t^k} \ominus X_t + w, \quad w \sim \mathcal{N}(0, R) \tag{2.12}$$

where R is the measurement noise matrix. The likelihood of z_t^k given the robot pose, the landmark association and landmark pose is then:

$$p(z_t^k; X_t, L_{y_t^k}) \sim \mathcal{N}(L_{z_t^k} \ominus X_t, R) \tag{2.13}$$

Combining (2.10) and (2.12), the joint log likelihood of odometry and landmark measurements is:

$$p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}) = \prod_{t=1}^{T} p(o_t; X_{t-1}, X_t) \prod_{t=0}^{T} \prod_{k=1}^{K_t} p(z_t^k; X_t, L_{y_t^k})$$

and the log likelihood is:

$$\log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}) = \sum_{t=1}^{T} \phi(o_t; X_{t-1}, X_t) + \sum_{t=0}^{T} \sum_{k=1}^{K_t} \phi(z_t^k; X_t, L_{y_t^k}) \tag{2.14}$$

where $\phi(o_t; X_{t-1}, X_t)$ and $\phi(z_t^k; X_t, L_{y_t^k})$ are odometry and landmark factors respectively. Using the probability distribution formula for Gaussian noise, it can be shown that each factor follows a quadratic form:

$$\begin{aligned} \phi(o_t; X_{t-1}, X_t) &= -\frac{1}{2} \left(X_t \ominus X_{t-1} - o_t\right) Q^{-1} \left(X_t \ominus X_{t-1} - o_t\right) \\ \phi(z_t^k; X_t, L_{y_t^k}) &= -\frac{1}{2} \left(L_{y_t^k} \ominus X_t - z_t^k\right) R^{-1} \left(L_{y_t^k} \ominus X_t - z_t^k\right) \end{aligned} \tag{2.15}$$

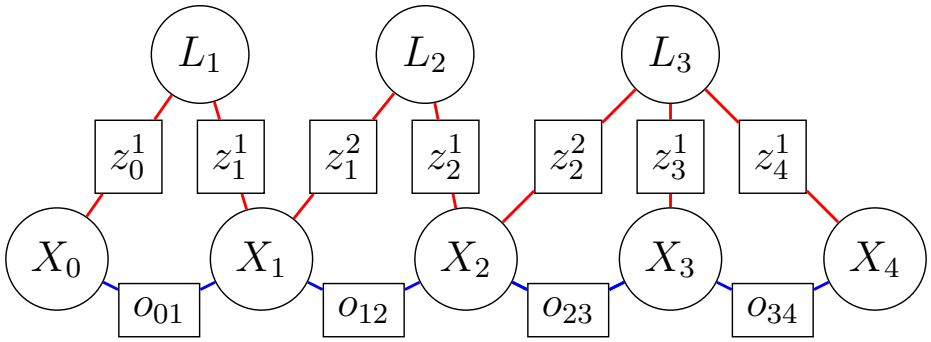


Figure 2-2: Pose Graph for SLAM. X_t denote robot poses, L_i denote landmarks, blue edges denote odometry and red edges denote landmark measurements.

Note that the log likelihood is nonlinear in X_t and L_t as $\ominus$ is a nonlinear operation in (2.10) and (2.12).

A factor graph representation for SLAM is also referred to as a pose graph. Figure 2-2 illustrates such a pose graph. Variables represent either a robot pose or a landmark. And factors are either odometry or landmark measurements.

The pose graph SLAM problem can be formulated as:

Problem 1. *A pose graph SLAM problem is the problem of optimizing robot poses* $\mathbf{X}_{0:T}$ *and landmark locations* $\mathbf{L}$*, such that the log likelihood of observed odometry and landmark measurements is maximized*

$$\max_{\mathbf{X}_{0:T},\mathbf{L}} \log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}). \tag{2.16}$$

Notice the factors in the pose graph are nonlinear, therefore equation 2.16 is a nonlinear optimization problem. Approaches such as gradient descent could be used to solve these problems [72]. By utilizing sparsity of graphical models, such approaches could scale to thousands of variables easily. However, with the growth of exploration time and distance traveled by the robot, the number of variables and factors required for real-world SLAM problems will quickly become intractable. Therefore, it is important to reduce the size of the graph to obtain fast optimization on such graphs.

2.3 Summary

This chapter reviewed the background on factor graphs and their application in SLAM problems. Next chapter will formulate the focused mapping problem with the factor graph representation, and develop inference algorithms accordingly.

Chapter 3

Focused Mapping for Minimal-collision Navigation

3.1 Introduction

One of the core enabling capabilities for mobile robots operating in uncertain and GPS-denied environments is simultaneous localization and mapping (SLAM). The built map will subsequently be used to perform tasks. SLAM problems often require high dimensional models to represent robot poses, landmarks, and obstacles. Graphical models are a powerful tool in this case because they can explicitly represent conditional independences between variables resulting in fast inference on large-scale problems [69, 73, 74]. However, when the robot travels longer distance, the variables required to represent robot poses grows and memory demand growths. And when the robot stays longer in the environment, it obtains more and more measurements of the landmarks and the computation demand on processing these measurements grows. Naively applying all of the robot poses and measurements into graphical models can result in an unbounded growth in memory and computational requirements, which will ultimately result in failure to achieve the desired task.

On the other hand, given a task, the required fidelity and choice of map representation are, in general, task specific, and may not even be constant across the task. Consider the following examples:

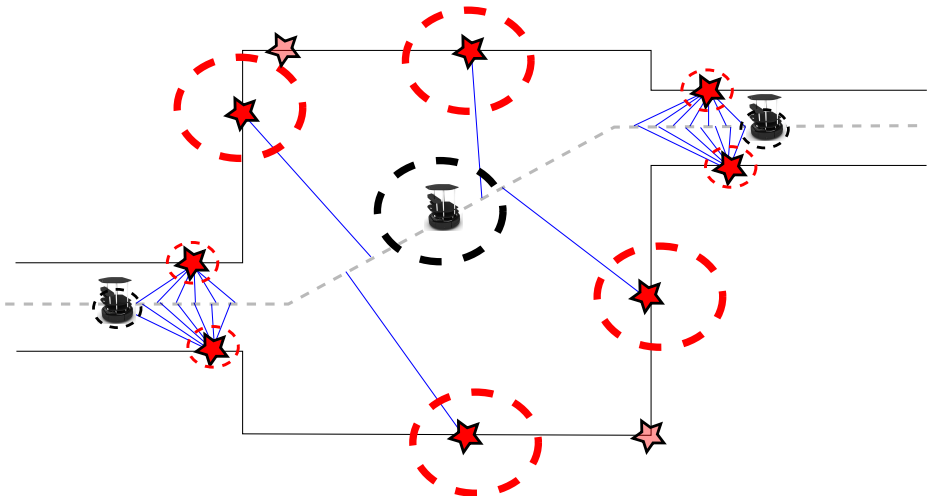


Figure 3-1: Resource-Constrained Navigation – It is more important to localize landmarks in narrow hallways than those in large rooms for the task of collision-free navigation. As a result, along the trajectory of robot (gray dashed line), far fewer measurements (blue) are required for landmarks (red stars) in the open area as compared with the narrow hallway. (Black and red dashed ellipses represent uncertainty in robot position and landmarks position, respectively.)

- A robot is navigating through an unknown environment consisting of both open areas and narrow hallways.
- An autonomous car is operating on a road network consisting of highways and local roads without an accurate map.
- A marine robot is localizing a set of underwater mines amongst clutter.

When robots are operating in such complicated scenarios and have constrained onboard resources, it is important to prioritize what variable and measurements are maintained in the map. As illustrated in Figure 3-1, in the indoor navigation scenario, the space consists of both narrow hallways and open areas. It is more important to have accurate estimates of landmark locations in tight corridors, therefore the robot has lower uncertainty in its poses and less chance of collisions compared with open areas. Similarly on the road network, landmarks on highly traveled roads are more useful [75] and in the underwater scenario it is more important that the robot localizes the mines as opposed to the clutter. Consequently, the robot can save resources, such as memory and computational effort, by focusing the mapping operation to more

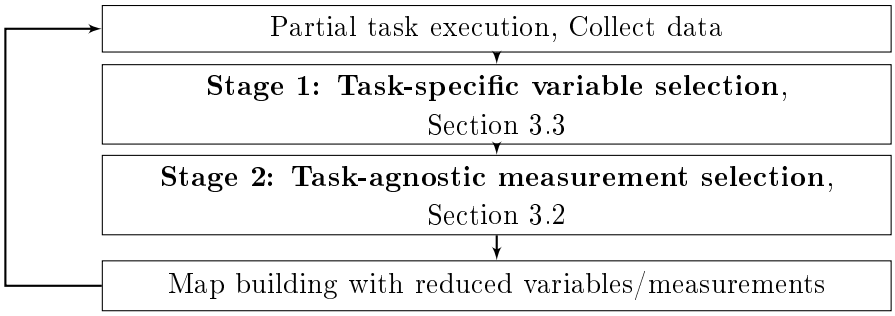


Figure 3-2: Two-stage focused inference

explicitly support the task.

This chapter presents a more flexible mapping framework that supports task-driven prioritization. The framework is applied to the task of robot navigation. The workflow for the general case is shown in Figure 3-2. The robot first navigates around to to gathers some data. The next step is to select the focused variables that are deemed important for subsequent navigation tasks. Section 3.3 discusses the specific case of selecting landmarks to support collision-free navigation (the focused variables). The third step is to select the subset of measurements that are most useful for estimating the focused variables, as described in general terms in Section 3.2. The last step is to build a map using the reduced set of variables and measurements. The robot continues in this loop until the task is finished or its resources are exhausted. In the incremental approach, the robot will obtain new data each time it executes the task. The reduced map from old data is taken as an input to the two-stage variable and measurement reduction algorithm with new collected data to further update the map.

3.2 Measurement Sparsification on Focused Variables

We begin by assuming that the selection of focused variables has already been performed, and then proceed to formulate the problem of measurement selection to support inference over these focused variables. This section leaves the formulation as general as possible with regards to task execution and focused variable selection, which will be discussed in the following sections.

In general, there are two ways to approach sparsifying measurements. The first one is forward selection where the map starts with no measurements and adds measurements when they are deemed important [26]. The other way is backward map reduction where the map starts with an optimized graph with full measurements, then removes measurements when they are deemed redundant [19]. In our resource-constrained scenario, an optimized map over full variables and measurements is not available a priori. Furthermore, the number of measurements to retain is very small compared to the total number of measurements available. If each steps only adds or removes one measurement, then backwards map reduction will involve much more steps of operations than that of forward selection. Therefore, a forward selection approach is used in our resource-constrained setting here.

3.2.1 Problem Formulation

Denote $X = \{X_1, \cdots, X_N\}$ as a set of hidden random variables that are (partially) observable through measurements $z = \{z_1, \cdots, z_K\}$ which are collected in the initial data collection phase, where K can be very large.

Then, in its most general form, the measurement selection problem can be posed as follows:

Problem 2. ***Unfocused Measurement Selection:*** *Select the subset of measurements $z^R = \{z_1^R, \cdots, z_{K^R}^R\} \subset z$, such that some information metric $f(\cdot)$ over the hidden variables X is maximized, subject to some cost function $g(\cdot)$ constraint on the measurement set:*

$$\begin{aligned} &\max_{z^R \subset z} \ f(X; z^R) \\ &s.t. \quad g(z^R) \leq c \end{aligned} \tag{3.1}$$

Commonly used information metrics include entropy, mutual information, and KL divergence. In this thesis, the cost function used is the cardinality of the set $g(z^R) = |z^R|$, in practice, the number of measurements often relates to the model

complexity, thus the computation cost. Therefore the resource budget c could be selected based on computation resource available.

Here, instead of quantifying information gain on all variables in Problem 2, the robot has specific tasks, such as navigation. In such cases, not all variables are equally important. For example, in order to navigate through some environment without collision, landmarks in narrow passages would be more important for the robot to accurately localize itself. Assume we already obtained such a set of important variables, the "focused" variables $\widetilde{X}$ [1], which are a more compact representation of the variables in X: $\widetilde{X} = \{\widetilde{X}_1, \cdots \widetilde{X}_{\widetilde{N}}\}$, with $\widetilde{N} \ll N$. To maintain generality, we represent the mapping from the unfocused variables to the focused variables by a *prioritization function*:

Definition 1. ***Prioritization Function*** *The function $w : \mathcal{R}^N \to \mathcal{R}^{\widetilde{N}}$ that maps the set of unfocused variables onto the set of focused variables.*

For example, in the degenerate case of variable selection, $\widetilde{X} = w(X) = WX$ where W is an $\widetilde{N} \times N$ matrix with a single 1 in each row. However, the formulation allows for more complex mappings from the unfocused to the focused set.

The problem of focused measurement selection consists of choosing the best subset of the full measurement set z to optimally localize the focused variables.

Problem 3. ***Focused Measurement Selection:*** *Select the subset of measurements $z^R = \{z_1^R, \cdots, z_{K^R}^R\} \subset z$, such that some information metric $f(\cdot)$ over the **focused** hidden variables $\widetilde{X}$ is maximized, subject to some cost function $g(\cdot)$ constraint on the measurement set:*

$$\begin{aligned} &\max_{z^R \subset z} \; f(\widetilde{X}; z^R) \\ &s.t. \quad |z^R| \leq c \end{aligned} \tag{3.2}$$

When a factor graph is used to model X, the joint log posterior can be expressed

[1] The $\widetilde{}$ notation is used throughout to refer to the focused set of variables.

as sum of factors $\psi_a(x_{\{a\}})$:

$$p(x|z^R) \propto \exp\left\{\sum_{a\in\mathcal{A}} \phi_a(x_{\{a\}})\right\} \tag{3.3}$$

where $\mathcal{A}$ is the set of all factors, $x_{\{a\}}$ are the variables in factor a.

The reduction from the full set of variables to the focused variables is achieved by mapping the posterior through the prioritization function $\tilde{x} = w(x)$ to produce a new posterior over the focused variables:

$$p(\tilde{x}|z^R) \propto \exp\left\{\sum_{a\in\widetilde{\mathcal{A}}} \tilde{\phi}_a\left(\tilde{x}_{\{a\}}\right)\right\}, \quad \tilde{x} = w(x) \tag{3.4}$$

where $\widetilde{\mathcal{C}}$ is the new (smaller) set of factors over $\tilde{X}$, and $\tilde{\phi}_c$ are the resulting factor functions.

If we define the function $f(\cdot)$ in (3.2) to be the Shannon entropy of the conditional distribution:

$$H(\widetilde{X}|z^R) = \mathbb{E}_{\widetilde{X}|z^R}\left[-\log p(\tilde{x}|z^R)\right] \tag{3.5}$$

where $\mathbb{E}$ is the expectation operator, then we obtain the resulting equation for the entropy of the focused variables, $\widetilde{X}$ conditional on the subset of measurements z^R as:

$$f(\widetilde{X}; z^R) = H(\widetilde{X}|z^R) = \mathbb{E}_{\widetilde{X}|z^R}\left[-\sum_{a\in\widetilde{\mathcal{A}}} \tilde{\phi}_a(\tilde{x}_{\{a\}})\right] + C, \tag{3.6}$$

where C is a constant.

Note that computing the transformation from ϕ to $\tilde{\phi}$ may be hard in general. Furthermore, in general the graph over $\widetilde{X}$ will be much denser than the graph of X, and computing $H(\widetilde{X}|z^R)$ may be computationally expensive. However, it will be shown that $H(\widetilde{X}|z^R)$ can be computed in closed form given two assumptions:

Assumption 2. *The clique potentials can be approximated as Gaussian distributions.*

Assumption 3. *The prioritization function $w(\cdot)$ is an affine transformation.*

Assumption 2 is, in fact, less limiting than the standard additive Gaussian noise assumption in the SLAM literature, since even in the case of robust cost functions we can always approximate the posterior as a Gaussian distribution using the Laplacian approximation [76]. Assumption 3 essentially requires the focused variables to be linear combinations of the original variables, which is still more general than other variable selection methods [33, 34] which restrict the set of focused variables to be a strict subset of the original set. For example, given corners/edges of an object, a focused variable could be the center of object, which is a linear combination of corners/edges.

Section 2.1.3 discusses the point that the probability could be approximated by a Gaussian distribution with information matrix, Λ_{z^R} given by the Hessian:

$$\hat{p}(x|z^R) = \mathcal{N}(\zeta, \Lambda_{z^R}^{-1}), \quad \Lambda_{z^R} = \sum_{a \in \mathcal{A}} -\frac{\partial^2}{\partial x^2} \phi_a(x^*_{\{a\}}), \tag{3.7}$$

One point of note is that Λ_{z^R} is inherently dependent on the linearization point chosen x^* and selected measurements z^R.

3.2.2 Affine Prioritization Function

In Definition 1, we defined the prioritization function $w(\cdot)$, which is a task-specific and predefined function that maps the set of all variables onto the set of focused variables. In this section we impose the restriction that this function is affine in order to provide a closed-form means of getting from (3.3) to (3.4):

$$\widetilde{X} = w(X) = WX, \tag{3.8}$$

where $W \in \mathcal{R}^{\widetilde{N} \times N}$. For example, Figure 2-1 gave an example of a factor graph representing variables $\{X_1, X_2, X_3, X_4, X_5\}$, the following gives an affine prioritization

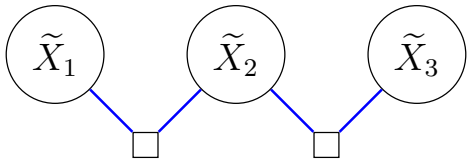


Figure 3-3: Transformed factor graph. New variables $\widetilde{X}_1 = X_2$, $\widetilde{X}_2 = \frac{1}{3}X_3 + \frac{2}{3}X_4$, $\widetilde{X}_3 = X_5$. New factors: blue $\widetilde{\psi}_{12}(\widetilde{X}_1, \widetilde{X}_2)$, green $\widetilde{\psi}_{2,3}(\widetilde{X}_2, \widetilde{X}_3)$.

.

function

$$\begin{bmatrix} \widetilde{X}_1 \\ \widetilde{X}_2 \\ \widetilde{X}_3 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 2/3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{W} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix}. \tag{3.9}$$

Then the new graph for the set of focused variables $\{\widetilde{X}_1, \widetilde{X}_2, \widetilde{X}_3\}$ is illustrated in Figure 3-3.

This restriction on the prioritization to be affine guarantees that the posterior over the focused variables will still be Gaussian: $\hat{p}(\tilde{x}|z^R) = \mathcal{N}(\tilde{\zeta}, \Lambda_{z^R}^{-1})$ [69]. Furthermore, we can easily write an expression for the information matrix:

$$\Lambda_{z^R} = (W\Lambda_{z^R}^{-1}W^T)^{-1}, \tag{3.10}$$

as a result the approximate entropy of the focused variables given the selected measurements can be written in closed form:

$$\hat{H}(\widetilde{X}|z^R) = -\frac{1}{2}\log\left|\widetilde{\Lambda}_{z^R}\right| + C = \frac{1}{2}\log\left|W\Lambda_{z^R}^{-1}W^T\right| + C \tag{3.11}$$

We finish by restating Problem 3 based on the Gaussian approximation and restriction to affine prioritization functions:

Problem 4. ***Approximate Focused Measurement Selection*** *Select the subset of measurements $z^R = \{z_1^R, \cdots, z_{K^R}^R\} \subset z$, such that approximate entropic information over the* ***focused*** *hidden variables $\widetilde{X}$ is maximized subject to the same constraint as*

(3.2):

$$\begin{aligned} \max_{z^R \subset z} \quad & -\log \left| W \Lambda_{z^R}^{-1} W^T \right| \\ s.t. \quad & |z^R| \leq c \end{aligned}$$

3.2.3 Efficient Solution

Note that problem 4 optimize an objective function that is defined on a set. Such a set function is discrete and combinational, and computing the optimal solution is typically intractable on SLAM-size problems. We instead uses a greedy selection procedure that maximizes the incremental information gain on the next measurement.

Each new measurement z_k^R added to the set will introduce a new clique potential, $\phi_k(x_{\{k\}}^*)$, into the joint posterior, where the set of variables $X_{\{k\}}$ are the ones affected by measurement z_k^R. We denote the intermediate set of $k \leq K^R$ measurements that have already been selected as $z^{R_k} = \{z_1^R, \cdots, z_k^R\}$.

The approximate entropy reduction (or information gain) over the focused variables brought about by adding a new measurement z_k^R is:

$$\Delta \hat{H}(\widetilde{X}|z_k^R) = \hat{H}(\widetilde{X}|z^{R_{k-1}}) - \hat{H}(\widetilde{X}|z^{R_k}), \tag{3.12}$$

is the value that we want to maximize. In the following theorem we show that this quantity can be efficiently computed:

Theorem 1. *The approximate reduction in entropy over the focused variables brought about by introducing the new measurement z_k^R will be:*

$$\Delta \hat{H}(\widetilde{X}|z_k^R) = -\frac{1}{2} \log \left| I - (I + J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k)^{-1} L_k^T \widetilde{\Lambda}_{z^{R_{k-1}}} L_k \right| \tag{3.13}$$

where $L_k \triangleq W \Lambda_{z^{R_{k-1}}}^{-1} J_k$, J_k is the measurement covariance weighted stacked Jacobian [23], and W is the affine prioritization function.

Proof. We proceed similarly to [26], but with the added complication that there is a

transformation from the unfocused to the focused variables ($\widetilde{X} = WX$) required to evaluate (3.12).

From (3.7), the information matrix after the introduction of z_k^R will be:

$$\Lambda_{z^{R_k}} = \Lambda_{z^{R_{k-1}}} - \frac{\partial^2 \phi_k(x^*_{\{k\}})}{\partial x^2} = \Lambda_{z^{R_{k-1}}} + J_k J_k^T \tag{3.14}$$

where $J_k \in \mathcal{R}^{N \times |x_{\{k\}}|}$, with $|x_{\{k\}}|$ the number of variables in clique k and only blocks corresponding to the variables in the clique $x_{\{k\}}$ being non-zero. The new entropy after adding measurement z_k^R can be evaluated as

$$\hat{H}(\widetilde{X}|z_k^R) = \frac{1}{2} \log \left| W(\Lambda_{z^{R_k}})^{-1} W^T \right| \tag{3.15}$$

Then the reduction in entropy, $\Delta\hat{H}(\widetilde{X}|z_k^R)$, is

$$\begin{aligned}
\Delta\hat{H}(\widetilde{X}|z_k^R) &= -\frac{1}{2} \log \left| W \Lambda^{-1}_{z^{R_k}} W^T \right| + \frac{1}{2} \log \left| W \Lambda^{-1}_{z^{R_{k-1}}} W^T \right| \\
&= -\frac{1}{2} \log \left| W(\Lambda_{Z^R_{k-1}} + J_k J_k^T)^{-1} W^T \right| + \frac{1}{2} \log \left| W(\Lambda_{Z^R_{k-1}})^{-1} W^T \right| \\
&= -\frac{1}{2} \log \frac{\left| W(\Lambda_{z^{R_{k-1}}} + J_k J_k^T)^{-1} W^T \right|}{\left| \left(W \Lambda^{-1}_{z^{R_{k-1}}} W^T \right) \right|}.
\end{aligned}$$

By applying the matrix inversion lemma:

$$(A + BB^T)^{-1} = A^{-1} - A^{-1}B(I + B^T A^{-1} B)^{-1} B^T A^{-1},$$

and the determinant property $|I + AB| = |I + BA|$, (3.16) reduces to:

$$-\frac{1}{2} \log \frac{\left| W \left(\Lambda^{-1}_{z^{R_{k-1}}} - \Lambda^{-1}_{z^{R_{k-1}}} J_k (I + J_k^T \Lambda^{-1}_{z^{R_{k-1}}} J_k)^{-1} J_k^T \Lambda^{-1}_{z^{R_{k-1}}} \right) W^T \right|}{\left| \left(W \Lambda^{-1}_{z^{R_{k-1}}} W^T \right) \right|} \tag{3.16}$$

$$= -\frac{1}{2} \log \frac{\left| W \Lambda^{-1}_{z^{R_{k-1}}} W^T - L_k (I + J_k^T \Lambda^{-1}_{z^{R_{k-1}}} J_k)^{-1} L_k^T \right|}{\left| \left(W \Lambda^{-1}_{z^{R_{k-1}}} W^T \right) \right|} \tag{3.17}$$

$$= -\frac{1}{2} \log \left| I - (W \Lambda^{-1}_{z^{R_{k-1}}} W^T)^{-1} L_k (I + J_k^T \Lambda^{-1}_{z^{R_{k-1}}} J_k)^{-1} L_k^T \right| \tag{3.18}$$

Algorithm 3-1 Measurement Selection for Prioritized Landmarks

Input: Initial information matrix $\Lambda_{z^{R_0}}$, focused variables $\widetilde{X}$, all measurements z, budget c

Output: measurements z^R

1: $k \leftarrow 0$, $z^{R_0} \leftarrow \emptyset$
2: **while** $|z^{R_k}| < c$ **do**
3: $\quad k \leftarrow k + 1$
4: $\quad z_k^* = \underset{z_k^R \in z \setminus z^{R_{k-1}}}{\text{argmax}} \ \Delta\hat{H}(\widetilde{X}|z_k^R)$
5: $\quad z^{R_k} = z^{R_{k-1}} \cup \{z_k^*\}$
6: $\quad \Lambda_{z^{R_k}} = \Lambda_{z^{R_{k-1}}} + J_k J_k^T$
7: $\quad \widetilde{\Lambda}_{z^{R_k}} = (W\Lambda_{z^{R_k}}^{-1} W^T)^{-1}$
8: **end while**

$$
\begin{aligned}
&= -\frac{1}{2}\log\left|I - \tilde{\Lambda}_{z^{R_{k-1}}}^{-1} L_k (I + J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k)^{-1} L_k^T\right| \\
&= -\frac{1}{2}\log\left|I - (I + J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k)^{-1} L_k^T \tilde{\Lambda}_{z^{R_{k-1}}} L_k\right|,
\end{aligned} \tag{3.19}
$$

which is the required result. □

Computational Complexity of (3.13)

Similar to [26], we have avoided the need to compute the entropy over the entire variable set. However, unlike in [26] where the calculation of information gain scales only with the size of the measurements, we have a slightly more complicated scenario because of the prioritization transformation W. Upon further inspection of (3.13), calculating $J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k$ has a computational complexity of $\mathcal{O}(|x_{\{k\}}|^4)$ (similar to [26]). In addition, note that calculating the ith element of L_k (which is computed as $L_k^i = W^i \Lambda_{z^{R_{k-1}}}^{-1} J_k$, where W^i is ith row of W) requires us to check the submatrix of $\Lambda_{z^{R_{k-1}}}^{-1}$ that corresponds to the non-zero elements of W^i *and* J_k. Therefore, the overall complexity of computing L_k is $\mathcal{O}(||W||_0 |x_{\{k\}}| + |x_{\{k\}}|^4)$, where $||W||_0$ is the number of non-zero elements in the prioritization transformation. Typically the clique size, $|x_{\{k\}}| \ll N$, and since $\widetilde{N} \ll N$ we should expect that $||W||_0 \ll N$, therefore the overall complexity of computing L_k is much less than the problem size N.

Algorithm 3-1 summarizes the measurement selection approach.

3.3 Variable Sparsification for Navigation

In this section, we return to the question of how to select the focused variables that are an input to the measurement selection scheme described in the previous section. The selection of focused variables mainly depends on the robot's tasks and goals. Here we specifically consider the task of collision-free autonomous navigation.

Assume the robot has gathered some data with image and depth/laser sensors. The dataset contains both the robot's trajectory as well as measurements of landmarks. The robot needs to track a set of landmarks that can be used to localize itself and navigate through the environment. Denote the robot's trajectory as a sequence of random variables $X = \{X_1, \cdots, X_T\}$. In GPS-denied environments, X is not directly observable. However, the robot can always measure the incremental change between two sequential poses (odometry), for example from an IMU or wheel encoder.

There also exists a set of landmarks from which focused landmarks can be selected. Denote the set of landmarks as $L = \{L_1, L_2, \cdots L_N\}$.

3.3.1 Selection of Focused Landmarks

While it could be possible to have thousands of landmarks, often a small, carefully chosen subset can lead to sufficiently accurate navigation. In particular, for resource-constrained systems, reducing the number of landmarks will significantly reduce the computation required for data association, which will in turn enable faster and more efficient on-line trajectory planning and navigation.

Narrow passages are challenging for collision-free autonomous robot motion planning. However, in the case of high uncertainty in the map and robot position, the "narrowness" should be redefined. As shown in Figure 3-4 a "geometrically" wide passage might still be problematic for a robot that does not have access to accurate landmark information and thus has poor localization accuracy. We refer to this passage as being "geometrically wide" but "probabilistically narrow" and we will formalize these terms below.

The evaluation of probabilistic narrowness involves two key components: 1) Cal-

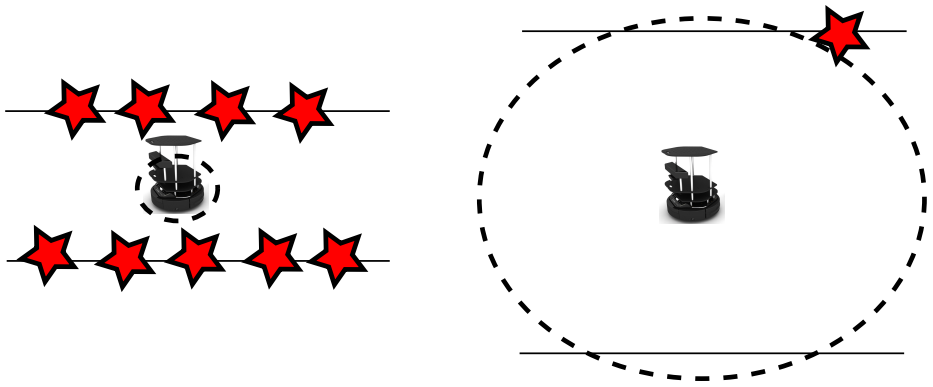


Figure 3-4: Probabilistic narrowness is associated with the probability of collision while navigating the environment. **Left**: This geometrically narrow corridor is probabilistically wide because there are many features with which to localize. **Right**: Conversely, this geometrically wide hallway is probabilistically narrow.

culating an estimate of the robot's position uncertainty and 2) Calculating the probability of collision based on the robot's uncertainty and the distance to obstacles as determined by the robot.

Robot position uncertainty

To generate an estimate of the robot's position uncertainty at any given point x along the robot's path X, we use the concept of belief stabilization introduced in [41].

Assume we have a closed-loop controller that can stabilize the robot's state to belief state x. Such a controller is typically comprised of an estimator and a separated controller. The estimator generates an *a posteriori* distribution over all robot poses based on the existing map of landmarks and the local observation of landmarks. Given these estimates, the separated controller will generate a control signal that drives the robot toward x. To design an analytic measure of narrowness we rely on a simple Linear-Quadratic-Gaussian (LQG) controller, which combines a Kalman filter and a linear quadratic regulator. It can be shown [41] that starting from any $\Sigma_0 > \Sigma^*(x)$, estimation covariance decreases monotonically and approaches the steady state covariance $\Sigma^*(x)$, which is the fixed point of a Riccati recursion at location x:

$$\{\Sigma^* = Q + A(\Sigma^* - \Sigma^* H^T (H\Sigma^* H^T + R)^{-1} H\Sigma^*) A^T\}_x \tag{3.20}$$

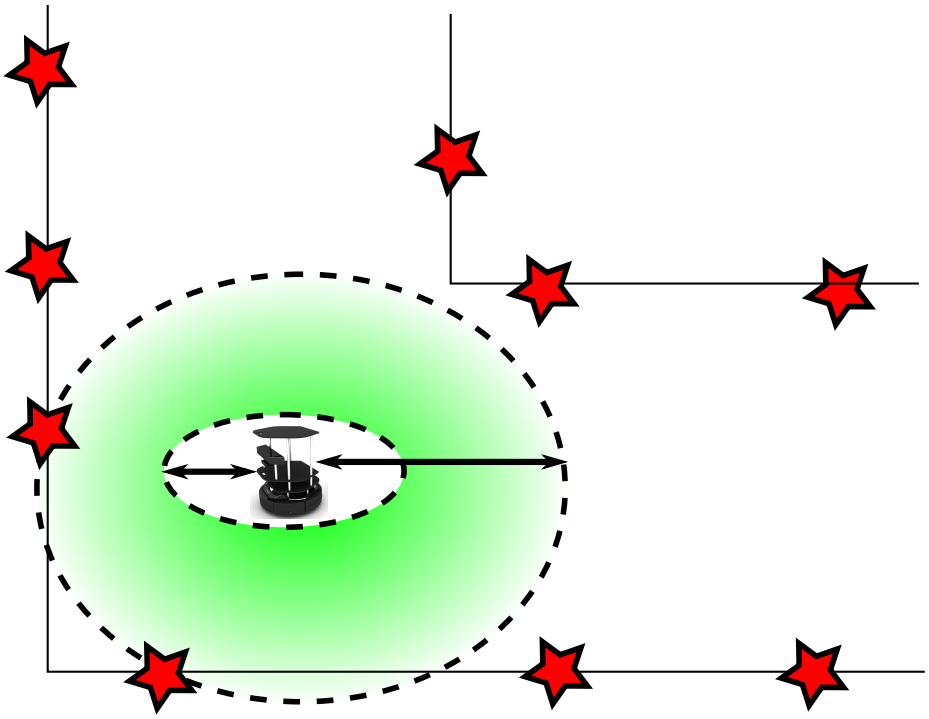


Figure 3-5: Stabilization as Lower Bound on Collision: Based on the landmark estimates, we solve a Riccati recursion to determine the minimum robot uncertainty at each point in the free space. This value is used as the measure of "probabilistic narrowness" introduced in Figure 3-4.

The Jacobians, A and H are computed by linearizing the measurement model at point x on the path, where Q and R are the process and measurement noise, respectively. A visual depiction of the stabilization process is shown in Figure 3-5. We associate with each pose x a set of visible landmarks $L(x)$ (the ones connected to the robot by blue lines in Figure 3-5). The value of the steady-state covariance will be dependent on the set of visible landmarks as expressed through the measurement noise covariance R. Note that Jacobian matrices and set of visible landmarks would be different for different points on the path. The main computational advantage in using this measure is that $\Sigma^*(x)$ only depends on x not the path that leads to x.

Collision probability

Using $\mathcal{N}(x_t, \Sigma^*(x_t))$ as a measure of uncertainty for each point x_t on the path, the collision probability can be defined by a Monte Carlo method. For each x_t sample the normal distribution, denote I as the set samples drawn from $\mathcal{N}(x_t, \Sigma^*(x_t))$. Denote

O_{bs} as the set of obstacles, and $I \cap O_{bs}$ as the set of samples that collide with obstacles. Then the collision probability can be defined as:

$$P_c(x_t) = \lim_{|I| \to \infty} \frac{|I \cap O_{bs}|}{|I|} \tag{3.21}$$

Monte Carlo methods are typically expensive and not suitable for resource-constrained scenarios. There have been many approaches to approximate it, such as [28, 77, 78]. While all of these methods can be used, in this chapter we utilize an approximate measure that is computationally cheaper. First, denote the set of readings received from the laser range finder in its local frame as $D = \{d^i\}$ where each d^i is a 2D vector connecting x_t to a point on the obstacle surface. Then, we simply compute the minimum Mahalanobis distance between x_t and the obstacle surface as

$$P_c(x_t) = \min_{d^i \in \bar{D}} \{(d^i)^T \Omega_t d^i\} \tag{3.22}$$

where $\Omega_t = (\Sigma^*(x_t))^{-1}$ is the information matrix corresponding to x_t. $\bar{D}$ could be equal to D or could be a sub-sampled version of D based on the available computational resources. In the extreme case, $\bar{D}$ could be a single point $\bar{D} = x^{obst} = \text{argmin}_{d^i \in D} \|d^i\|$ that is the closest point on the obstacle surface.

In other words, P_c measures how many standard deviations obstacles are away from the mean of the localization distribution.

Notice the approximation retains the ability to favor some unbalanced covariance choices. For example, when one eigenvalue is significantly larger than the other, resulting in a "collision" in the direction of the larger one, the approximation will still have a high cost along that direction.

The landmark selection problem is framed as finding the poses along the trajectory with the highest collision probability then selecting landmarks that can reduce this pose's uncertainty.

Algorithm 3-2 Minimum Collision Probability Landmark Selection

Input: Robot poses X, landmarks L, odometry noise Q, measurement noise R, budget α

Output: Selected landmarks L_k^f

$k \leftarrow 0$, $L_0^f \leftarrow \emptyset$
Compute $P_c(x_t)$ for all $x_t \in X$ by (3.20) and (3.22)
$P(l) = \max_{x_t \text{ s.t. } l \in L(x_t)} P_c(x_t)$ for all $l \in L$
while $k < \alpha$ **do**
    $k \leftarrow k + 1$
    $l_k^* \leftarrow \operatorname{argmax}_{l \in L \setminus L_{k-1}^f} P(l)$
    $L_k^f = L_{k-1}^f \cup \{l_k^*\}$
    update $P_c(x_t)$ for x_t that can observe l_k^*
    update any $P(l)$ that may have changed
end while

Problem 5. ***Minimum Collision Probability Landmark Selection:*** *Select the α landmarks $L_f \subset L$, such that the worst case probability of collision is minimized:*

$$\begin{aligned} \min_{L_f \subset L} \ & \max_{x_t \in X} P_c(x_t) \\ s.t. \ & |L_f| \leq \alpha \end{aligned} \tag{3.23}$$

The problem is solved by greedily selecting landmarks as summarized in Alg. 3-2. At each iteration, pick the landmark l^* that is associated with maximal $P_c(x_t)$, then update $P_c(x_t)$ for all the x that can observe l^*. Since the number of poses that can observe any individual landmark is low, this greedy approximation can be computed efficiently.

3.3.2 Focused Measurement Selection for Navigation

Here we detail how to apply this landmark selection scheme resulting from solving Problem 5 to the measurement selection process described in Sec. 3.2. Define random

variables $\mathbf{X} = \{X_0, \cdots X_T\}$ to represent robot poses and $\mathbf{L} = \{L_1, \cdots, L_M\}$ to represent landmarks. Recall that the joint log probability of a factor graph is proportional to the sum of odometry factors $\phi_a(o_a; X_a)$ and landmark factors $\phi_a(z_a; X_a, L_a)$:

$$p(\mathbf{X}, \mathbf{L}) \propto \exp\left(\sum_{a \in \mathcal{A}} \phi_a(o_a; X_a) + \sum_{a \in \mathcal{A}} \phi_a(z_a; X_a, L_a)\right). \tag{3.24}$$

An odometry, o_a, adds a factor between two subsequent poses $\phi_t(o_t; X_t, X_{t-1})$ and a landmark measurement, z_t^k, adds a factor between a pose and a landmark $\phi_t^k(z_t^k; X_t, L_k)$.

The focused variables are selected focused landmarks: $\tilde{X} = L_f \subset L$. Denote the rest of the unfocused landmarks and robot poses as $\tilde{X}' = \{L \setminus L_f, X\}$. Then the affine prioritization function represented by W is

$$\tilde{X} = W \begin{bmatrix} L_f \\ \tilde{X}' \end{bmatrix}, \quad W = \begin{bmatrix} I_{\tilde{N} \times \tilde{N}} & \mathbf{0}_{\tilde{N} \times N - \tilde{N}} \end{bmatrix}. \tag{3.25}$$

Using the standard assumption that the odometry and landmark measurement are corrupted by additive Gaussian noise [20], then the factors have quadratic forms as show in (2.10) and (2.12). The W matrix and the factors can now be used in Algorithm 3-1 for the second stage (focused measurement selection).

Notice that the Gaussian approximation of the joint likelihood (3.7) is subject to a linearization point. An ideal linearization point would be the maximum likelihood value where the gradient is 0. However, computing this optimal value would involve optimizing the full graph with all variables and edges, which would become slow and resource intensive quickly when the problem gets bigger. Instead we use the odometry based initial values as the linearization point. While it may be subject to performance loss, it significantly reduces the computation burden, which is important for resource constrained robots.

For this choice of W, (3.13) can be further simplified. First decompose $\Lambda_{z^R}^{-1}$ into

blocks of focused and unfocused variables:

$$\Lambda_{z^R}^{-1} = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \tag{3.26}$$

with $A = A^T > 0$ and $C = C^T > 0$. Then

$$L_k^T \tilde{\Lambda}_{z^{R_{k-1}}} L_k = J_k^T \Lambda_{z^{R_{k-1}}}^{-1} W^T \left(W \Lambda_{z^{R_{k-1}}}^{-1} W^T \right)^{-1} W \Lambda_{z^{R_{k-1}}}^{-1} J_k \tag{3.27}$$

Inserting (3.26) into (3.25) gives:

$$\left(W \Lambda_{z^{R_{k-1}}}^{-1} W^T \right)^{-1} = A^{-1} \tag{3.28}$$

and

$$\begin{aligned} L_k^T \tilde{\Lambda}_{z^{R_{k-1}}} L_k =& J_k^T \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} I \\ 0 \end{bmatrix} A^{-1} \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} J_k \\ =& J_k^T \begin{bmatrix} A & B \\ B^T & B^T A^{-1} B \end{bmatrix} J_k \\ =& J_k^T \left(\Lambda_{z^{R_{k-1}}}^{-1} - \begin{bmatrix} 0 & 0 \\ 0 & C - B^T A^{-1} B \end{bmatrix} \right) J_k \\ =& J_k^T \left(\Lambda_{z^{R_{k-1}}}^{-1} - \begin{bmatrix} 0 & 0 \\ 0 & (\Lambda_{z^{R_{k-1}}}^{C})^{-1} \end{bmatrix} \right) J_k \end{aligned} \tag{3.29}$$

where $\Lambda_{z^{R_{k-1}}}^{C}$ denotes the sub-matrix of $\Lambda_{z^{R_{k-1}}}$ corresponding to the unfocused variables and $(\Lambda_{z^{R_{k-1}}}^{C})^{-1}$ is the marginal covariance matrix of unfocused variables. Inserting (3.29) into (3.13) yields:

$$\begin{aligned} \Delta \hat{H}(\tilde{X} | z_k^R) &= -\frac{1}{2} \log \left| I - (I + J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k)^{-1} L_k^T \tilde{\Lambda}_{z^{R_{k-1}}} L_k \right| \\ &= \frac{1}{2} \log \left| I + J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k \right| - \frac{1}{2} \log \left| I + J_k^T \Lambda_{z^{R_{k-1}}}^{-1} J_k - L_k^T \tilde{\Lambda}_{z^{R_{k-1}}} L_k \right| \end{aligned}$$

$$= \frac{1}{2}\log\left|I + J_k^T \Lambda^{-1}_{z^{R_{k-1}}} J_k\right| - \frac{1}{2}\log\left|I + J_k^T \begin{bmatrix} 0 & 0 \\ 0 & (\Lambda^C_{z^{R_{k-1}}})^{-1} \end{bmatrix} J_k\right| \quad (3.30)$$

Note that the form of (3.30) is intuitive. The first term is the information gain on all variables, while the second term has a similar form, but is computed only on unfocused variables. The difference of these two terms is the information gain on focused variables. Depending on whether the new measurement is of a focused landmark or unfocused landmark, (3.30) can be further simplified, as outlined in the following.

Case 1: Measurement of focused landmark, $l_i \in L_f$ In this case, J_k is non-zero only at the ith row corresponding to the focused landmark and the jth row corresponding to the robot pose, $J_k = [\cdots, J_k, \cdots, J_j, \cdots]^T_k$. Denote Λ^{-1}_{ij} as the element corresponding to i, j location of matrix $\Lambda^{-1}_{z^{R_{k-1}}}$, and $(\Lambda^C_{ij})^{-1}$ as the element corresponding to i, j location of matrix $(\Lambda^C_{z^{R_{k-1}}})^{-1}$, then (3.30) can be further simplified to:

$$\Delta\hat{H}(\tilde{X}|z^R_k) = \frac{1}{2}\log\left|I + \begin{bmatrix} J_i \\ J_j \end{bmatrix}^T \begin{bmatrix} \Lambda^{-1}_{ii} & \Lambda^{-1}_{ij} \\ \Lambda^{-1}_{ji} & \Lambda^{-1}_{jj} \end{bmatrix} \begin{bmatrix} J_i \\ J_j \end{bmatrix}\right| - \frac{1}{2}\log\left|I + J_j^T (\Lambda^C_{jj})^{-1} J_j\right| \quad (3.31)$$

Case 2: Measurement of unfocused landmark, $l_i \notin L_f$

$$\begin{aligned}\Delta\hat{H}(\tilde{X}|z^R_k) = &\frac{1}{2}\log\left|I + \begin{bmatrix} J_i \\ J_j \end{bmatrix}^T \begin{bmatrix} \Lambda^{-1}_{ii} & \Lambda^{-1}_{ij} \\ \Lambda^{-1}_{ji} & \Lambda^{-1}_{jj} \end{bmatrix} \begin{bmatrix} J_i \\ J_j \end{bmatrix}\right| \\ &- \frac{1}{2}\log\left|I + \begin{bmatrix} J_i \\ J_j \end{bmatrix}^T \begin{bmatrix} (\Lambda^C_{ii})^{-1} & (\Lambda^C_{ij})^{-1} \\ (\Lambda^C_{ji})^{-1} & (\Lambda^C_{jj})^{-1} \end{bmatrix} \begin{bmatrix} J_i \\ J_j \end{bmatrix}\right| \end{aligned} \quad (3.32)$$

Note that the first terms of (3.32) and (3.31) are identical. However, only J_j contributes to the second term in (3.31) but both J_i and J_j contribute to the second term in (3.32). Case 2 has larger information gain over the unfocused variables as compared to Case 1, and consequently, the change in total information is reduced.

3.3.3 Pose Graph Sparsification

The variables in the graph consists of both landmarks and robot poses. After marking certain landmarks as focused variables and selecting measurements, the graph structure is significantly sparsified. However, the graph size is still big, as all the variables including both landmarks and robot poses are still maintained in the graph, which will exceed the robot memory constraint quickly.

Observe that after c measurements are selected, there are at most c robot poses that are connected to any landmarks in the graphical model. Most robot poses are connected to its previous pose and subsequent pose. Marginalizing out such a robot pose does not suffer any information loss [79]. If an unfocused landmark is not connected to any robot poses, marginalizing out it does not introduce information loss, either.

After these two marginalization, the number of variables in the graph is linear in the measurement constraint, which is given as an input as a memory and computation constraint.

3.4 Navigation with Uncertain Landmarks

Different from classical path planning algorithms with known map of landmarks for localization, the environment map here is given as a set of stochastic landmarks $L \sim \mathcal{N}(\widehat{L}, R^L)$ estimated from SLAM. In this section, we discuss how to use such a map to navigate. In particular, an LQG controller is designed that tracks a trajectory connecting the robot's initial point to its goal. First, the trajectory is discretized to T steps as $(x_i^d)_{i=0}^T$ and $(u_i^d)_{i=0}^{T-1}$ where, x_k^d and u_k^d denote the desired state and control signal at the k-th time step on the trajectory. Assume that the robot starts at $x_0 \sim \mathcal{N}(\widehat{x}_0, \Sigma_0)$.

We formulate the problem by incorporating a stochastic map into the LQG framework. Assume that the locations of the landmarks are random variables with a mean and covariance given by the map data structure.

The nonlinear partially-observable state-space equations of the system are as fol-

lows:

$$x_{k+1} = f(x_k, u_k, w_k), \quad w_k \sim \mathcal{N}(0, Q_k) \tag{3.33a}$$

$$z_k = h(x_k, L, v_k), \quad v_k \sim \mathcal{N}(0, R_k), \quad L \sim \mathcal{N}(\widehat{L}, R^L) \tag{3.33b}$$

Represent the control problem as:

$$\begin{aligned}
&\min_{\mu_t(\cdot)} \ \mathbb{E}\left[\sum_{t=0}^{N}(x_t - x_t^d)^T W_x (x_t - x_t^d) + (u_t - u_t^d)^T W_u (u_t - u_t^d)\right] \\
&\quad x_{k+1} = f(x_k, u_k, w_k), \quad w_k \sim \mathcal{N}(0, Q_k) \\
&\quad z_k = h(x_k, L, v_k), \quad v_k \sim \mathcal{N}(0, R_k), \quad L \sim \mathcal{N}(\widehat{L}, R^L) \\
&\quad u_t = \mu_t(b_t) \\
&\quad b_t = p(x_t | z_{0:t}, u_{0:t-1})
\end{aligned} \tag{3.34}$$

To transform the problem in (3.34) into an LQG problem, we first compute a time-varying linear system by linearizing the nonlinear system about the nominal trajectory $(x_k^d, u_k^d)_{k\geq 0}$:

$$\begin{aligned}
x_{k+1} &= f(x_k^d, u_k^d, 0) + A_k(x_k - x_k^d) + B_k(u_k - u_k^d) + G_k w_k, \\
&\quad w_k \sim \mathcal{N}(0, Q_k)
\end{aligned} \tag{3.35a}$$

$$\begin{aligned}
z_k &= h(x_k^d, \widehat{L}, 0) + H_k(x_k - x_k^d) + M_k^L(L - \widehat{L}) + M_k v_k, \\
&\quad v_k \sim \mathcal{N}(0, R_k), \ L \sim \mathcal{N}(\widehat{L}, R^L)
\end{aligned} \tag{3.35b}$$

where

$$A_k = \frac{\partial f}{\partial x}(x_k^d, u_k^d, 0), \quad B_k = \frac{\partial f}{\partial u}(x_k^d, u_k^d, 0), \tag{3.36a}$$

$$G_k = \frac{\partial f}{\partial w}(x_k^d, u_k^d, 0), \quad H_k = \frac{\partial h}{\partial x}(x_k^d, \widehat{L}, 0), \tag{3.36b}$$

$$M_k = \frac{\partial h}{\partial v}(x_k^d, \widehat{L}, 0), \quad M_k^L = \frac{\partial h}{\partial v^L}(x_k^d, \widehat{L}, 0) \tag{3.36c}$$

Now define the following errors:

- LQG error (main error): $e_k = x_k - x_k^d$
- Map error: $v^L = L - \widehat{L}$
- Kalman filter error (KF estimation error): $\widetilde{e}_k = x_k - \widehat{x}_k^+$
- LQR error (estimation of LQG error): $\widehat{e}_k^+ = \widehat{x}_k^+ - x_k^d$

where $\widehat{x}_k^+$ refers to the mean of estimated state at the k-th time step. Let $\delta u_k = u_k - u_k^d$ and $\delta z_k = z_k - z_k^d := z_k - h(x_k^d, \widehat{L}, 0)$, then the linearized models in (3.35) can be rewritten as:

$$\begin{aligned} e_{k+1} &= A_k e_k + B_k \delta u_k + G_k w_k, \\ & \quad w_k \sim \mathcal{N}(0, Q_k) \qquad (3.37a) \\ \delta z_k &= H_k e_k + M_k^L v^L + M_k v_k, \\ & \quad v_k \sim \mathcal{N}(0, R_k),\ v^L \sim \mathcal{N}(0, R^L) \qquad (3.37b) \end{aligned}$$

Defining $\bar{M}_k = [M_k^L \ \ M_k]$ and $\bar{v}_k = [v^{L^T} \ \ v_k^T]^T$ and $\bar{R}_k = \text{diag}[R^L, R_k]$, the observation equation can be written as:

$$\delta z_t = H_k e_k + \bar{M}_k \bar{v}_k, \quad \bar{v}_k \sim \mathcal{N}(0, \bar{R}_k) \qquad (3.38)$$

The last step is to write the control problem in the linear error space as:

$$\begin{aligned} \min_{\mu_t(\cdot)} \ & J = \mathbb{E}\left[\sum_{t=0}^{N} e_t^T W_x e_t + \delta u_t^T W_u \delta u_t\right] \\ & e_{k+1} = A_k e_k + B_k \delta u_k + G_k w_k, \quad w_k \sim \mathcal{N}(0, Q_k) \\ & \delta z_t = H_k e_k + \bar{M}_k \bar{v}_k, \quad \bar{v}_k \sim \mathcal{N}(0, \bar{R}_k) \\ & \delta u_t = \mu_t(b_t) - u_t^d \\ & b_t = p(e_t + x_t^d | \delta z_{0:t}, \delta u_{0:t-1}) \qquad (3.39) \end{aligned}$$

Now, the problem in (3.39) is in the standard LQG form. Note that $e_k = \widetilde{e}_k + \widehat{e}_k^+$ and, based on the separation principle [80], it can be shown that minimizing the quadratic

objective in (3.39) can be divided into two separate minimizations over the estimation error $\widehat{e}_k^+$ and the separated controller error $\widetilde{e}_k$. In the following, we discuss how a KF and an LQR can be designed for this linearized system and finally combine them to construct a time-varying LQG controller.

Kalman Filter: In Kalman filtering, we aim to provide an estimate of the system's state based on the available partial information we have obtained until time k, i.e., $z_{0:k}$. The error estimate is a random vector denoted by e_k^+, whose distribution is the conditional distribution of the state on the obtained observations so far, which is called belief and is denoted by b_k:

$$b_k = p(x_k^+) = p(x_k|z_{0:k}) = \mathcal{N}(\widehat{e}_k^+ + x_k^d, P_k) \tag{3.40}$$

$$\widehat{e}_k^+ = \mathbb{E}[e_k|\delta z_{0:k}, \delta u_{0:k-1}] \tag{3.41}$$

$$P_k = \mathbb{C}[e_k|\delta z_{0:k}, \delta u_{0:k-1}] \tag{3.42}$$

where $\mathbb{E}[\cdot|\cdot]$ and $\mathbb{C}[\cdot|\cdot]$ are the conditional expectation and conditional covariance operators, respectively.

Kalman filtering consists of two steps at every time stage: prediction step and update step. In the prediction step, the mean and covariance of prior e_k^- is computed. For the system in Eq. (3.37) prediction step is:

$$\widehat{e}_{k+1}^- = A_k\widehat{e}_k^+ + B_k\delta u_k \tag{3.43}$$

$$P_{k+1}^- = A_k P_k^+ A_k^T + G_k Q_k G_k^T \tag{3.44}$$

In the update step, the mean and covariance of posterior e_k^+ is computed. For the system in Eq. (3.37), the update step is:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + \bar{M}_k \bar{R}_k \bar{M}_k^T)^{-1} \tag{3.45}$$

$$\widehat{e}_{k+1}^+ = \widehat{e}_{k+1}^- + K_{k+1}(\delta z_{k+1} - H_{k+1}\widehat{e}_{k+1}^-) \tag{3.46}$$

$$P_{k+1}^+ = (I - K_{k+1}H_{k+1})P_{k+1}^- \tag{3.47}$$

LQR controller: Once we obtain the belief from the filter, a controller can generate an optimal control signal accordingly. In other words, we have a time-varying mapping μ_k from the belief space into the control space that generates an optimal control based on the given belief $u_k = \mu_k(b_k)$ at every time step k. LQR controller is of this kind and it is optimal in the sense of minimizing following cost:

$$J_{LQR} = \mathbb{E}\left[\sum_{k\geq 0}(\widehat{e}_k^+)^T W_x(\widehat{e}_k^+) + (\delta u_k)^T W_u(\delta u_k)\right]$$

The linear control law that minimizes this cost function for a linear system is

$$\delta u_k = -F_k \widehat{e}_k^+ \tag{3.48}$$

where the time-varying feedback gains F_k can be computed recursively as follows:

$$F_k = (B_k^T S_{k+1} B_k + W_u)^{-1} B_k^T S_{k+1} A_k \tag{3.49}$$

$$S_k = W_x + A_k^T S_{k+1} A_k - A_k^T S_{k+1} B_k F_k \tag{3.50}$$

If the nominal path is of length N, then the $S_N = W_x$ is the initial condition of above recursion, which is solved backwards in time. Note that the final controller is:

$$\begin{aligned} u_k &= u_k^d + \delta u_k \\ &= u_k^d - (B_k^T S_{k+1} B_k + W_u)^{-1} B_k^T S_{k+1} A_k \widehat{e}_k^+ \end{aligned} \tag{3.51}$$

It should be especially noticed that when stochastic landmarks are incorporated in the navigation mechanism, we can predict the navigation accuracy based on the map accuracy. This information is utilized in the map generation phase. In other words, because we especially optimized the uncertainties on key landmarks in the map, the robot will have high confidence that it can traverse narrow passages and successfully reach the goal v_T.

3.5 Incremental Mapping

True mobile robot autonomy requires that the robot operate over long periods of time and potentially over large spatial scales. In such cases, batch operation on a big dataset would be extremely slow or exceed robot's resources constraints. In such cases, it is particularly important to conduct both variable and measurement reduction in incremental manner over smaller datasets. With variable reduction, the robot can maintain a small model thus saving memory. And with measurement reduction, the robot can maintain a sparse graphical model thus save computation and do fast inference.

The robot will proceed as follows: after some operation in the environment, it has collected some data. First, the two stage landmark and measurement selection procedure presented in Sections 3.2 and 3.3 are conducted to reduce the new collected dataset. Second, the reduced dataset will be used to update the partial map constructed for the environment the robot is operating in. Finally, the robot can navigate with the updated partial map to new defined goal locations, such as new frontiers, or new places given by users. During navigation, the robot gathers new data, which enables the robot to incrementally interleave the mapping/exploration operation with the measurement/landmark reduction and stay within resource budgets. The overall algorithm flow is illustrated in Fig. 3-2.

At some time point during operation, assume the robot has finished $t-1$ operations of two-stage selection. Denote the graph to $t-1$ steps of operation as G_{t-1}, the data the robot gathers during operation t as z^t, and the landmarks observed during operation t as L^t, then the two-stage problem becomes:

Problem 6. ***A. Incremental Focused Measurement Selection:*** *Given graph G_{t-1}, select a minimal set of measurements z^{R_t} from new obtained measurements z^t during operation t, such that information metric $f(\cdot)$ over the* ***focused*** *hidden variables $\tilde{X}^t$ is bounded:*

$$\max_{z^{R_t} \subset z^t} \; g(z^{R_t})$$

$$s.t. \quad f(\tilde{X}^t; z^{R_t}|G_{t-1}) \geq c^t \tag{3.52}$$

B. Incremental Landmark Selection*: Given graph G_{t-1}, select a minimal set of landmarks L_f^t from landmarks L^t observed during operation t, such that the maximum probability of collision is bounded:*

$$\begin{aligned} &\min_{L_f^t \subset L^t} \ |L_f^t| \\ &\quad s.t. \quad \max_{x_k} P_c(x_k|G_{t-1}) \leq \alpha^t \end{aligned} \tag{3.53}$$

Note that Problem 6 differs from Problems 3 and 5 in two aspects. First, the new metrics in (3.52) and (3.53) are conditioned on the reduced graphical model from the previous operation G_{t-1}. Secondly, Problem 6 enforces the collision probability and information gain as constraints, instead of optimizing information gain and collision probability given a landmark and measurement budget. Notice in a single batch setting, the robot has access to all of the data, which represents the entire environment. In an incremental setting, the robot only has access to a single subset of the data at a given time. Each subset represents part of the environment and different subsets may correspond to different parts of the environments. For example, one subset represents narrow spaces while others corresponds to more open space. If the incremental problem is set up as as Problems 3 and 5, then a uniform resource budget is enforced for all different data subsets. As a result, it will lead to wasted resources on open spaces and low accuracy on narrow places. Therefore, Problem 6 enforces collision probability and information gain as constraints, then greedily selects landmarks and measurements until the constraints are satisfied. The incremental algorithm is summarized in Algorithm 3-3.

Algorithm 3-3 Incremental Focused Mapping

Input: Initial graph $G_0 = \emptyset$, $t = 0$

1: **while** not stopped **do**
2: $t = t + 1$
3: Operate robot and get data z^t, o^t
4: Select landmarks L_f^t with (3.53) and Algorithm 3-2
5: Select measurements z^{R_t} with (3.52) and Algorithm 3-1
6: $G_t = G_{t-1} \cup \{z^{R_t}, L_f^t\}$
7: **end while**

3.6 Experiments

3.6.1 Simulation

In this section experiments are run in a simulated environment, where ground truth is available to compare accuracies. Figure 3-6 shows the simulation environment. It includes both open areas on the left hand side and narrow passages on the right hand side. The parameters and statistics are listed in Table 3-1. We first run the robot once to get the initial dataset, then reduce robot poses and landmark measurements with various approaches. The sparsified maps are optimized with iSAM [13]. Repeated trials were performed on the generated maps to test the collision probability. Six selection strategies were compared:

(1) *optimal*: use all landmarks and all measurements;

(2) *coverage, glc*: use maximal coverage to select landmarks and robot poses [29,81]; use generic linear constraints (GLC) to marginalize out unselected variables [79];

(3) *all, info*: use all landmarks, and add measurements based on information gain on the overall graph [26];

(4) *focus, down-sampling*: select focused landmarks and uniformly down-sample measurements;

(5) *focus, info*: select focused landmarks for minimal collision, and select measurements based on information gain on focused variables(our proposed method);

Table 3-1: Simulated Dataset

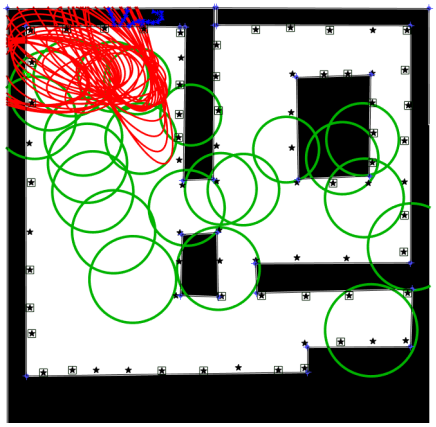
environment size	120m×120m
distance traveled	1066.9m
robot field of view	35m, 180^o
landmarks (black stars)	74
odometry measurements	1992
landmark measurements	12760
focused landmarks α	30
landmark measurement budget $K^R = 90$	90

Case (1) is a bound on other approaches. Case (2) is a map reduction approach where it starts with full variables and measurements. Typical use of (2) would require optimization of the full graph followed by reducing the map at the optimal point. However, in our case, resource is constrained and we try to avoid full graph optimization. Experiments both with and without prior full graph optimization are done and compared. Case (3) is a measurement sparsification approach but does not sparsify variables. Case (4) uses focused landmarks to minimize collisions, but uses a naive way to sparsify measurements.

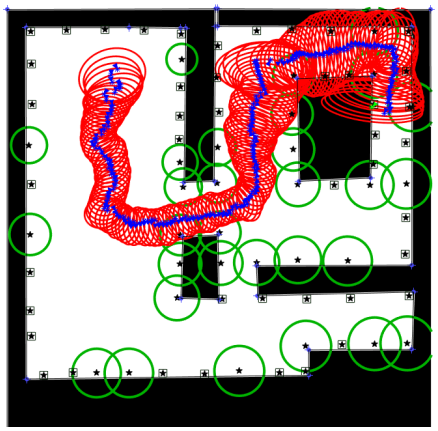
Figure 3-6 (a)–(e) shows sample trials for cases (2), (3), (4) and (5), respectively and Figure 3-7 shows the overall probabilities of collision obtained from all trials. The trials are stopped whenever there is an actual collision with an obstacle. The landmarks that are not selected are distinguished by small black boxes around them. The 3σ-ellipses corresponding to landmark uncertainty are shown in green. The blue line represents the nominal trajectory that the robot is trying to follow. The red ellipses represent the uncertainties of the robot along the nominal trajectory. In the focused landmark selection cases (5), the proposed procedure picks the landmarks that contribute more in reducing robot's uncertainty in desired regions (narrow passages) and spend the computational budget to reduce the uncertainty of these focused landmarks. In case (4), the measurements are spread across landmarks. As a result, each landmark gets very little resource thus the method failed to recover a meaningful map for navigation, thus showing that, in this case, measurement selection alone would not produce an acceptable result. In case (3) only select landmarks, more resources

Table 3-2: Comparison of simulated mapping results

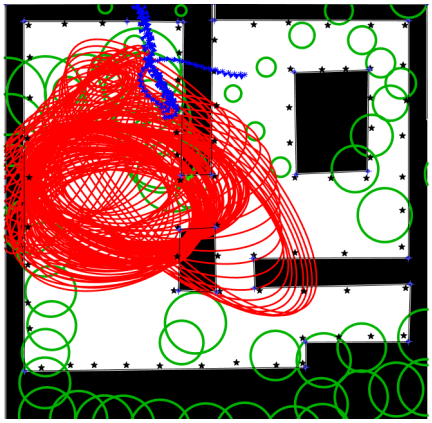
case	Variable Selection	Measurement Selection	Number of Landmarks	Number of Poses	Number of Edges	Mean Error on Landmarks (m)	Min Mahalanobis Distance (m)
1	all	all	74	1993	14753	0.02	57.67
2	max coverage	GLC	30	136	532	44.5	8.12
3	all	info gain	74	56	131	33.48	8.87
4	focus	down-sample	28	72	143	4.88	11.89
5	focus	info gain	30	30	120	**0.1345**	**14.53**



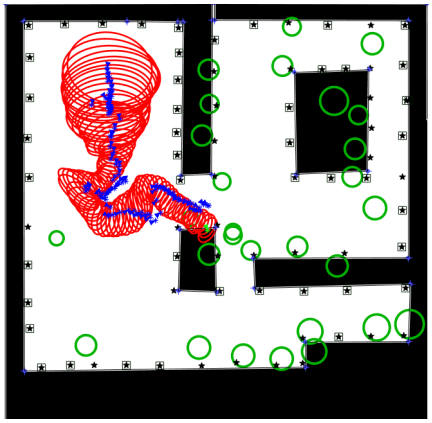
(a) case 2
maximal coverage and glc,
without optimal initialization



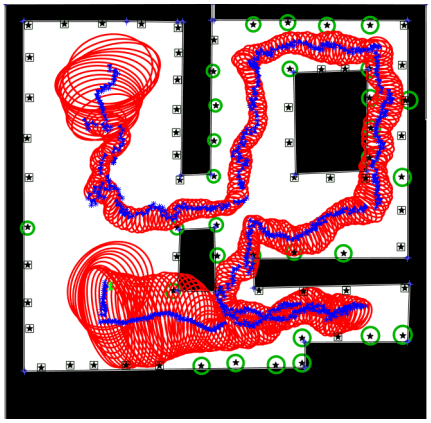
(b) case 2
maximal coverage and glc
with optimal initialization



(c) case 3
only select measurements



(d) case 4
only select landmarks



(e) case 5
two-stage selection

Figure 3-6: Navigation with focused mapping. Green circles represent selected landmarks with their size representing uncertainty. Blue lines are nominal trajectories each robot wants to follow with red circles representing pose uncertainty. Two-stage selection approach has much less uncertainty in narrow passages thus better navigation performance.

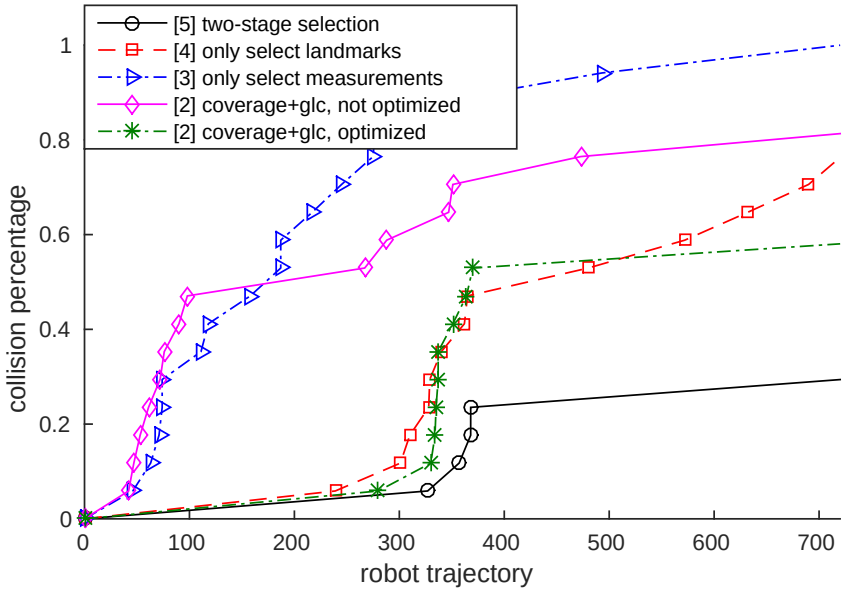


Figure 3-7: Collision probability of Monte Carlo simulations. The focused two-stage has lower collision probability compared to the unfocused case.

are spent on focused landmarks, thus the map is more accurate, but measurement selection is not based on how much they contribute to uncertainty reduction, thus the landmark positions are much less accurate than selecting both (case 3). The key point to note from Figure 3-6(a) is that the robot uncertainty is preferentially reduced in the areas of the environment where the corridors are tight and there is a higher chance of collision. In case (2), landmarks and robot poses are reduced based on maximal coverage, and variables are removed via sparse GLC. First GLC is more time-consuming than our approach(about 10 times slower), because the unfocused variables and measurements are the majority, sequentially removing them thus takes longer than sequentially adding the minority focused variables and measurements . Without a prior optimization over the full graph, sparse GLC loses information through marginalization, and the remaining graph failed to recover the right location of landmarks. A prior optimization over the full graph gives optimal landmarks locations and GLC maintains the right locations through marginalization. However, full graph optimization is not scalable, especially on resource constrained systems.

We further compare the cases from a mapping perspective in Table 3-2 based on three metrics: the sparsity of graph, represented by number of landmarks, robot poses and edges in the graph; the accuracy of SLAM, represented by the error on landmark locations; and the navigation performance, indicated by min Mahalanobis distance to obstacles along the robot trajectory. The proposed focused two-stage landmark and measurement selection approach achieves a sparser graph than others, and maintain accurate estimate of landmarks, and has maximal Mahalanobis distance, which means lower collision probability.

3.6.2 Office Environment

In the real-world experiment, a Turtlebot was run in a cluttered office space. The robot is equipped with an ASUS Xtion Pro RGB-D camera (we only use the RGB camera in this work) and a Hokuyo URG-04LX-UG01 laser range finder. Figure 3-9 shows the floor plan of the environment. AprilTags [82] were installed to create an initial pool of landmarks, as shown in Figure 3-8. The area consists of an office with desks, a doorway, and an open atrium with couches and chairs scattered. A summary of the dataset is provided in Table 3-3.

Table 3-3: Office Dataset

length	10min24s
distance traveled	115.5m
# odometry measurements	5211
# landmark measurements	8252
# landmarks	114

The odometry measurements are obtained from the Turtlebot's wheel encoder. The landmark measurements are obtained by running the AprilTag detector with the RGB images, which gives the relative orientation and range of the tags in the robot's frame. The selected measurements and odometry information are then fed into the standard SLAM solver iSAM [21] to optimize the graph. Note that we did not use the laser data for SLAM, only for detecting the closest distance to obstacles.

Figure 3-10 compares mapping results of case (1) (*optimal*), case (3) only select

Figure 3-8: The layout of the environment used for hardware experiments

measurements, case (4) only select landmarks and case (5) two-stage selection. The rebuilt robot trajectory is shown with a color map, where the red color on the trajectory indicates the risky (close to obstacles) regions and blue indicates the safer regions. Magenta circles represent landmarks with the size representing its uncertainty. The focused approach (Figure 3-10b) can concentrate the measurements on the narrow passage and door way, resulting in less uncertainty there. The other approaches scatter the measurements across different landmarks, and thus have much higher landmark uncertainty in narrow passages.

Figure 3-11 shows the navigation results. Magenta stars represent designed waypoints for robot to follow. Blue lines represent the estimated robot trajectory by Kalman filter. The proposed two-stage approach (3) has the fewest collisions, while "only select landmarks" (4) has more collisions and "only select measurements" (5) has very poor estimates of landmarks, thus the most number of collisions.

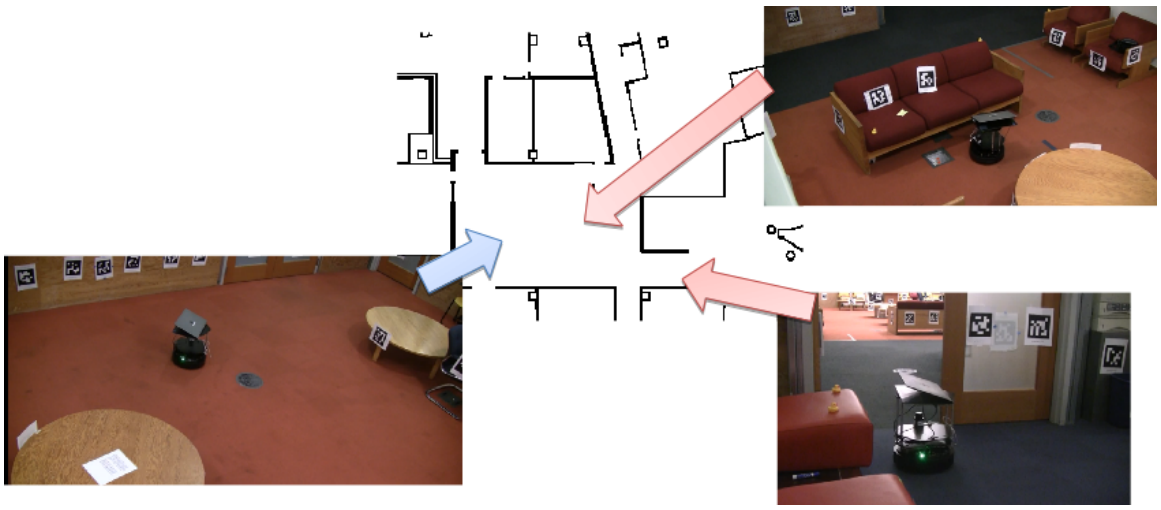


Figure 3-9: Floor plan of the hardware experiment environment. Narrow passages include a door way and a sofa cluster.

Table 3-4: Navigation performance

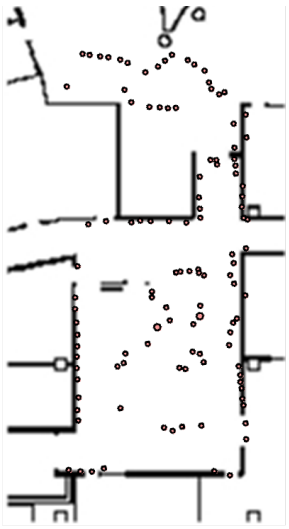
Method	length	# collisions
(3) two-stage selection	10m28s	2
(4) only select landmarks	10m52s	13
(5) only select measurements	28m40s	20

3.6.3 Incremental Selection

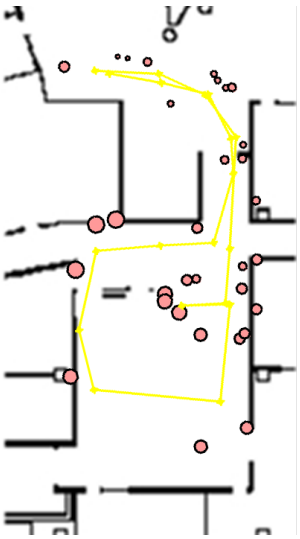
In incremental setting, the robot iterates between selecting landmark and measurement selection on streaming mini-batches of data. When a new mini-batch comes, it selects focused variables and measurements using results from the previous mini-batches as a prior. Both simulated and real-world data are used to show the incremental capability developed in section 3.5.

In the simulated environment, the dataset in Table 3-1 is divided into 5 mini-batches. The robot receives the 5 batches sequentially and run Algorithm 3-3 on each of them. After two-selection on each batch, the robot updates the posterior map with reduced variables and measurements, and throw away the original data. The Mahalanobis distance for selecting landmarks is set to be 100 and the information gain for selecting measurements is set to be 0.5.

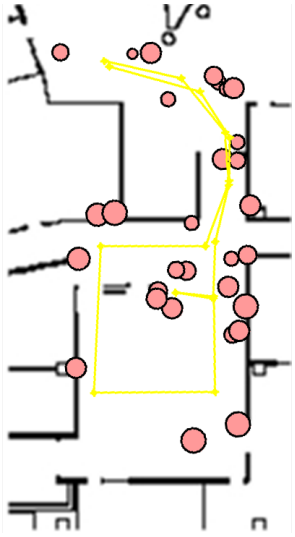
Figure 3-12 shows how the algorithm progress. In the first two mini-batches of data, the robot started in open area, selected few landmarks. As it traveled into



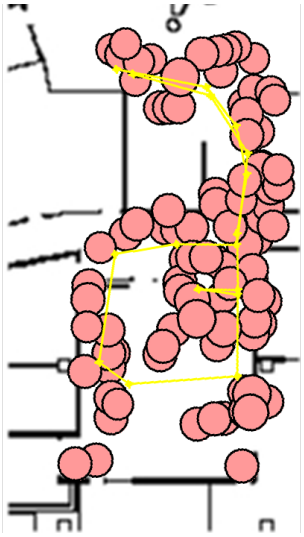
(a) case 1
optimal



(b) case 5
two-stage selection

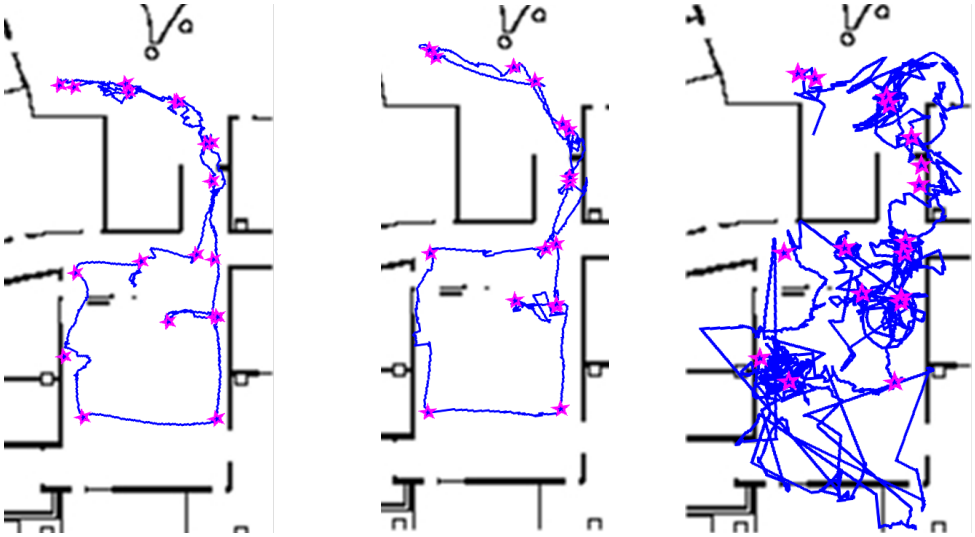


(c) case 3
only landmarks



(d) case 4
only measurements

Figure 3-10: Mapping results of Office dataset. Magenta circles represent landmarks with the size representing its uncertainty. The proposed two-stage approach (case 3) outperforms either selecting measurement only (case 5) or selecting landmark only (case 4) isolated



(a) case 5
two-stage selection

(b) case 3
only select landmarks

(c) case 4
only select measurements

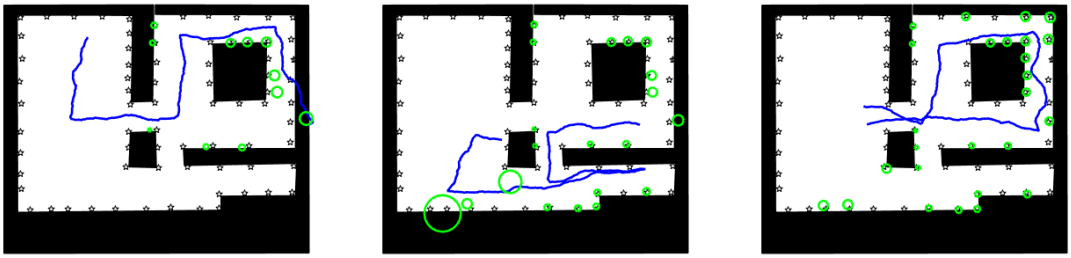
Figure 3-11: Navigation performance of office dataset. Magenta stars represent designed waypoints for robot to follow. Blue lines represent the estimated robot trajectory by Kalman filter. The proposed two-stage approach (Focus,Info) has least collisions, while measurement reduction (Full, Info) has bad estimates and landmark reduction (Focus, DownS) has more collisions

narrow passages, it selected more landmarks as focused variables. Notice the robot has not visited repeated places to close a loop yet, thus the uncertainty over the landmarks is high. After mini-batch 3, the robot comes back to the narrow places it visited in mini-batch 1 and 2. Loops are closed and the uncertainty on focused landmarks is significantly reduced. Notice that the robot did not keep mini-batch 1 and 2, but only relied on a reduced prior map to close the loops, which significantly reduced the memory and computation requirements.

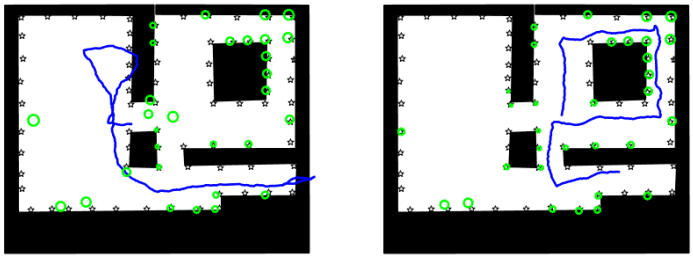
Table 3-5 compares the performance with incremental selection with batch selection. Even the incremental approach didn't use the full dataset at all, it maintained comparable sparsity and accuracy on landmarks as the batch procedure.

The incremental algorithm is also tested on the real world dataset in Table 3-3. The dataset is divided into five mini-batches. Their statistics are listed in Table 3-6.

Figure 3-13 shows the map after processing each mini-batch. Magenta circles represent selected landmarks after processing each new mini-batch. Black lines represents the robot's trajectory in each mini-batch. Same as the simulated case, the robot



(a) mini-batch 1 (b) mini-batch 2 (c) mini-batch 3



(d) mini-batch 4 (e) mini-batch 5

Figure 3-12: Incremental selection on simulated dataset. Green circles represent selected landmarks after processing each new mini-batch. Blue line represents the robot's trajectory in each mini-batch. The robot is able to use existing focused landmarks as priors and augment focused landmarks when passing through narrow passages.

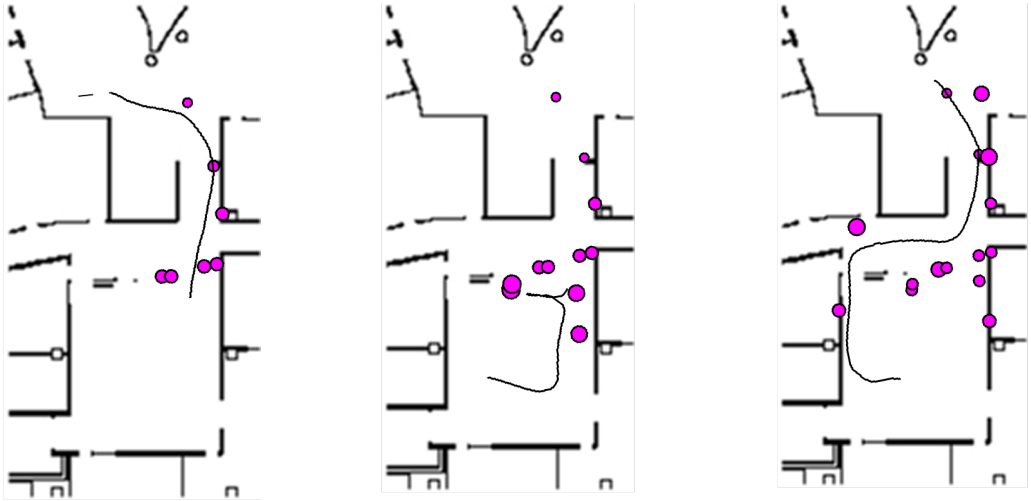
Table 3-5: Comparison between batch and incremental selection

	batch	incremental
Number of Landmarks	30	33
Number of Poses	30	43
Number of Edges	120	150
Mean Error on Landmarks	0.1345	0.8001
Min Mahalanobis Distance	14.53	27.05

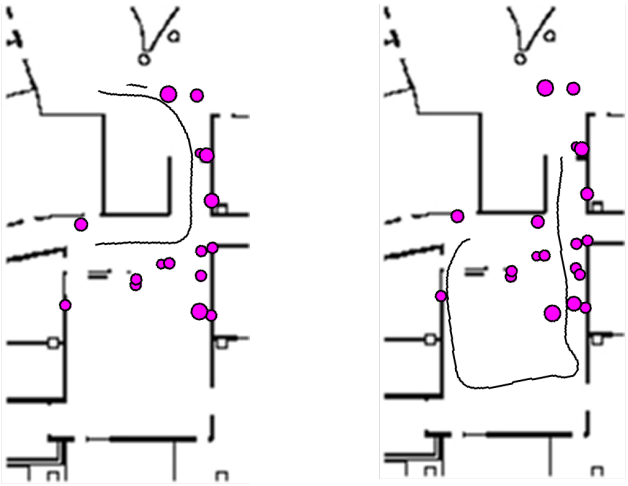
is able to use existing focused landmarks as priors and augment focused landmarks with new observed data.

3.7 Summary

This chapter presented a two-stage landmark and measurement selection procedure for map building on resource-constrained robots. Simulations and hardware results demonstrate that the approach can identify a relevant subset of landmarks and accurately localize them to reduce the probability of colliding with obstacles as compared



(a) mini-batch 1 (b) mini-batch 2 (c) mini-batch 3



(d) mini-batch 4 (e) mini-batch 5

Figure 3-13: Incremental selection on office dataset. This dataset contains 5 baches. Magenta circles represent selected landmarks after processing each new mini-batch. Black line represents the robot's trajectory in each mini-batch. The robot is able to use existing focused landmarks as priors and augment focused landmarks with new observed data.

Table 3-6: Statistics of Mini-batches

Mini-batch No.	Time(s)	Distance	No. measurements
1	120	16.87	1711
2	120	13.67	1049
3	120	25.12	1902
4	120	25.18	1240
5	120	29.10	1390

with existing approaches. As a result, the robot is able to navigate the environment for long periods of time, without the memory or computational requirements growing beyond the constraints. We have specifically focused on the navigation task here, but there are many inference and planning tasks that require a similar prioritization of variables. In this work, we provide a theoretically sound basis for selecting measurements to localize these important variables preferentially. This is an important capability for many resource-constrained real-time systems.

Chapter 4

Active Mapping with a Topological Feature Graph

The focused mapping technology developed in Chapter 3 relies on the existence of a dataset to build sparse maps. Such datasets are typically obtained by manually operating the robot in the environment first. However, for fully autonomous systems considered in this chapter, the robot would need to actively plan its paths to map the environment and localize itself within it. The problem of designing robot trajectories to actively explore an unknown environment and minimize the map error is referred to as active simultaneous localization and mapping (active SLAM). Active SLAM is non-trivial because the robot must trade-off the benefits of *exploring* new areas and *exploiting* visited areas to close loops [83].

Previous work has heavily relied on the occupancy grid (OG) map (grid of independent binary random variables denoting occupancy) to compute the information gain on map exploration and check feasibility. Such OG maps have large memory footprints such are not favorable on resource constrained systems. The landmark-based graph representation, on the other hand, can be sparsified to have low memory can computation costs. However, landmarks do not offer obstacle information, therefore active-SLAM on landmark-based graphs is challenging as it's hard for the robot to check path feasibility. This chapter proposes the first, to our knowledge active SLAM approach that plans robot paths to directly optimize a global landmark-based

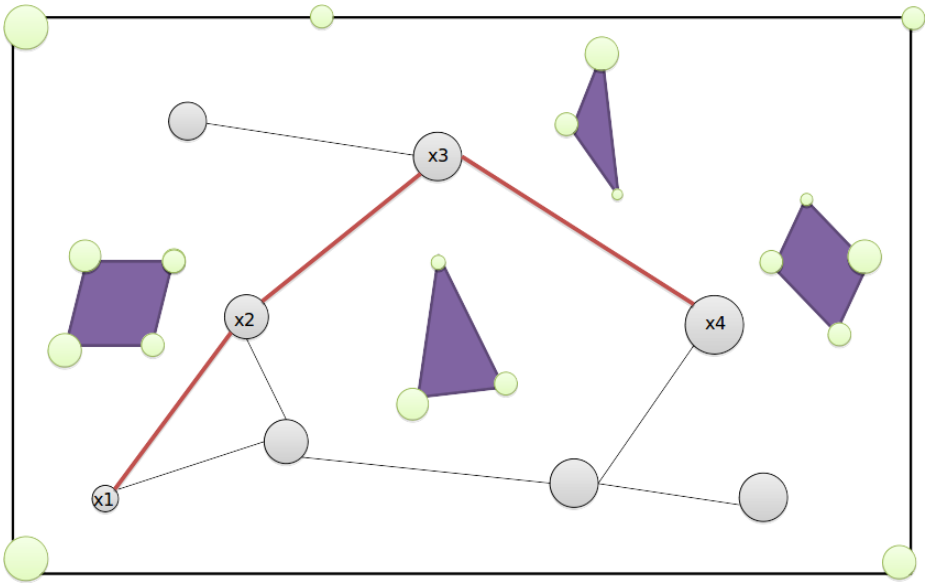


Figure 4-1: Active SLAM problem – purple polygons represent obstacles, green circles represent landmarks with their size denoting uncertainties in pose estimates. The problem is to find milestones (gray circles) of robot poses, and plan a trajectory (red line) that minimizes landmark uncertainties.

representation without any underlying OG representation. Rather than formulating the problem as area coverage over an OG map [84], we set it up as entropy reduction over the landmarks subject to a budget constraint. Since the landmark estimates and pose trajectory are necessarily correlated, we can remove the pose uncertainty from the traditional objective function and directly optimize over the map quality.

Fig. 4-1 shows an example scenario. The locations of landmarks are marked by green circles and the size of each circle represents its uncertainty. Gray circles represent samples of robot poses, and purple polygons represent obstacles. The planning problem is then to quantify information gains on the samples and find a trajectory connecting the samples that can minimize landmark uncertainties.

4.1 Active SLAM Problem

Recall that $\mathbf{L} = \{L_1, L_2, \cdots L_M\}$ denotes the static landmarks in the environment. Notice that the number of landmarks present in the environment could be less than M, and is not known a priori. The exact locations of the present landmarks are not

known *a priori* either and need to be established by the robot. When moving in the environment, the robot's trajectory is a sequence of poses $\mathbf{X}_T = \{X_0, X_1, \cdots, X_T\}$, where X_0 gives the initial distribution of the robot pose, typically set as the origin with low uncertainty. The robot can obtain two kinds of observations. The odometry o_t is the change between two consecutive poses with probability model $p(o_t; X_t, X_{t-1})$. A landmark measurement z_t^k is a measurement between the current pose X_t and landmarks y_t^k. The corresponding probability model of z_t^k is $p(z_t^k; X_t, L_{y_t^k})$. Denote $z_t = \{z_t^1, \cdots, z_t^{K_t}\}$.

Let $p(\mathbf{L}) = \prod_{i=1}^{M} p(L_i)$ denote the prior for landmarks. The joint posterior of $\mathbf{X}$ and $\mathbf{L}$ is then the product of priors and likelihood of the observations $\mathbf{o} = \{o_1, \cdots, o_T\}$ and $\mathbf{z} = \{z_1, \cdots, z_T\}$:

$$p(\mathbf{X}, \mathbf{L}|\mathbf{o}, \mathbf{z}) \propto p(\mathbf{L}) \prod_{t=1}^{T} p(o_t|X_t, X_{t-1}) \prod_{k=1}^{K_t} p(z_t^k|X_t, L_{y_t^k}). \tag{4.1}$$

The SLAM problem of jointly inferring the most likely posterior (MAP) landmark positions and robot poses can be defined as:

$$(\mathbf{X}^*, \mathbf{L}^*) = \arg\max_{\mathbf{X},\mathbf{L}} p(\mathbf{X}, \mathbf{L}|\mathbf{o}, \mathbf{z}) \tag{4.2}$$

With factor graph representation, (4.2) can be solved by readily available graph-SLAM algorithms/packages such as g2o, iSAM or GTSAM.

To get the odometry and landmark measurements, the robot is typically manually operated in the environment to gather a dataset first and then the map is optimized with the gathered batch data. In this chapter, the robot actively plans its own trajectory to incrementally learn the map. Considering that robots are typically constrained in computation/memory, the trajectory should be planned in such a way that resources should be spent on gathering information that is directly related to the robot's goal. The task of this chapter is to incrementally build a map of the environment, therefore information gain is defined as entropy reduction only on variables representing landmarks.

While there are many ways to quantify information gain, this chapter uses Shan-

non entropy [71] as the measure of uncertainty in random variables. Denote the control command at time t as u_t, and let $\mathbf{u}_T = \{u_1, \cdots, u_T\}$. The active SLAM problem is summarized as follows:

Problem 7. ***Active SLAM:*** *Design control commands* $\mathbf{u}_T = \{u_1, u_2, \cdots, u_T\}$*, such that the robot follows a trajectory that the obtained odometry* $\mathbf{o} = \{o_1, \cdots, o_T\}$ *and landmark measurements* $\mathbf{z} = \{z_1, \cdots z_T\}$ *can minimize the entropy* $H(\cdot)$ *over the belief of landmarks* $\mathbf{L}$*:*

$$\begin{aligned} \min_{\mathbf{u}_T = \{u_1, \cdots, u_T\}} \quad & H(\mathbf{L}|\mathbf{o}, \mathbf{z}) \\ s.t. \quad & q(\mathbf{u}_T) \leq c \\ & X_t = g(X_{t-1}, u_t) \\ & o_t = X_t \ominus X_{t-1} + v, \quad v \sim \mathcal{N}(0, Q) \\ & z_t^k = L_{y_t^K} \ominus X_t + w, \quad w \sim \mathcal{N}(0, R) \\ & t = 1, \cdots, T \end{aligned} \tag{4.3}$$

where $q(\cdot)$ *is a measure of control cost, in the case of finite time horizon,* $q(\mathbf{u}_T) = T$*. Function* $X_t = g(X_{t-1}, u_t)$ *describes the robot dynamics. Function* $o_t = X_t \ominus X_{t-1} + v$ *describes the odometry model and* $z_t^k = L_{y_t^K} \ominus X_t + w$ *is the landmark measurement model.*

Fig. 4-2 presents a graphical model of this problem. X_t represents robot poses, $\mathbf{L}$ represents environment landmarks. The goal is to design control policies $u_{1:T}$ to maximize information gain over landmark belief $\mathbf{L}$.

4.2 Method

4.2.1 Topology feature graph

One important reason that the use of grid-map representation has been a popular choice for active-SLAM is that a grid-based map contains all the necessary information for path planning. A graph-based representation, although much sparser, lacks

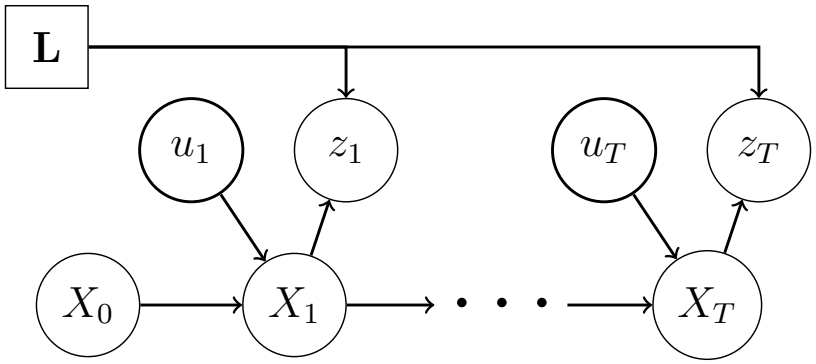


Figure 4-2: Active SLAM. Design a control policy $u_{1:T}$ such that the landmark measurements z_t and odometry obtained along the path maximizes information gain over environmental landmarks $\mathbf{L}$.

information about free/occupied space and the topology of the environment. Consequently, planning paths over a traditional graph-based representation is ill-posed. To overcome this, we propose to store additional information with each landmark that allows us to generate a full, yet sparse, representation of the environment over which we can then plan paths.

In this thesis, it is assumed that the robot is a ground robot that operates in 2D space. Extension to 3D scenarios can be achieved by triangulating obstacle surfaces and is left to future work. Relying on the fact that landmarks are usually on the surface or corner of obstacles, this chapter proposes the *Topological Feature Graph* (TFG) representation. A TFG is a graph $\mathcal{G} = \{L, E\}$, with its vertices representing landmarks and edges representing obstacles. More specifically, if two landmarks are connected by an edge, then these two landmarks belongs to the same flat obstacle surface and the edge is not traversable[1].

These edges can be learned from either a depth image, a laser scan or even sequences of images [55]. The robot first segments the depth map or laser scan into several components representing different obstacle surfaces, then checks if two landmarks detected belong to the same component. If so, the robot creates an edge between these two landmarks. This idea is illustrated in Figure 4-3a. The stars are vertices that represents landmarks, and the black lines are edges representing obstacle

[1]Landmarks can be extended to objects that have sizes, in which case obstacles would be represented by both objects represented by vertices and surfaces represented by edges.

surfaces.

Compared to the grid map representation, the TFG offers several advantages in structured environments. First it requires many fewer variables to represent the environment, and thus provides significant memory savings. Second, the map complexity can easily adapt to various complexities in the environment. Instead of using equal sized cells at all places, a TFG can model more landmarks in cluttered/narrow spaces and less landmarks in wider/simpler spaces. Third, if new loop closures are detected and drifts of some subgraphs are corrected, the obstacles will be corrected with the landmark positions: the robot does not have to relearn the occupancy of the associated space. And finally, this representation has a closed-form collision check for robot path planning rather than sampling-based methods, leading to significant computation savings in path planning.

4.2.2 Sequential Planning Problem

Recall that the goal is to plan robot controls that gain maximal information from the environment as formulated in Problem 7. Notice that solving Problem 7 in batch is hard in general, because at any time t, observations beyond t are not available, thus planning controls $u_t, \cdots u_T$ will require modeling future observations and taking into account all possible outcomes, which is typically intractable.

To solve this problem in a tractable manner, a widely used technique is to split Problem 7 into T stages, optimize a goal point at each stage [85, 86]. For each stage, a separate path planner can be used to generate controls.

Let $p(\mathbf{L})$ denote a prior of the landmarks. At stage t, the observation history $o_{1:t}$ and $z_{1:t}$ can be summarized in a posterior distribution of $\mathbf{L}, \mathbf{X}$ at time t. Denote the maximal posterior(MAP) values of $\mathbf{X}$ and $\mathbf{L}$ as $\mathbf{X}_t^*$ and $\mathbf{L}_t^*$, they can be obtained by standard SLAM solvers:

$$\begin{aligned} \mathbf{X}_t^*, \mathbf{L}_t^* &= \text{argmax } p(\mathbf{X}, \mathbf{L} | \mathbf{o}_t, \mathbf{z}_t) \\ &= \text{argmax } p(\mathbf{L}) \prod_{\tau=1}^{t} p(o_\tau | X_\tau, X_{\tau-1}) p(z_\tau | X_\tau, \mathbf{L}) \end{aligned} \tag{4.4}$$

From section 2.1.3, the probability of $\mathbf{X}$ and $\mathbf{L}$ can be approximated by a Gaussian distribution. The mean is its MAP values $(\mathbf{X}_t^*, \mathbf{L}_t^*)$ and its information matrix Λ is the second moment:

$$\mathbf{X}, \mathbf{L}|\mathbf{o}_t, \mathbf{z}_t \sim \mathcal{N}(\mathbf{X}_t^*, \mathbf{L}_t^*; \Lambda^{-1}) \tag{4.5}$$

$$\begin{aligned} \Lambda =& \frac{\partial^2 \log p(\mathbf{L})}{\partial(\mathbf{X}, \mathbf{L})^2} + \sum_{\tau=1}^{t} \frac{\partial^2 \log p(o_\tau)}{\partial(\mathbf{X}, \mathbf{L})^2} + \sum_{i=1}^{M} \frac{\partial^2 \log p(z_\tau)}{\partial(\mathbf{X}, \mathbf{L})^2} \\ =& \begin{bmatrix} \Lambda_f & \Lambda_{fr} \\ \Lambda_{rf} & \Lambda_r \end{bmatrix} \end{aligned} \tag{4.6}$$

where Λ_f corresponding to landmarks and Λ_r corresponds to robot poses. Laplacian approximation gives close-form solutions for entropy. The marginal information matrix for landmarks is $\Lambda_{\mathbf{L}} = \Lambda_f - \Lambda_{fr}\Lambda_r^{-1}\Lambda_{rf}$, and the entropy is

$$H(\mathbf{L}|\mathbf{o}_t, \mathbf{z}_t) = -\frac{1}{2} \log |\Lambda_{\mathbf{L}}| + \text{constant} \tag{4.7}$$

Further with the associated connectivity edges between landmarks, we obtain the TFG at time t. Denote it as TFG_t, which summarizes the information the robot has about the environment up until time t.

The Laplacian approximation simplifies the information quantification, but directly optimizing over controls u_t is still very difficult. Control inputs u_t affect robot paths though robot dynamics, and optimization under both robot dynamic constraints and obstacle constraints would be computationally prohibitive. As such, the problem is further simplified here by planning a trajectory for the robot first, then using a separate path-following controller to drive the robot along the planned trajectory. In this way, controller design is decoupled from path planning.

Problem 8. ***Path Planning for Active SLAM*** *At stage t, given prior topological feature graph TFG_t, find a path $\widehat{X}_t \cdots, \widehat{X}_\tau, \cdots, \widehat{X}_{t+1}$, such that the posterior entropy*

on landmarks $\mathbf{L}$ *is minimized:*

$$\begin{aligned}
\min_{\widehat{X}_t,\cdots,\widehat{X}_{t+1}} \quad & H(\mathbf{L}|TFG_t, \hat{\mathbf{o}}_{t+1}, \hat{\mathbf{z}}_{t+1}) \\
s.t. \quad & \widehat{X}_\tau = g(\widehat{X}_{\tau-1}, u_\tau) \\
& \hat{o}_\tau = \widehat{X}_\tau \ominus \widehat{X}_{\tau-1} + v, \quad v \sim \mathcal{N}(0, Q) \\
& \hat{z}^k_\tau = L_{y^K_\tau} \ominus \widehat{X}_\tau + w, \quad w \sim \mathcal{N}(0, R) \\
& \tau = t, \cdots, t+1
\end{aligned} \tag{4.8}$$

where $\hat{\mathbf{o}}_\tau = \{\hat{o}_t, \cdots, \hat{o}_\tau, \cdots, \hat{o}_{t+1}\}$ *represents the odometry obtained along the trajectory. And* $\hat{\mathbf{z}}_{t+1} = \{\hat{z}_t, \cdots, \hat{z}_\tau, \cdots \hat{z}_{t+1}\}$ *represents landmark measurements obtained along the trajectory.*

Given the path $\widehat{X}_t, \cdots, \widehat{X}_{t+1}$ and the partial TFG at time t, a separate path-following controller could be used to drive the robot along the trajectory. In this way, path planning and control are decoupled from the active SLAM problem, and we gain performance in computation and speed.

4.2.3 Expected Information Gain

Quantifying the exact information gain from $\widehat{X}_t$ to $\widehat{X}_{t+1}$ in Problem 8 is challenging because it involves discretizing the trajectory from X_t to $\widehat{X}_{t+1}$ into a sequence of robot poses $\widehat{X}_\tau$, then computing the information gain from measurements at each pose. Information gain of measurements on later poses will depend on earlier poses along the path. Therefore, the complexity will grow exponentially with the path length. To solve the information quantification problem in real-time, we only plan a goal point for the robot, design the robot to stabilize its pose at the goal point, rotate in-place to obtain accurate observations of the local environment, and compute information gain only on these locally observable landmarks at the goal point. The observation would be some layout of a subset of the local landmarks. As shown in Figure 4-3b, gray balls denote observation points, and the blue circle indicate the set of landmarks it can observe at those observation points.

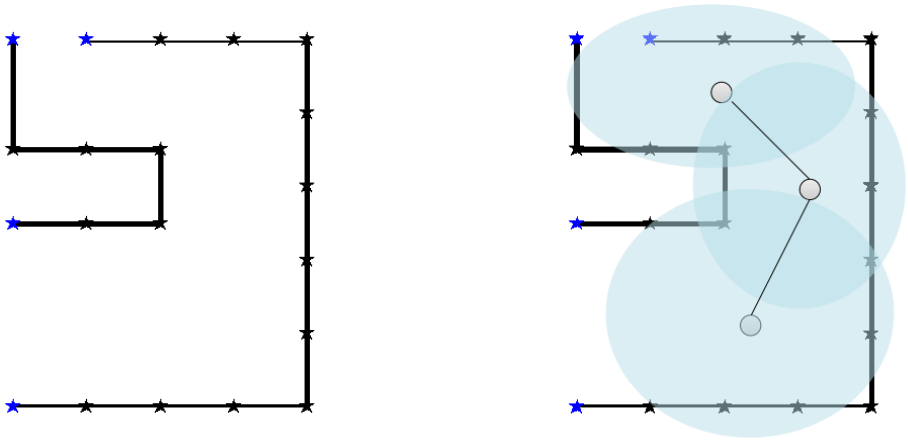
Problem 9. ***Goal Planning for Active SLAM*** *At stage t, given prior topological feature graph TFG_t, find the next goal point $\widehat{X}_{t+1}$ such that the entropy on landmarks* $\mathbf{L}$ *is minimized:*

$$\begin{aligned} \min_{\widehat{X}_{t+1}} \quad & H(\mathbf{L}|TFG_t, \hat{z}_{t+1}) \\ s.t. \quad & \hat{z}_{t+1} = h(\hat{X}_{t+1}, \mathbf{L}) \end{aligned} \tag{4.9}$$

Goal points also provide a way to segment the overall map into local maps and sparsify the underlying SLAM factor graph: the robot accurately maps the environment at goal points, thus measurements between two goal points contains less information compared to those at goal points. Therefore, landmark measurements along the path are only used to localize the robot, but are not used to update landmark estimates. This may cause some loss of information. However, with this simplification, we can marginalize out robot poses between two observations points, and the SLAM factor graph will become a joint graph of partial graphs at goal points. In this way, the complexity of the SLAM factor graph only scales with the number of observation points and not the number of robot poses.

Furthermore, paths are generated with respect to the current estimate of landmark locations. If measurements along a path are used to update landmark estimates, new loop closures may cause shifts in landmark locations. The old path may become invalid and the robot may run into obstacles. Leaving out measurements along the path also avoids this potential failure.

Notice that $\widehat{X}_{t+1}$ is in continuous $\mathbb{R}^2$ space. Different $\widehat{X}_{t+1}$ would give different combinations of observable landmarks, thus solving problem 9 exactly would be hard. Instead, we use a random sampling approach. Given TFG_t, a location is *reachable* if it can be observed from some previous goal location. The planner samples locations in the robot's reachable space, computes entropy reduction for each goal point, then selects the next goal point as the one that gives maximal entropy reduction.



(a) Topology Feature Graph(TFG) **(b)** Goal points

Figure 4-3: Topological Feature Graph (TFG) and goal points. Vertices (stars) represent landmarks, edges (black lines) represent obstacle surfaces, blue stars represent landmarks at a frontier, gray balls represent goal points. Blue regions illustrate local observable landmarks.

The maximal entropy reduction problem can be stated as follows:

$$\begin{aligned} \widehat{X}_{t+1} &= \underset{X_{t+1}}{\operatorname{argmax}} \quad \Delta H(\mathbf{L}|X_{t+1}, TFG_t) \\ &= \underset{X_{t+1}}{\operatorname{argmax}} \quad H(\mathbf{L}|TFG_t) - H(\mathbf{L}|X_{t+1}, TFG_t) \end{aligned} \tag{4.10}$$

Theorem 2. *Set prior covariance for unknown landmarks in such a way that it is much larger than covariance of observed landmarks. Given topological landmark graph TFG_t, the entropy reduction at goal location X_{t+1} can be approximated by the sum of entropy reduction on local observable landmark dH_o, and of new landmarks dH_u*

$$\Delta H(X_{t+1}|X_{t+1}, TFG_t) \approx \Delta H_o + \Delta H_u \tag{4.11}$$

where $\Delta H_u = n_x \log|I + \sigma_u a_u|$, n_x is the number of new landmarks n_x, $\log|I + \sigma_u a_u|$ the expected information gain on an unknown landmarks.

Proof. For simplicity, the subscript t is dropped in the following, but it should be noted that this analysis is based on TFG_t.

The information matrix can be written into two parts that corresponds to observed

landmarks Λ_f or robot poses Λ_r

$$\Lambda = \begin{bmatrix} \Lambda_f & \Lambda_{fr} \\ \Lambda_{rf} & \Lambda_r \end{bmatrix}$$

The robot task here is to map the landmark, therefore we only look at the marginal information matrix on landmarks:

$$\Lambda_{\mathbf{L}}^t = \Lambda_f - \Lambda_{fr}\Lambda_r^{-1}\Lambda_{rf} = \begin{bmatrix} \Lambda_o & 0 \\ 0 & \Lambda_u \end{bmatrix}$$

where Λ_o corresponds to landmarks observed at least once, and Λ_u corresponds to landmarks that have not been observed yet. At goal point $\widehat{X}_{t+1}$, denote $\hat{z}_{t+1}$ are the expected new landmark measurements, then the new joint likelihood becomes:

$$p(TFG_t, \hat{z}_{t+1}; \widehat{X}_{t+1}, \mathbf{X}, \mathbf{L}) \sim p(TFG_t; \mathbf{X}, \mathbf{L}) p(\hat{z}_{t+1} | \widehat{X}_{t+1}, TFG_t) \tag{4.12}$$

The corresponding factor graph is the factor graph at t plus new landmark measurements $p(\hat{z}_{t+1}|\widehat{X}_{t+1}, TFG_t)$. Using the same ML values X_t^*, $\mathbf{L}_t^*$ in (4.2), the new information matrix $\Lambda_{\mathbf{L}}^{t+1}$ would be the original information matrix $\Lambda_{\mathbf{L}}^t$, plus some new terms coming from factors $p(\hat{z}_{t+1}|\widehat{X}_{t+1}, TFG_t)$:

$$\Lambda_{\mathbf{L}}^{t+1} = \begin{bmatrix} \Lambda_o & 0 & 0 \\ 0 & \Lambda_u & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} A_o & 0 & H_o \\ 0 & A_u & H_u \\ H_o^T & H_u^T & B \end{bmatrix} \tag{4.13}$$

where

$$B = B_o + B_u$$
$$\begin{bmatrix} A_o & H_o \\ H_o^T & B_o \end{bmatrix} = \frac{\partial^2 p(z_{t+1}|\widehat{X}_{t+1}, TFG_t)}{\partial(\mathbf{L}_o, \widehat{X}_{t+1})^2} \tag{4.14}$$

$$\begin{bmatrix} A_u & H_u \\ H_u^T & B_u \end{bmatrix} = \frac{\partial^2 p(z_{t+1}|\widehat{X}_{t+1}, TFG_t)}{\partial(\mathbf{L}_u, \widehat{X}_{t+1})^2}$$

The marginal information matrix on landmarks can be computed from the Schur complement:

$$\begin{aligned} \Lambda_{\mathbf{L}}^{t+1} &= \begin{bmatrix} \Lambda_o + A_o & 0 \\ 0 & \Lambda_u + A_u \end{bmatrix} - HB^{-1}H^T \\ H &= \begin{bmatrix} H_o \\ H_u \end{bmatrix} \end{aligned} \tag{4.15}$$

Note that elements in A and H are 0 if the corresponding landmark is not observable at observation point $\hat{X}_{t+1}$. The incremental change in the information objective $H(\cdot)$ is:

$$\begin{aligned} \Delta H &= -\log|\Lambda_{\mathbf{L}}^t| + \log|\Lambda_{\mathbf{L}}^{t+1}| \\ &= \log\left|\begin{bmatrix} \Lambda_o + A_o & 0 \\ 0 & \Lambda_u + A_u \end{bmatrix} - HB^{-1}H^T\right| - \log\left|\begin{bmatrix} \Lambda_o & 0 \\ 0 & \Lambda_u \end{bmatrix}\right| \end{aligned} \tag{4.16}$$

Take the inverse of the matrix in second term, combine it with the first term, then use $\Lambda_o^{-1} = \Sigma_o$, $\Lambda_u^{-1} = \Sigma_u$ to obtain

$$\Delta H = \log\left|\begin{bmatrix} I + \Sigma_o A_o & 0 \\ 0 & I + \Sigma_u A_u \end{bmatrix} - \begin{bmatrix} \Sigma_o & 0 \\ 0 & \Sigma_u \end{bmatrix} HB^{-1}H^T\right|$$

Extract the first term to get

$$\Delta H = \log\left|I + \begin{bmatrix} \Sigma_o A_o & 0 \\ 0 & \Sigma_u A_u \end{bmatrix}\right| + \log\left|I - \begin{bmatrix} I + \Sigma_o A_o & 0 \\ 0 & I + \Sigma_u A_u \end{bmatrix}^{-1} HB^{-1}H^T\right|$$

Apply $|I - BA| = |I - AB|$ on the second term

$$
\begin{aligned}
&= \log \left| I + \begin{bmatrix} \Sigma_o A_o & 0 \\ 0 & \Sigma_u A_u \end{bmatrix} \right| \\
&+ \log \left| I - B^{-1} H^T \begin{bmatrix} (I + \Sigma_o A_o)^{-1} & 0 \\ 0 & (I + \Sigma_u A_u)^{-1} \end{bmatrix} H \right| \\
&= \log |I + \Sigma_o A_o| + \log |I + \Sigma_u A_u| \\
&+ \log \left| I - B^{-1} H_o^T (I + \Sigma_o A_o)^{-1} H_o - B^{-1} H_u^T (I + \Sigma_u A_u)^{-1} H_u \right|
\end{aligned} \tag{4.17}
$$

When a landmark has not been previously observed, the prior covariance Σ_u is typically large, therefore $H_u^T (I + \Sigma_u A_u)^{-1} H_u$ is small compared to $H_o^T (I + \Sigma_o A_o)^{-1} H_o$. Furthermore, notice that when the prior Σ_u and information delta A_u are block diagonal, with each block representing a landmark, $\log |I + \Sigma_u A_u| = n_x \log |I + \sigma_u a_u|$, and we have the following approximation:

$$
\begin{aligned}
\Delta H \approx & \log |I + \Sigma_o A_o| + \log \left| I - B^{-1} H_o^T (I + \Sigma_o A_o)^{-1} H_o \right| \\
& + n_x \log |I + \sigma_u a_u| \\
= & \log |I + \Sigma_o A_o - H_o B^{-1} H_o| + n_x \log |I + \sigma_u a_u| \\
= & \Delta H_o + \Delta H_u
\end{aligned} \tag{4.18}
$$

where $\Delta H_o = \log |I + \Sigma_o A_o - H_o B^{-1} H_o|$ is the information gain obtained by having new measurements on observed landmarks, $\Delta H_u = n_x \log |I + \sigma_u a_u|$ is information obtained by having new measurements on previously unobserved landmarks, n_x is the number of new landmarks observed, and σ_u and a_u are the variance and information gain of a single new landmark. □

Theorem 2 indicates that the information gain on a goal point can be split into two parts: the first part ΔH_o is the information gain obtained by re-observing and improving known landmarks, and ΔH_u is the information gain from exploring new landmarks. In our experiments, n_x is computed by using a predefined landmark

density in the environment multiplied by the size of a frontier at observation point X_{t+1}.

4.2.4 Frontier Detection

In order to detect frontiers, we track how each landmark is connected to its neighbors. A landmark borders a frontier if at least one side of it is not connected to any neighbors. As shown in Figure 4-3a, the blue stars represent landmarks at frontiers. At each sample location, the size of frontier is computed as following:

1. Compute landmarks the robot expects to observe
2. Sort the landmarks according to their orientation relative to the robot
3. If two consecutive landmark are not connected to any neighbors, they represent a frontier.

With this frontier detection approach, we have a uniform information metric for both observed landmarks and unobserved landmarks at frontiers. Thus our approach gives a natural balance between exploration and exploitation: if there are large frontiers offering the potential to discover many new landmarks, the robot will pick observation points to explore frontiers. If there are only small frontiers or none at all, the robot might go to visited places to improve existing landmarks estimates.

4.2.5 Path Planning

In Section 4.2.2, we obtained a set of collision-free samples, therefore the path planner will only compute connectivity and cost between these samples, and form a probabilistic roadmap (PRM). The trajectory to the next best observation point is generated by computing a minimum cost path on a PRM. The cost of an edge between two sample points involves two factors:

- The length of the link, which reflects the distance that needs to be traveled and thus the control costs.

- Collision penalty.

Computing the exact collision probability of a given path is a computationally expensive procedure. However, exploiting the fact that a collision check for a point using a TFG representation can be carried out analytically enables expensive methods such as Monte Carlo methods for real-time collision evaluation. in this work, assuming Gaussian localization uncertainty, we rely on very efficient approximate methods to compute a measure of risk instead of the exact collision probability.

Denote x_o as the closest obstacle point. Then $||x - x_o||^2$ represents the squared distance to the closest point and reflects the chance of collision. Thus we use $||x-x_o||^2$ as an additive penalty in the edge cost in path planning. With our TFG representation, computing $||x - x_o||^2$ reduces to computing point-line and line-line distances, which can be achieved trivially.

One of the key benefits of relying on the TFG for path planning is the analytic computation of collisions. In other words, since TFG is composed of set of lines, one can analytically verify a given point is in the obstacle region or not by checking TFG lines around the robot. Such a fast collision check enables accurate methods such as Monte Carlo to evaluate collision probability along the path. We rely on a chance constraint formulation (similar to [39, 40, 87]) to compute paths that satisfy $\Pr(\text{path} \in \text{Obstacle}) < \delta$. If one relaxes this constraint to $\Pr(x_k \in \text{Obstacle}) < \delta \forall k, \quad x_k \sim \mathcal{N}(\hat{x}_k, P_k)$

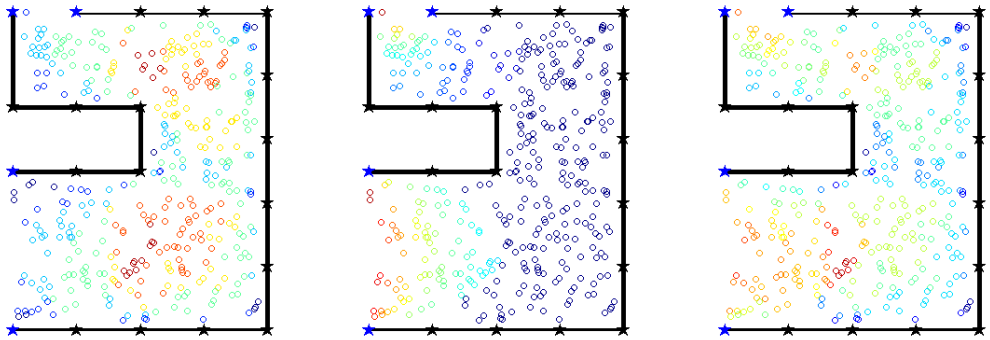
$$\Pr(x_k \in \text{TFG edge}) < \delta \quad \forall k, \quad x_k \sim \mathcal{N}(\hat{x}_k, P_k) \tag{4.19}$$

where, x_k is the k-th point on the trajectory

4.3 Experiments

4.3.1 Information Measures

We first illustrate how the proposed framework can balance exploration and exploitation. Figure 4-4 shows an example scenario: black lines represent obstacles, stars

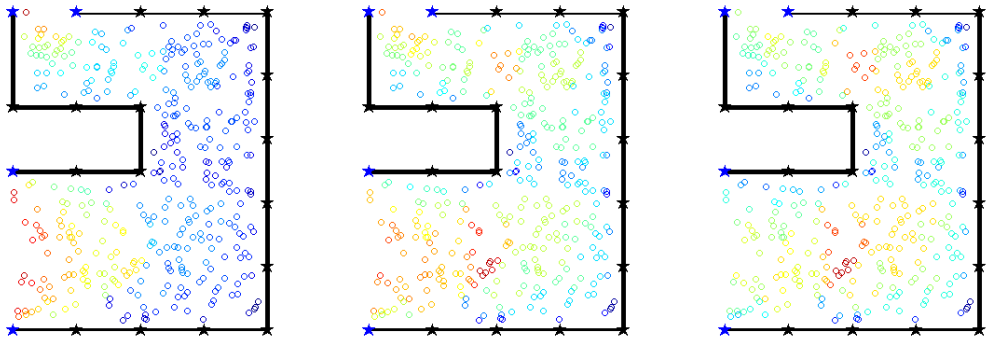


(a) Exploitation **(b)** Exploration **(c)** Total

Figure 4-4: Information gain. Black lines (obstacles) and stars (landmarks) comprise the TFG. Blue stars indicate frontier landmarks. Circle color represents information gain on samples.

represent landmarks with blue stars bordering frontiers. Circles are samples in the free space with color representing their information gain: red is high gain and blue is low. Figure 4-4a displays the information gain on observed landmarks: the total information gain is largest at samples that can potentially observe the greatest number of landmarks. On the other hand, Figure 4-4b shows the information gain on new landmarks. The samples closer to frontiers will have a chance to observe new landmarks, thus they have higher exploration gains than samples further from frontiers. Assuming a fixed new landmark density, larger frontiers offer the potential to observe more new landmarks and therefore nearby samples have greater exploration information gain. Figure 4-4c shows the total exploration and exploitation information gain.

Summing both the exploitation and exploration information, Figure 4-5 displays the total information gain under high/medium/low prior variance on new landmarks. When prior variance on new landmarks is high, observing a new landmark will give large information gains, thus the exploration term dominates the exploitation term, and the robot prefers sample points at frontiers. On the other hand, if the prior variance is set to be low, observing new landmarks does not add much information, and the robot will prefer to revisit places with observed landmarks and improve its estimate of their positions.



(a) High **(b)** Medium **(c)** Low

Figure 4-5: Total information gain with varying unseen landmark density. When the robot expects to see many landmarks beyond frontiers, information gain at frontiers is high. Otherwise, the robot prefers spots that can observe the most landmarks in visited places.

4.3.2 Simulation

We compared our framework with a nearest-frontier exploration algorithm [83] using the Gazebo simulator. The frontier exploration simulation used the popular GMapping [88] system for localization and mapping and used wavefront frontier detection [89] to identify frontiers.

The simulated Turtlebot receives noisy odometry, laser scans, and landmark measurements. Table 4-1 contains the simulation parameters. Figure 4-6a displays a screenshot of the simulated environment (simulated AprilTags are spaced roughly one meter apart along the walls).

Figure 4-6b and 4-6c display the maps generated by frontier exploration and TFG active SLAM respectively over one run. Note that there is obvious distortion along the hallways, and the boundary of some obstacles in the center are blurred as well. On the other hand, TFG active SLAM was able to close loops on landmarks and thus maintain the shape of the building in its map.

Figure 4-8 compares the robot pose error of nearest-frontier and TFG active SLAM over 4 runs. The solid lines represent error mean and shades represent error range. TFG active SLAM consistently has significantly less error in its robot pose estimates, especially in position. Figure 4-7 compares the map coverage with time spent exploring. In TFG active SLAM, the robot balances exploration with loop closing and

Table 4-1: Simulation parameters

size of environment	46m×22m
No. of landmarks	274
sensor range	10m
field of view	124 degrees
particles for gmapping	100
rate for gmapping update	0.33Hz
rate for landmark measurements	10Hz

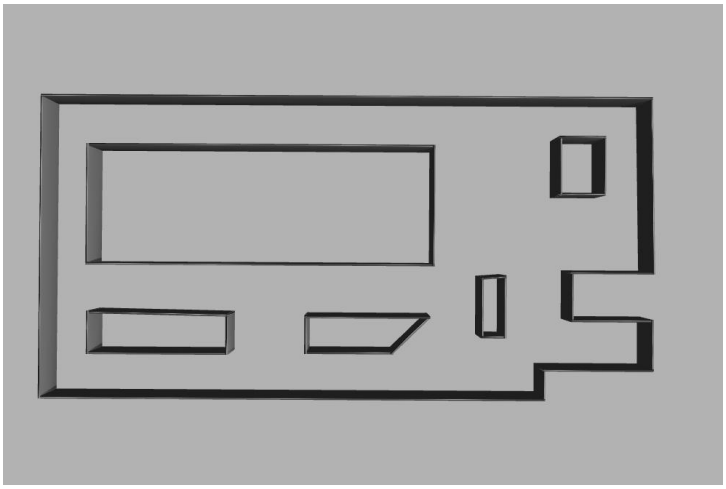
Table 4-2: Simulation Performance Comparison

	TFG Active SLAM	grid map frontier
No. of variables	274	800000
CPU idle time	75%	0%
time (s)	2433 ± 546	2293 ± 375
position error (m)	0.147 ± 0.115	5.26 ± 3.53
orientation error (rad)	0.0217 ± 0.016	0.0213 ± 0.0165

is thus slightly slower in covering the whole space when compared with greedy frontier exploration. Table 4-2 compares algorithm performance. Although TFG active SLAM takes slightly longer to explore the environment, it uses orders of magnitude fewer variables to represent the world, which leads to memory savings. Frontier exploration also updates particles and the grid map continuously while TFG exploration only updates its map at goal points, requiring only light computation throughout most of its operation.

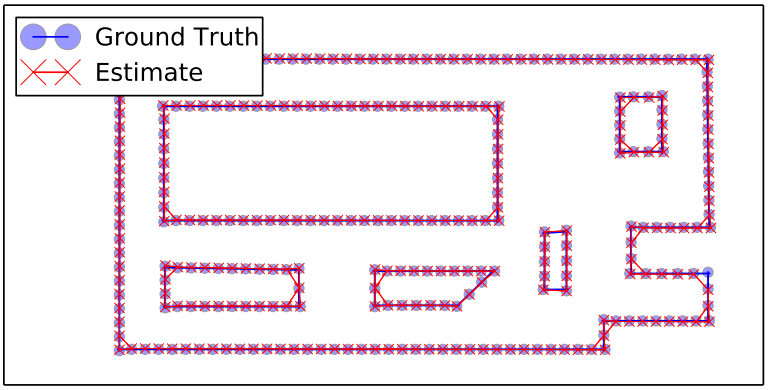
4.3.3 Laboratory Environment

The new framework is tested in an indoor space with the Turtlebot platform, using a computer with specifications listed in Table 4-3. The computational resources used are readily available in many modern on-board systems. The focus of this chapter is not on landmark detection or data association, thus AprilTags are used as landmarks in the indoor space. Figure 4-10 gives some example views of the environment. Figure 4-9 shows how the robot's mapping progressed throughout the experiment. It started with a partial map, then gradually picked up the frontiers and expanded the map to cover the space. The black lines are obstacles and black dots are landmarks. The red



(a) Gazebo simulation environment

(b) frontier exploration with grid map



(c) active SLAM via TFG

Figure 4-6: SLAM result comparison. When the odometry drifted, frontier exploration with occupancy grid map have distorted maps. While active SLAM using TFG is able close loops on landmarks and have much more accurate maps.

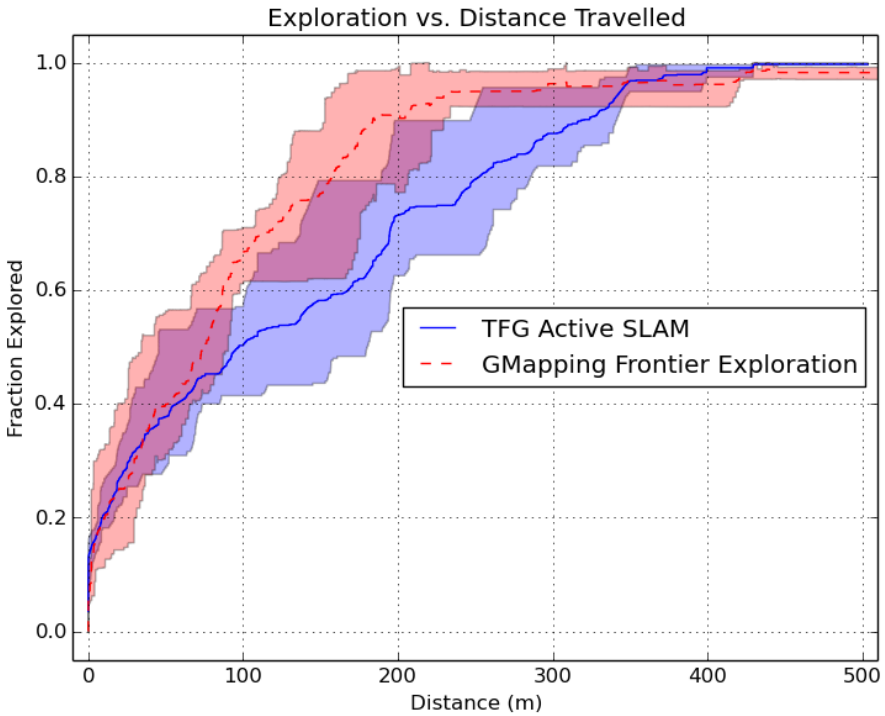


Figure 4-7: Map coverage vs distance travelled. TFG builds an accurate map while exploring and is thus slightly slower than greedy nearest-frontier exploration.

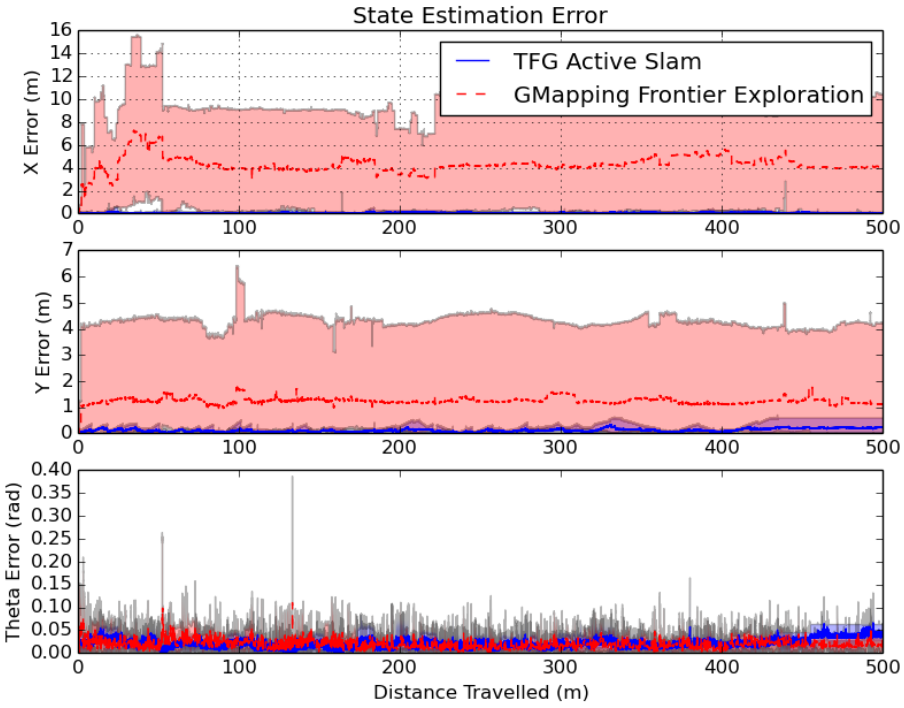


Figure 4-8: Robot pose error over multiple runs. Solid line repsents mean and shade represent range. TFG has consistently smaller errors.

Table 4-3: Hardware Specification

Robot	Turtlebot (Kobuki base)
Processor	Intel Core i3 dual @2.3GHz
RAM	4GB
Operating System	Ubuntu 14.04

dot is the robot's current position and the red lines are its planned trajectories.

4.4 Summary

This chapter proposed a Topological Feature Graph (TFG) that exploits the sparsity of graphical models, but also extended to further include obstacle representation. It lead to a unified information quantification metric to inherently balance between exploration and exploitation of map building. The resulting path planning algorithm enable the robot to autonomously generate trajectories to explore an unknown environment. Our approach significantly saves computation and memory resources compared to state-of-art occupancy grid (OG) map based approaches.

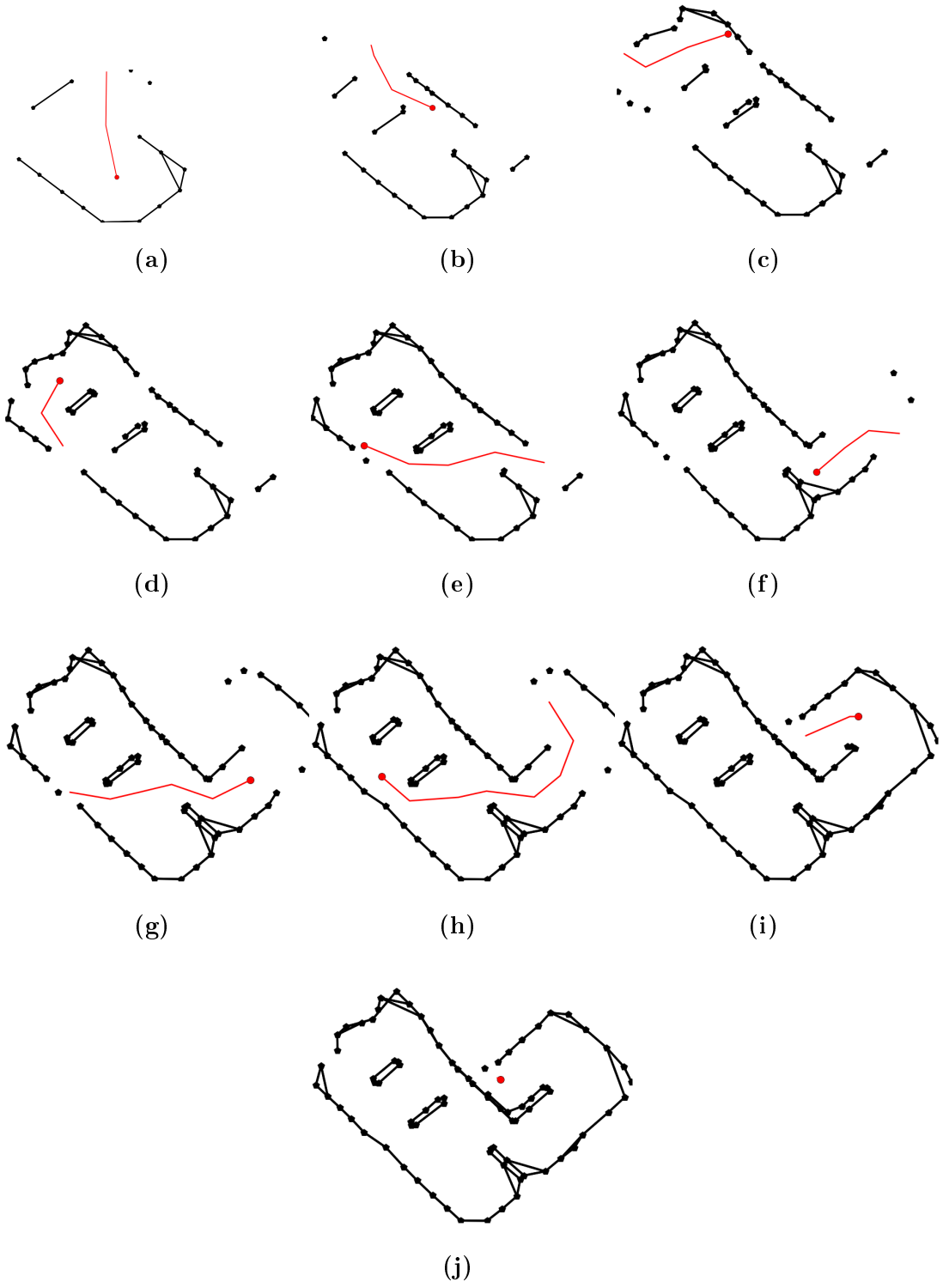


Figure 4-9: Robot path and TFG in hardware experiment. Black stars represent AprilTags, and black lines represent obstacles. The red circle represents the robot's current location, and the red line represents robot's planned trajectory. The robot started with a partial map, then gradually explores the frontiers and expands the map to cover the space.

Figure 4-10: Views of the space. An GPS-denied indoor environment with AprilTags as landmarks.

Chapter 5

Object Mapping via Non-parametric Pose Graph

5.1 Introduction

Approaches developed in Chapter 3 and 4 rely on the existence of uniquely identifiable landmarks. However, in natural environments, landmark detections typically have noise, and are not even unique. To enable SLAM in natural environments, a fundamental challenge is to recognize instances of landmarks, such as objects, and associate them with unique identifiers.

The focus of this chapter is performing SLAM in unknown environment by recognizing objects as landmarks (object SLAM). A factor graph is the natural representation, as objects can be easily represented as landmarks. A map represented by objects is desirable, also because objects are very rich in semantic meanings. By using objects, robots can interact with other agents and perform tasks at semantic level, such as searching for people in a forest, grasping objects, and detecting moving cars on streets. However, the convergence of factor graph SLAM algorithms relies heavily on correct data association of the landmarks. Even a single false association can cause the algorithm to diverge [15, 16]. Especially, when visual features are used as landmarks, feature detections vary a lot when viewpoint and lighting condition changes. Feature labeling and association becomes very challenging. Therefore, it is

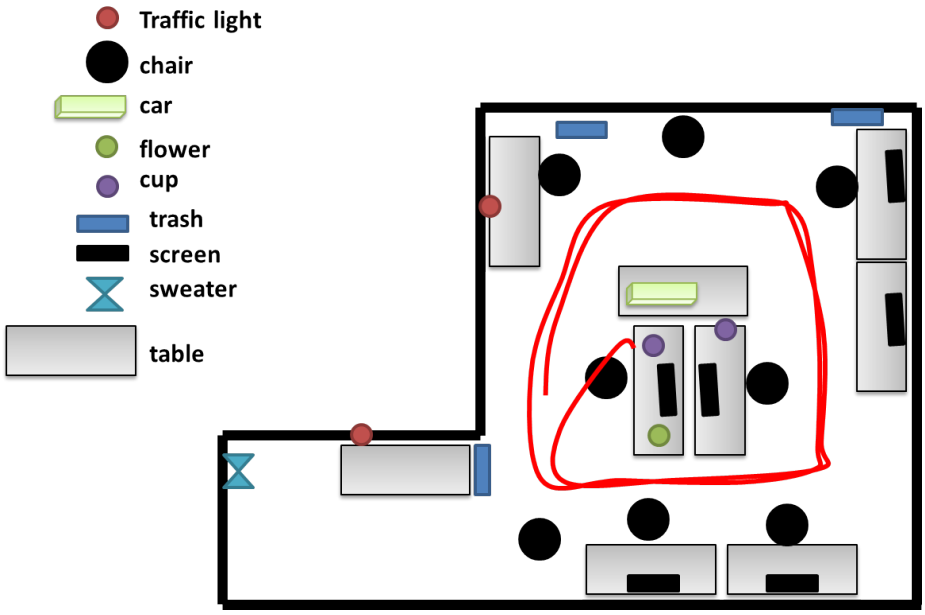


Figure 5-1: In object SLAM, each object class has multiple instances, data association (associate detect objects to unique object identifiers) is ambiguous. Data association and SLAM are inherently coupled: good data association guarantees the convergence of SLAM, and good SLAM solution gives good initialization of data association.

popular to do SLAM with visual features in open-loop fashion [90, 91].

Object SLAM requires the robot to be able to detect objects, generate measurements, and associate these measurements to unique identifiers. In this chapter, *object detection* refers to the problem of identifying the occurrence of objects of some predefined object classes within an image. An object measurement is a 3D location of the detected object with respect to the robot pose. *Data association* refers to the problem of associating object measurements to unique identifiers across images. The problem of object detection has been an important topic in the computer vision community. Deep learning approaches have achieved significant success on object detections within individual images [92–96]. These approaches also have the ability to generalize: once a detector is trained to recognize an object class, such as chairs, the detector can detect different instances of the same class even in different shape, color, and background settings. However, object detections only suggest the existence of objects of certain predefined object classes in an image, but provide no data association between images: given that an object of a certain class is detected in two images, the

object detector provides no information on whether or not the detections in the two images are the same object. This is problematic for SLAM especially when there are multiple objects of the same object class in an environment. How reliable SLAM can be achieved using only these ambiguous object detections remains an open question.

As illustrated in Figure 5-1, there are multiple instances of the same object class, such as chairs. The robot would need to establish the data association of object detections across images from different views. Note that data association and SLAM are inherently coupled problems: good data association guarantees the convergence of SLAM algorithms, and good SLAM solution gives good initialization of data association.

This chapter proposes a novel world representation, the nonparametric pose graph to jointly perform data association and SLAM. In the proposed model, factor graphs are used to localize robots and objects, while a Dirichlet process (DP) – a nonparametric model prior – is used to associate detections to unique object identifiers in the scene. The inference of the data associations and the optimization of the the robot and object poses are performed alternatively in this algorithm.

5.2 Object Measurements via Deep Learning

This section sets up the approach to generate object measurements via deep learning. The limitations of such an approach are discussed at the end of the section, which highlights the necessity of back-end data association and SLAM algorithm.

5.2.1 Deep Learning Based Object Detection

Object detection refers to the problem of identifying the existence of objects of certain classes and find bounding boxes for them in single images. Object detection in the past decade was mainly based on the use of SURF [97] and HOG features [98]. Although researchers have developed algorithms that demonstrated good performance for single class object detection (e.g. pedestrians), the multi-class object detection problem remains difficult. In particular, prior to 2012, the state-of-the-art method (deformable

part models) achieved 33.4% accuracy on the PASCAL VOC 2010 dataset [93], which contains 20 object classes.

Region-based convolutional neural network (R-CNN) by Girshick et al [93] first tried using deep learning methods to do object detection. This algorithm first uses the selective search [99] algorithm to propose bounding boxes in an image, which potentially contain objects. Each box is then subsequently scaled and fed into a CNN to detect the object class. This approach achieved 53.7% accuracy on the PASCAL VOC dataset. However, R-CNN is extremely slow (13 seconds per image) because all bounding boxes are fed into the same CNN sequentially. When there are significant overlapping between bounding boxes, there is significantly redundant computation on the same image.

In Faster R-CNN [100], Ren et. al. ran the full image through the CNN only once, and they only use features in top layer in each bounding box patch for object detection. They further proposed a region proposal network (RPN) that learns how to generate bounding box proposals by looking at the top layer features. This new algorithm achieves 76% accuracy and an average speed of 100 milliseconds per image.

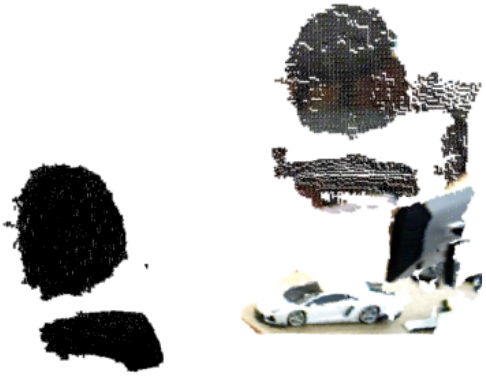
Faster R-CNN [100] uses the VOC dataset for training. Most of the object classes in the VOC dataset [92] are rare in urban or indoor settings, such as cows, horses, sheep, airplanes, and boats. Our work instead trained a faster R-CNN model on the ImageNet 2014 dataset [95], which contain categories that are more relevant to indoor/urban settings, including *cars, motorcycles, bicycles, traffic lights, televisions, chairs, flowerpots, cups, and keyboards*. Note that this framework can be easily modified to parse out any other subset of classes from the ImageNet dataset that are relevant to the specific applications.

5.2.2 Object Measurements

An object measurement refers to a 3D location with respect to the robot pose. To generate such measurements, location information relative to the robot is required in addition to object detection. In this chapter, this is done by inquiring the corresponding pixels in the depth images. The procedure to generate a 3D object measurements

(a) Object detection with RGB image.

(b) Corresponding depth image



(c) Object localization in 3D space

Figure 5-2: Deep learning based object detection

is outlined as follows:

1. Train a faster-RCNN to generate object detections in RGB images
2. Crop bounding boxes in the depth image in correspondence with the RGB bounding box.
3. Filter out background pixels that are too far away.
4. Generate point cloud from RGB and depth pairs.
5. Compute the centroid of the point cloud as center of the object.

Figure 5-2a shows the detected object with faster R-CNN from a single image of an office environment. Figure 5-2b shows the corresponding depth image, and Figure 5-2c shows the 4 point clouds for the 4 detected objects in 3D space.

It is clear from Figure 5-2a that object detection with deep learning faces two major challenges for pose graph based SLAM. First, there are multiple instances of the object class, such as "chair" in Figure 5-2a. Without correct data association, it is hard to distinguish different object instances. Standard pose-graph SLAM algorithms can only optimize poses with exact data association, such as `g2o`, `iSAM`, and `GTSAM`. The second challenge is high false positive rates. As the `chair` detected in Figure 5-2a, deep learning algorithms report objects now and then when there are actually none. Blindly using these unfiltered detections in standard SLAM algorithms will lead to creation of landmarks that do not correspond to any real-world objects and cause loop closure failures.

Notice that the centroid is used as the center of objects in this case. When objects are looked at from different views, and be partially occluded, centroids would not be a consistent measure of the object locations. In our experience, the error could be 10-20cm. However, we will show that in office settings, our algorithm still converges even under this significant occlusion and view point noise.

5.3 Nonparametric Pose Graph

This section sets up the joint data association and SLAM problem by extending the current pose graph to a novel nonparametric pose graph that tightly couples object association with robot poses. A new algorithm is also introduced to jointly infer the data association and perform SLAM with this new model.

5.3.1 Factor Graph with Known Data Association

Given a batch dataset, recall that a sequence of time indexed random variables $\mathbf{X}_{0:T} = \{X_0, \cdots, X_T\}$ is used to denote the robot's trajectory from the start to the end. These poses are not directly observable and need to be established. The robot can always measure the incremental change between two sequential poses via an IMU or wheel encoder, which is referred to as odometry. Denote o_t as the odometry measurement between pose X_t and pose X_{t-1}.

During navigation, the robot also observes various objects from the environment. We assume that the environment is static, so objects neither change positions nor shapes:

Assumption 4. *Objects in the environment are static: they do not change positions or shapes.*

Denote the positions of the objects as $\mathbf{L} = \{L_1, \cdots, L_M\}$. These variables are unknown and need to be established. At time t, the robot obtains K_t object measurements, denoted as $\mathbf{z}_t = \{z_t^1, z_t^2, \cdots, z_t^{K_t}\}$. The unique object identifiers these measurements are associated to are $\mathbf{y}_t = \{y_t^1, y_t^2, \cdots y_t^{K_t}\}$, where $y_t^i \in \{1, \cdots, M\}$. When object associations are perfectly known, $\mathbf{y}_t$ is known when the measurements $\mathbf{z}_t$ are obtained.

Recall from section 2.2 that using standard model that odometry and object measurements are corrupt by Gaussian noise, the joint log likelihood can be represented by a factor graph:

$$\log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}) = \sum_{t=1}^{T} \phi(o_t; X_{t-1}, X_t) + \sum_{t=0}^{T} \sum_{k=1}^{K_t} \phi(z_t^k; X_t, L_{y_t^k}) \tag{5.1}$$

where $\phi(o_t; X_{t-1}, X_t)$ represents a odometry factor and $\phi(z_t^k; X_t, L_{y_t^k})$ represents an object measurement factor. See equations (2.10), (2.12) and (2.15).

The pose graph SLAM problem optimizes robot poses $\mathbf{X}_{0:T}$ and object locations $\mathbf{L}$ such that the log likelihood is maximized

$$\max_{\mathbf{X}_{0:T}, \mathbf{L}} \log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}). \tag{5.2}$$

Standard SLAM solver packages solve Equation (5.2) with nonlinear optimization techniques such as gradient descent [13]. When the sparsity of the graph model is used, the optimization can be done very efficiently.

5.3.2 Factor Graph with Multi-class Objects

Before we move into nonparametric factor graph for imperfect data association, first notice in object SLAM, except for measuring the 3D location of objects, we also observe an object class. The observed object class is not always reliable, thus we first establish the probabilistic model for object classes.

Assume there are N object classes in total. For object i, denote u as an observation of the object class. The probability of u is modeled with a Categorical distribution:

$$p(u = j) = \pi_i(j), \quad j = 1, \cdots, N \tag{5.3}$$

Denote $\pi_i = \{\pi_i(0), \cdots, \pi_i(N)\}$. And $\sum_{n=0}^{N} \pi_i(n) = 1$. And if the true object class is j, we have $\pi(j) \gg \pi(k)$ for $k \neq j$.

Notice class observation u can only be any real object classes $1, \cdots, N$. However, we especially design $\pi_i(0)$ to represent the probability of false positives. This design would help the algorithm to filter non-exist object detections in real-world experiments.

In order to have closed form updates on posterior, we apply Dirichlet prior to π_i for object i:

$$\pi_i \sim \mathrm{Dir}(\beta_i). \tag{5.4}$$

when there is an observation of class j, $u = j$, the posterior distribution of π_i is:

$$\pi_i | u \sim \mathrm{Dir}(\beta_i + e_j). \tag{5.5}$$

where e_j represents a unit vector with jth element to be 1.

Notice $\beta_i(0)$ represents the initial likelihood of object i to be a false positive. Since observations cannot be 0, when there are more and more observations of object i being observed, the posterior $\beta_i(0)$ will monotonically decrease. This is consistent with the intuition that if repeated observations are obtained from some object, then it has lower chance to be a false positive.

Combine the multi-class probabilistic setting with the original SLAM problem: each object measurement would be a pair $\{z_t^k, u_t^k\}$, where continuous variable z_t^k represents the 3D location measurement, and discrete variable u_t^k represents the observed object class. Recall that $y_t^k = i \in \{1, \cdots, M\}$ represents that the k-th measurement at time t is from object i. Then u_t^k is a sample from the posterior distribution $\pi_{y_t^k}$.

$$p(u_t^k = j) = \pi_{y_t^k}(j), \quad j \in \{1, \cdots, N\} \tag{5.6}$$

The joint log likelihood becomes:

$$\begin{aligned} &\log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}, \mathbf{u}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}) \qquad (5.7)\\ &= \sum_{t=1}^{T} \phi(o_t; X_{t-1}, X_t) + \sum_{t=0}^{T} \sum_{k=1}^{K_t} \Big(\phi(z_t^k; X_t, L_{y_t^k}) + \log \pi_{y_t^k}(u_t^k) \Big) \\ &= \sum_{t=1}^{T} \phi(o_t; X_{t-1}, X_t) + \sum_{t=0}^{T} \sum_{k=1}^{K_t} \phi(z_t^k; X_t, L_{y_t^k}) + \sum_{t=0}^{T} \sum_{k=1}^{K_t} \log \pi_{y_t^k}(u_t^k) \qquad (5.8) \end{aligned}$$

The new optimization problem is then

$$\max_{\mathbf{X}_{0:T}, \mathbf{L}, \pi} \log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}, \mathbf{u}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}, \pi). \tag{5.9}$$

Compared to (5.2), in problem (5.9), the observed data further includes object class observations $\mathbf{u}_{0:T}$, and the variables to be estimated further include the class of objects π. From (5.7), given data association $\mathbf{y}_{0:T}$, the joint likelihood can be factorized into the sum of likelihood of $\mathbf{z}_{0:T}$ and $\mathbf{o}_{0:T}$, and the likelihood of $\mathbf{u}_{0:T}$. Therefore, the class classes $\pi_{0:T}$ is independent of the robot poses $X_{0:T}$ and object positions $\mathbf{L}$. Optimizing (5.9) is equivalent to solving problem (5.2) and computing the object class posterior π independently.

5.3.3 Nonparametric Graph Definition

Now we move to the case that the data association y_t^k is unknown and must be established. Deep learning-based algorithms label each object to be of some class, but

do not distinguish between different objects of the same class. When there are multiple instances of the same object class, such as multiple chairs in a room, possibilities for data association become combinatorial and thus challenging. Instead of relying on a reliable front-end procedure to associate objects, we use a back-end framework to jointly infer the data association and object locations. Note that because of the ambiguous data association, the total number of objects M is unknown ahead of time either, and needs to be established as well.

Nonparametric models are a set of tools that adapt the model complexity to data. It has the embedded mechanism that the model parameters could grow when there are new data being observed. In particular, Dirichlet Process (DP) is such a nonparametric stochastic process that models discrete distributions but with flexible parameter size. It can be taken as the generalization of a Dirichlet distribution with infinite dimension. Same as Dirichlet distribution is the conjugate prior for a categorical distribution, DP can be viewed as the conjugate prior for infinite, nonparametric discrete distributions [101]. In this work, we use a Dirichlet Process (DP) as the prior for data associations y_t^k. In particular, assume at any point, there are M objects being detected in total, the probability of y_t^k belongs to object i:

$$p(y_t^k = i) = \mathrm{DP}(i) = \begin{cases} \frac{m_i}{\Sigma_i m_i + \alpha} & 1 \leq i \leq M, \\ \frac{\alpha}{\Sigma_i m_i + \alpha} & i = M + 1. \end{cases} \tag{5.10}$$

where m_i is the number of measurements of object i, and α is the concentration parameter of DP prior that determines how likely it is to create a new object. The intuition behind this model is that the probability y_t^k is from some existing object $i \leq M$ is proportional to the number of measurements of object i, and the probability y_t^k is from a new object $M + 1$ is proportional to α.

The joint log likelihood of odometry $\mathbf{o}_{0:T}$, object measurement $\mathbf{z}_{0:T}$ and object classes $\mathbf{u}_{0:T}$ given data association $\mathbf{y}_{0:T}$ is

$$\log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}, \mathbf{u}_{0:T}; \mathbf{X}_{0:T}, \mathbf{y}_{0:T}, \mathbf{L}, \pi) \tag{5.11}$$

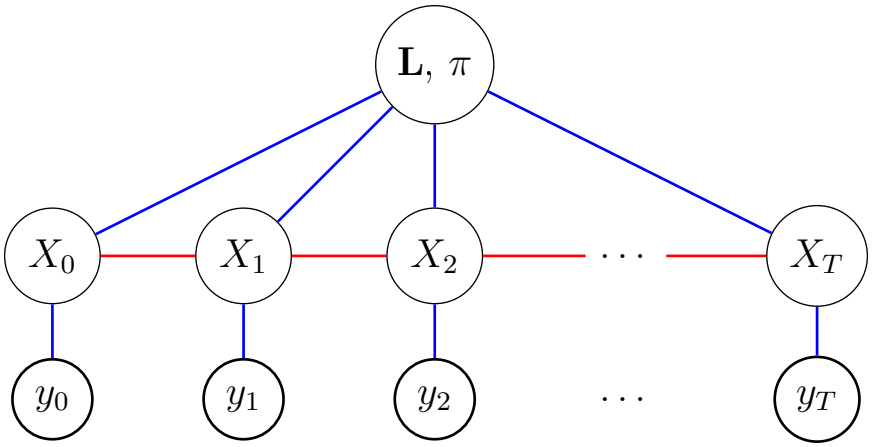


Figure 5-3: Factor graph for SLAM with imperfect data association. y_t represents the data association: the measurement at time t is from object y_t. In SLAM with imperfect data association, y_t is unknown and must be established at the same time.

$$= \sum_{t=1}^{T} \phi(o_t; X_{t-1}, X_t) + \sum_{t=0}^{T} \sum_{k=1}^{K_t} \left(\pi_{y_t^k}(u_t^k) + \phi(z_t^k; X_t, L_{y_t^k}) \right).$$

The joint log likelihood (5.11) has the same form as (5.7). However, in (5.11), the likelihood of object measurements $\mathbf{z}_{0:T}$ and object classes $\mathbf{u}_{0:T}$ are correlated through data association $\mathbf{y}_{0:T}$.

The new optimization problem is then

$$\max_{\mathbf{X}_{0:T}, \mathbf{L}, \mathbf{y}_{0:T}, \pi} \log p(\mathbf{o}_{1:T}, \mathbf{z}_{0:T}, \mathbf{u}_{0:T}; \mathbf{X}_{0:T}, \mathbf{L}, \mathbf{y}_{0:T}, \pi). \tag{5.12}$$

Compared with Equation (5.9), the new optimization problem Equation (5.12) is more challenging in that data associations $\mathbf{y}_{0:T}$ are unknown. As a result, log probabilities of object measurements Equation (5.11) no longer have a simple form, and the problem Equation (5.12) becomes a mixed integer nonlinear problem. Secondly, the number of true objects in the environment M is not necessarily known a priori, problem Equation (5.11) must infer M at the same time.

5.3.4 Nonparametric SLAM

From the last section, for $t = 1, \cdots, T, k = 1, \cdots, K_t$, the generative model for our problem is

$$y_t^k \sim \mathrm{DP}(\alpha), \tag{5.13a}$$

$$\pi_{y_t^k} \sim \mathrm{Dir}(\beta_{y_t^k}), \tag{5.13b}$$

$$o_t \sim \mathcal{N}(X_t \ominus X_{t-1}, Q), \tag{5.13c}$$

$$u_t^k \sim \mathrm{Cat}(\pi_{y_t^k}), \tag{5.13d}$$

$$z_t^k \sim \mathcal{N}(L_{y_t^k} \ominus X_t, R), \tag{5.13e}$$

where α, β, Q, and R are given parameters. Robot poses $\mathbf{X}_{0:T}$, landmark locations $\mathbf{L}$, object class distributions $\pi_{1:M}$ and object associations $\mathbf{y}_{0:T}$ are variables to be estimated. The odometry $o_{1:T}$ and object measurements $\mathbf{z}_{0:T}, \mathbf{u}_{0:T}$ are observed data.

Different from a canonical DP mixture model, the observed data $\mathbf{z}_{0:T}$, $\mathbf{u}_{0:T}$, and $\mathbf{o}_{0:T}$ are not independent samples given variables $\mathbf{X}_{0:T}$, $\mathbf{L}$, and π, but are correlated through the factor graph. Therefore, the inference involves computing maximum likelihood over factor graphs. When both associations and variables are to be established, standard approaches alternate between assigning data and optimizing variables. In the case of known object number M, K-means has a deterministic data association, while expectation-maximization associates data in a probabilistic way [69]. When the number of objects is not known a priori and DP is used as prior, Markov Chain Monte Carlo methods (e.g. Gibbs sampling) or variational inference algorithms are widely used [69]. However, in these algorithms, the likelihood of each label y_t^k to be any underlying object $\mathbf{L}$ needs to be computed and tracked all the time. The algorithm will need to go through all of the data multiple times to converge to a steady state distribution. The large scale and strong dependence of data in our problem make such approaches inappropriate.

It is shown in [102] that under the small variance assumptions, Gibbs sampling can be simplified to DPmeans. Instead of sampling the posterior distribution, y_t is

assigned to be the maximum likelihood object if the likelihood is within some certain threshold, otherwise it is assigned to a new object. Intuitively, in this case, small variance means that the noise in odometry, object measurement and object class is relative small, so that the posterior distribution of y_t is peaky.

Assumption 5. *Variance in odometry, object measurement and object class is small, so that the posterior distribution of data association has small variance and a unique maximal likelihood value.*

The DPmeans algorithm alternates between two steps: maximize likelihood on variables $\mathbf{X}_{0:T}, \mathbf{L}, \pi$, and assign data association $\mathbf{y}_{0:T}$ to their maximum likelihood objects. Algorithm 5-1 shows the overall flow of the approach. And the following explains the algorithm step by step.

Initialization (line 1) In initialization, all y_t^k are set to be an object by its own. Robot poses $\mathbf{X}_{0:T}$ and object locations $\mathbf{L}$ are initialized by their open loop estimation. The Dirichlet distribution prior for object class are set to be some initial value β_0.

Optimizing data association (line 3) While executing the main loop, the algorithm alternates between optimizing associations $\mathbf{y}_{0:T}$, and variables $\mathbf{X}_{0:T}$, $\mathbf{L}$, and β. When optimizing object association, fix $\mathbf{X}_{0:T}$, $\mathbf{L}$ and β, and compute the posterior of y_t^k as the product of its DP prior (5.10) and likelihood of measurements (u_t^k, z_t^k) (see (2.10) and (5.6)).

$$p_i \propto \mathrm{DP}(i) p(u_t^k; \pi_i) p(z_t^k; X_t, L_i). \tag{5.14}$$

Then y_t^k is assigned to the maximum likelihood object

$$y_t^k = \arg\max_i p_i. \tag{5.15}$$

Optimizing poses (line 10) When optimizing poses, object associations y_t^k are fixed. The posterior parameters for the Dirichlet distribution of object class can be

updated with

$$\beta_i(j) \leftarrow \beta_0(j) + \sum_{t,k} \mathbb{I}_{y_t^k=i} \mathbb{I}_{u_t^k=j}, \tag{5.16}$$

where β_i is the hyperparameter for the Dirichlet prior on π_i. Notation $\mathbb{I}_{a=b}$ represents indicators whether quantity a equals quantity b. Then $\sum_{k,t} \mathbb{I}_{y_t^k=i}$ is the total number of object detections assigned to object i, and $\sum_{k,t} \mathbb{I}_{y_t^k=i} \mathbb{I}_{u_t^k=j}$ represents from the detections of object i, how many are class j. With Dirichlet prior $\mathrm{Dir}(\beta_i)$, the maximum likelihood(ML) of each object class i is proportional to parameters β_i:

$$\pi_i = \mathrm{ML}(\mathrm{Dir}(\beta_i)). \tag{5.17}$$

The maximum likelihood value of robot poses $\mathbf{X}_{0:T}$ and object locations $\mathbf{L}$ can then be obtained by standard SLAM solvers (see (5.2)).

Remove false positive (line 18) Recall that we set $\pi_i(0)$ to be the probability that object i is a false positive. In initialization, $\beta_i(0)$ is set to be some positive number. When new measurements of object i are obtained and accumulated, β_i gets updated such that $\beta_i(j)$, $j > 0$ becomes bigger compared to $\beta_i(0)$. As a result, $\pi_i(0)$ decrease monotonically. In the last step, we filter out false positives by simply putting a threshold on $\pi_i(0)$.

5.3.5 Complexity Analysis

In this section, we analyze the complexity of the two major steps in Algorithm 5-1: optimizing data association $\mathbf{y}_{0:T}$, and solving SLAM $\mathbf{X}_{0:T}$, $\mathbf{L}$.

Optimizing data association This step goes through all the measurements, therefore the complexity is linear in the total number of measurements $\mathcal{O}(|\mathbf{y}_{0:T}|)$. For each measurement, we compute the posterior probability of belong to any existing object, or being a new object. The complexity is linear in the total number of existing objects $\mathcal{O}(M)$. Therefore the overall complexity is $\mathcal{O}(|\mathbf{y}_{0:T}|M)$.

Algorithm 5-1 Nonparametric SLAM

Input: Odometry measurements $o_{1:T}$, Object measurements $\mathbf{u}_{0:T}$, $\mathbf{z}_{0:T}$

Output: Poses $\mathbf{X}_{0:T}$, number of objects M, object association $\mathbf{y}_{0:T}$, object locations and classes $\mathbf{L}$, β

Initialize $\mathbf{X}_{0:T}$, $\mathbf{L}$ with open loop prediction, initialize $\beta_i = \beta_0$. Initialize each y_t^k to be an object of its own
while not converged **do**
    Fix $\mathbf{X}_{0:N}$, $\mathbf{L}$, β
    for Each measurement y_t^k **do**
        Computer posterior p_i of being object i:
            $p_i \propto \mathrm{DP}(i) p(u_t^k; \pi_i) p(z_t^k; X_t, L_i)$
        Assign y_t^k to be maximum likelihood association:
            $y_t^k = \arg\max_i p_i$
    end for
    Fix $\mathbf{y}_{0:T}$
    for each object i **do**
        update class π:
            $\beta_i(j) \leftarrow \beta_i(j) + \sum_{t,k} \mathbb{I}_{y_t^k = i} \mathbb{I}_{u_t^k = j}$
            $\pi_i = \mathrm{ML}(\mathrm{Dir}(\beta_i))$
    end for
    optimize $\mathbf{X}_{0:T}$, $\mathbf{L}$ with standard SLAM solver with (5.2)
end while
Remove false positive
    $\forall i$, delete object i if $\pi_i(0) > \epsilon$

Notice each measurement is assigned to be an object by itself in initialization, in which case $M = |\mathbf{y}_{0:T}|$, thus the total complexity has an upper bound of $\mathcal{O}(|\mathbf{y}_{0:T}|^2)$. However, when the algorithm converges to fewer and fewer objects, we have $M \ll |\mathbf{y}_{0:T}|$. And in the end, M converges to the true number of objects in the scene.

Optimizing poses Updating the Dirichlet distribution parameters β and class probabilities π involves counting the assignments of y_t^k. Therefore its complexity is $\mathcal{O}(|\mathbf{y}_{0:T}|)$. Optimizing poses $\mathbf{X}_{0:T}$ and $\mathbf{L}$ can make use of standard SLAM solvers, which is polynomial in the number of robot poses $|\mathbf{X}_{0:T}|$ and the number of objects M.

To sum up, the algorithm is polynomial in the number of robot poses $\mathbf{X}_{0:T}$, the number of objects M and the number of measurements $\mathbf{y}_{0:T}$.

Table 5-1: Simulated Dataset Overview

Distance Traveled	72.7m
field of view	4m, 120 degree
no. of odometry measurements	766
no. of object measurements	1098
odometry noise	$\mathcal{N}(0, 0.02^2)$
measurement noise	$\mathcal{N}(0, 0.1^2)$

5.4 Experiment

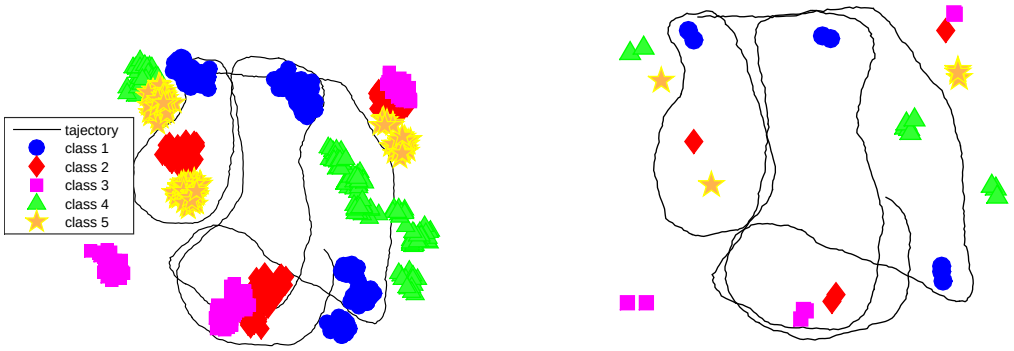
5.4.1 Simulation

In the simulation, 15 objects are randomly generated in a 2D plane. They are randomly assigned into 5 different object classes: red diamonds, blue circles, green triangles, yellow stars, and magenta squares. The robot trajectory is manually designed and passes through the environment several times. Figure 5-5a shows the ground truth of the generated dataset. At each pose x_t, the robot observes the relative position o_t^k and class u_t^k of the objects that are within its field of view. Gaussian noise is added to the odometry measurements as well as object measurements, see (2.10) and (2.12). The parameters of the dataset are listed in Table 5-1.

Figure 5-4a shows the object predictions purely based on open-loop odometry. There is significant amount of variance and drift in the distribution of these predicted object locations, which obscures the determination of exactly how many objects there actually are in the environment. The result after the first iteration is shown in Fig. 5-4b; the nonparametric pose graph clusters the measurements and uses it to correct robot poses. The total number of objects is reduced to 33. The result after the second iteration is shown in Fig. 5-4c; the algorithm further reduces the total number of objects to 20. After three iterations(Fig. 5-4d), the algorithm converges to the true underlying number of objects, which is 15.

The performance of the proposed nonparametric graph (NP-Graph) is compared to three existing methods:

1. *Frame by frame detection (FbF)*: each object in each frame is taken as new, and



(a) Iteration 0, 1098 objects

(b) Iteration 1, 33 objects

(c) Iteration 2, 20 objects

(d) Iteration 3, 15 objects

Figure 5-4: Result of nonparametric pose graph at different iterations. Initially there are 1098 object detections. The number reduces to 33 after the first iteration, reduces to 20 after the second iteration, and converges to the ground truth 15 after 3 iterations.

there are neither SLAM nor data association (see Figure 5-4a).

2. *Open-loop Object Detection (OL) [55]*: use robot odometry to perform data association across images, but do not use data association results to correct robot poses (see Figure 5-5c).

3. *Robust SLAM (R-SLAM) [16]*: back-end algorithm that finds the maximal set of consistent measurements, but eliminate inconsistent measurements (see Figure 5-5b).

Figure 5-5 and Table 5-2 compare the SLAM performance of four different algorithms. Figure 5-6 shows the cumulative position error of the robot trajectory. Figure 5-7 compares the number of objects identified and their localization error.

Table 5-2: Performance Comparison on Simulated Dataset

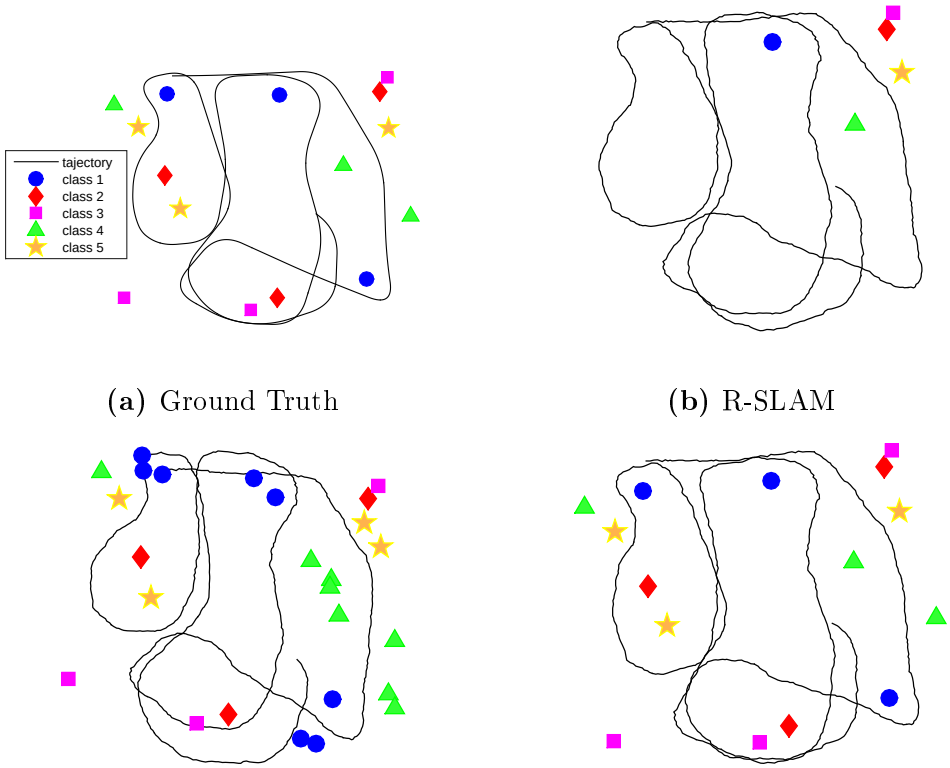
	mean pose error	cumulative trajectory error	percent of measurements used	number of objects	mean object error
NP-Graph	0.07	55.1	100	15	0.05
OL	0.42	320.6	100	39	0.39
R-SLAM	0.20	150.5	20.2	5	0.20
FbF	0.42	320.6	100	1098	0.49

FbF and OL purely rely on odometry and do not correct robot poses, therefore have the biggest error. R-SLAM uses a subset of object measurements to close loops on robot poses, thus the error is smaller. Our NP-graph based approach make use of all the object measurements, thus has the smallest error on both robot poses and object positions. FbF does not do any data association, thus significantly over estimate the number of objects. The OL approach does not optimize robot poses. When the robot comes back to a visited place, the odometry has drifted significantly thus the OL approach could not associate the objects to the same one observed before. As a result, the OL approach also over estimate the total number of objects. R-SLAM only keeps one set of consistent measurements for each object classes, therefore it is only able to detect one instance for each object class, and significantly underestimate the total number of objects. NP-Graph, on the other hand, utilize all of the object measurements and jointly infers both robot poses and the data associations, thus can correctly infer the right number of objects.

5.4.2 Office Environment

To test the performance in real-world scenarios, we collected a dataset of an office environment and used deep learning to detect objects, such as chair, screen, cups etc. The statistics about the office dataset is shown in Table 5-3.

Table 5-4 and Figure 5-8 compare the performance of FbF, R-SLAM, PL and our approach NP-Graph. While the ground truth for object positions is not available for this dataset, we compare the performance on the number of valid objects, the number of inlier measurements and the variance on object positions. An object is



(a) Ground Truth **(b)** R-SLAM

(c) OL **(d)** NP-Graph

Figure 5-5: Simulation. Black line represents the robot trajectory. Each marker color/shape represent an object class. FbF does neither data associate nor SLAM. OL associate object detection across images but does not optimize robot poses. R-SLAM only uses a subset of consistent object measurements to optimize robot poses. Our approach NP-graph optimizes both robot poses and data association.

Table 5-3: Office Dataset

image resolution	640×480
distance traveled	28.06m
during	167s
no. of odometry	696
no. of objects	30
no. of object detections	1588
odometry noise	$\mathcal{N}(0, 0.1))$
measurement noise	$\mathcal{N}(0, 0.5)$

defined as valid when its false positive probability $\pi_i(0)$ is below a threshold ($\epsilon = 2\%$), otherwise it is marked as a false positive. A measurement is denoted as an inlier when it is associated with a valid object. The object variance is determined from the

Table 5-4: Performance Comparison on Office Dataset

	percentage of measurement inliers	number of inlier objects	number of false positive objects	variance on objects
NP-Graph	88.0	31	88	0.058
OL	82.2	36	175	0.121
R-SLAM	22.5	7	0	0.225
FbF	0	0	1588	-

uncertainty in the predicted location of the object from its associated measurements. From Table 5-4, the NP-Graph has the highest percentage of inlier measurements, the closest number of objects to truth, and the smallest variance on the object locations.

While the ground truth for robot poses is not available, either, we compare the performance qualitatively. Figure 5-1 shows the floor map of the environment as well as the robot trajectory. Figure 5-8 compares the results of 4 approaches. FbF and OL estimation are open-loop approaches and over estimate total number of objects. R-SLAM only uses a subset of the object measurements. It can only identify one instance for each object class, and has bad estimates even it closes loops on robot poses. On the other hand, NP-Graph is able to close loops on robot poses and recover the turnings at corners. While there is no ground truth in the office dataset for computing object localization errors, it is worth noting that there is a sweater hanging on the shelf in the far bottom left corner, our algorithm is able to recover its distance while other approaches failed to.

Figure 5-9 shows a few examples of the detected and well associated objects, which includes chair, screen, keyboard, toy car and the sweater hanging in the back corner. These figures are extracted from point cloud of a single bounding box that is associated to the corresponding object. Note that these point clouds are only for illustration purposes, but not maintained in the algorithm. The algorithm only uses the centroid of these point clouds as object measurements.

5.5 Summary

When interacting with natural environments through object recognition, the robot faces the extra challenge of ambiguous data association. This chapter proposed a novel nonparametric pose graph that tightly couples the data association and SLAM problem. An inference algorithm is further developed to alternative between inferring data association and performing SLAM. Both simulated and real-world datasets show that our new approach has the capability of doing data association and SLAM simultaneously, and achieves better performance on both associating object detections to unique identifiers and localizing objects.

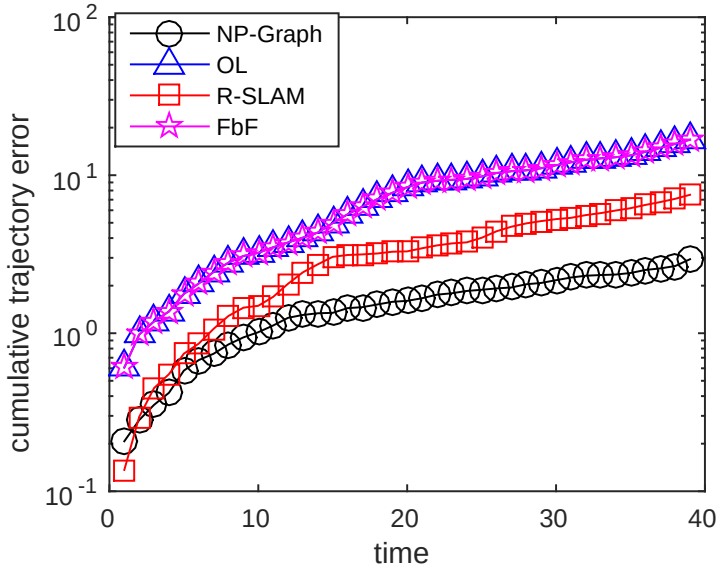


Figure 5-6: Cumulative robot pose error along the trajectory. NP-graph has significantly less error than other approaches.

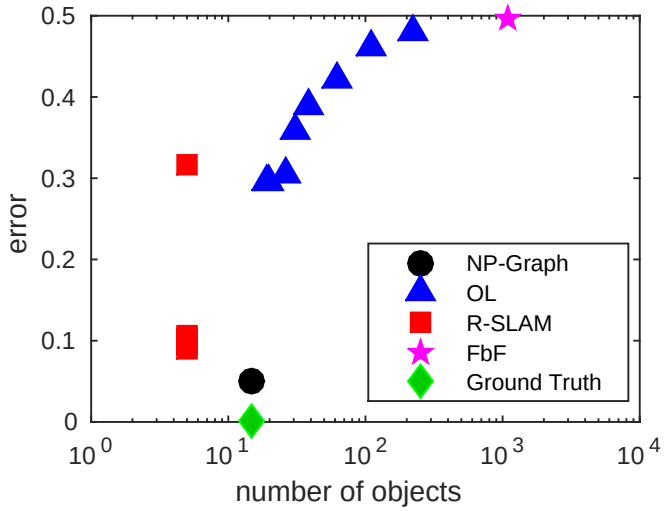
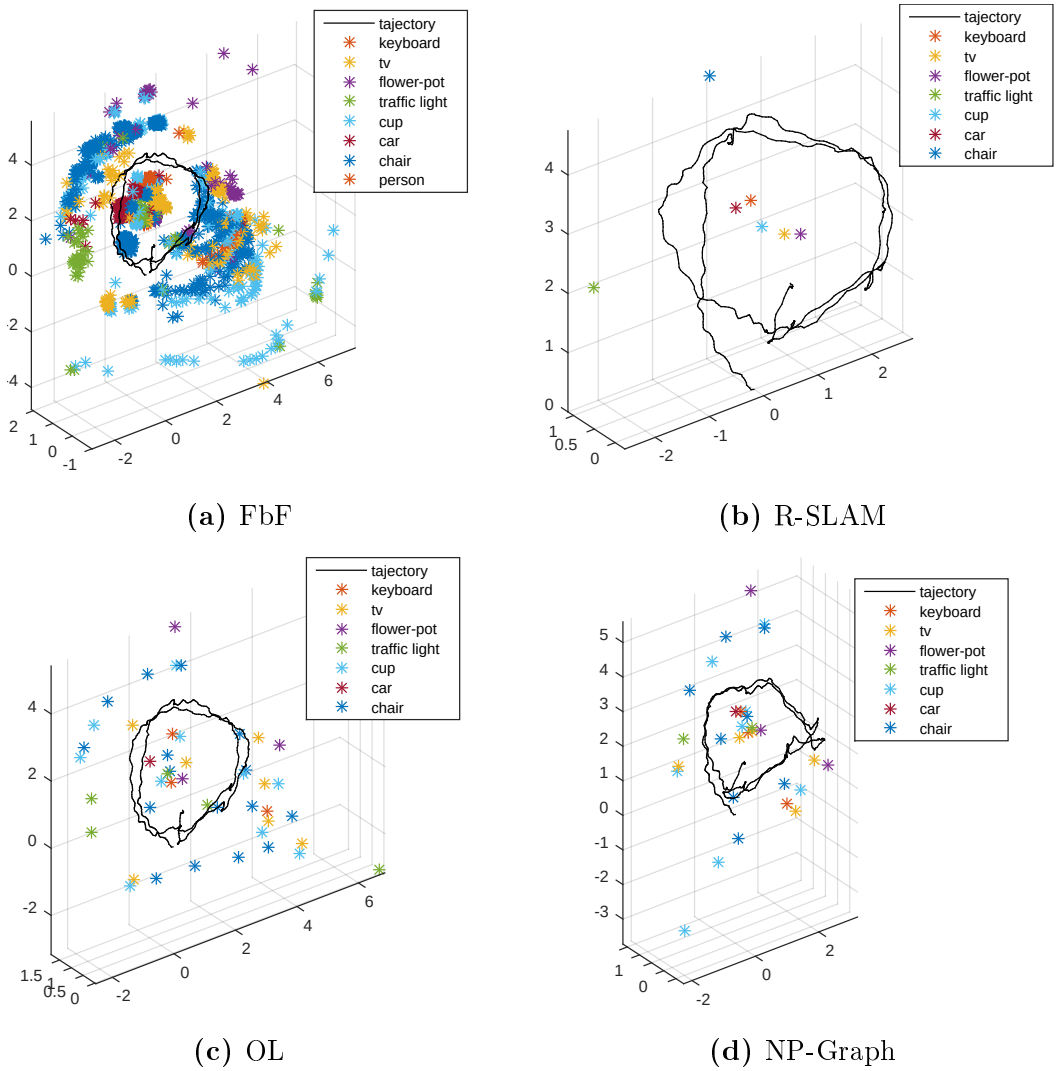


Figure 5-7: Comparison of number of objects and mean error on objects. Both FbF and OL have big error on object positions and overestimate the number of objects. R-SLAM has much smaller error on object positions, but underestimates number of objects. NP-graph recovers the true number of objects and has the least error.



(a) FbF

(b) R-SLAM

(c) OL

(d) NP-Graph

Figure 5-8: Office Dataset. Black line represents estimated robot trajectory. Markers represent objects. Each color represent an object class. FbF and OL over estimate number of objects. R-SLAM has bad estimate even it closes loops on robot poses. NP-Graph has the closest estimate of number of objects, recovers corners in robot trajectory.

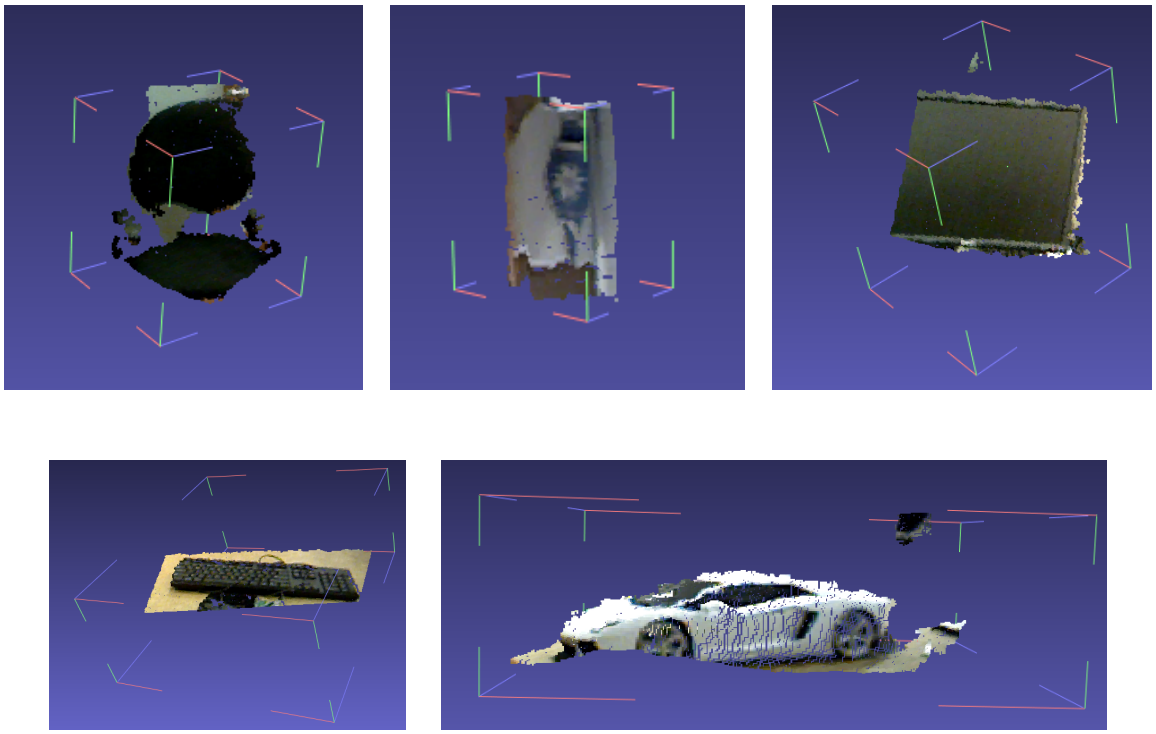


Figure 5-9: Example of detected objects, plotted from a single frame point cloud. From left to right, top to down are chair, sweater in the corner, screen, keyboard and toy car.

Chapter 6

Conclusions

6.1 Contributions

When a robot is obtaining large streams of sensing data but only has constrained resources, it is important to selectively process data and build sparse models. In case that the robot has specific tasks, the data to retain and model to build should be directly useful for the robot tasks. This thesis quantifies the usefulness of data and builds sparse models that can still achieve good navigation and mapping performance.

This thesis first presented a two-stage landmark and measurement selection procedure for resource-constrained robots operating in unknown environments. We specifically focused on the navigation task, but there are many inference and planning tasks that require a similar prioritization of variables. In this work, we provide a theoretically sound basis for selecting measurements to localize important variables preferentially. This is an important capability for many resource-constrained real-time systems. Simulations and hardware results demonstrate that the approach can identify a relevant subset of landmarks and accurately localize them to reduce the colliding probability with obstacles as compared to existing SLAM reduction approaches. As a result, the robot is able to navigate the environment for long periods of time, without the memory or computational requirements growing beyond the constraints.

To enable the robot to actively plan its own trajectories while mapping the environment, this thesis further proposed a purely graph-based active SLAM algorithm.

A novel Topological Feature Graph (TFG) is created that extends pose graphs for SLAM to include geometry representation of the obstacles. An information objective is used that directly quantifies uncertainty of a TFG. It captures correlations between robot poses and landmarks under a unified framework, thus new feature observations can help close loops and reduce uncertainties on observed landmarks. The exploration and exploitation naturally comes out of the framework for a given landmark density. An efficient sampling-based path planning procedure is developed within the TFG, which enables active SLAM. Experiments showed that the new approach is able to actively explore an environment with much less resources than existing occupancy grid based approaches.

To enable the robot interact with natural environments, the third part of the thesis utilize objects detected via deep learning as landmarks. Object SLAM is challenging as data association is ambiguous while their locations need to be established at the same time. A novel nonparametric pose graph was developed that tightly couples data association and SLAM problems. An algorithm is developed to alternate between inferring data association and performing SLAM. Both simulated and real-world datasets show that our new approach has the capability of doing data association and SLAM simultaneously, and achieves better performance on both associating object detections to unique identifiers and localizing objects than existing approaches.

6.2 Future Research

In object SLAM, objects are used as landmarks for navigation. However, it does not include obstacle representation. Future work would investigate the extension of the object-SLAM framework to represent obstacles explicitly. For example, it may be possible to use visual features to perform a triangulation of the obstacle surfaces that are visible to the robot. With such an obstacle representation, the robot may be able to achieve fully autonomous path planing and sparse mapping in unmodified environments.

Bibliography

[1] A. Elfes. Occupancy grids: A stochastic spatial representation for active robot perception. In *Sixth Conference on Uncertainty in AI*, pages 7–24, 1990.

[2] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, june 1989.

[3] S. Thrun. Learning occupancy grids with forward sensor models. *Autonompous Robots*, 15:111–127, 2003.

[4] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. The MIT press, Cambridge, Massachusetts, USA, 2005.

[5] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, Cambridge, MA, USA, 2005.

[6] Benjamin Charrow, Gregory Kahn, Sachin Patil, Sikang Liu, Ken Goldberg, Pieter Abbeel, Nathan Michael, and Vijay Kumar. Information-theoretic planning with trajectory optimization for dense 3d mapping. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[7] T. Whelan, H. Johannsson, M. Kaess, J.J. Leonard, and J. McDonald. Robust real-time visual odometry for dense rgb-d mapping. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5724–5731, May 2013.

[8] Thomas Whelan, Stefan Leutenegger, Renato Salas Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense SLAM without a pose graph. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[9] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3D reconstruction in dynamic scenes using point-based fusion. In *3D Vision - 3DV 2013, 2013 International Conference on*, pages 1–8, June 2013.

[10] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE ISMAR*. IEEE, October 2011.

[11] I. Mahon, S.B. Williams, O. Pizarro, and M. Johnson-Roberson. Efficient view-based SLAM using visual loop closures. *Robotics, IEEE Transactions on*, 24(5):1002–1014, Oct. 2008.

[12] Albert S. Huang, Abraham Bachrach, Peter Henry, Michael Krainin, Dieter Fox, and Nicholas Roy. Visual odometry and mapping for autonomous flight using an rgb-d camera. In *In Proc. of the Intl. Sym. of Robot. Research*, 2011.

[13] D.M. Rosen, M. Kaess, and J.J. Leonard. An incremental trust-region method for robust online sparse least-squares estimation. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1262–1269, May 2012.

[14] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J Leonard, and Frank Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

[15] Michael Montemerlo and Sebastian Thrun. Simultaneous localization and mapping with unknown data association using fastslam. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 2, pages 1985–1991. IEEE, 2003.

[16] M.C. Graham, J.P. How, and D.E. Gustafson. Robust incremental SLAM with consistency-checking. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 117–124, Sept 2015.

[17] S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716, 2004.

[18] M. Walter, R. Eustice, and J. Leonard. Exactly sparse extended information filters for feature-based SLAM. *The International Journal of Robotics Research*, 26(4):335–359, 2007.

[19] J. Vial, H. Durrant-Whyte, and T. Bailey. Conservative sparsification for efficient and consistent approximate estimation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 886–893, Sep 2011.

[20] F. Dellaert and M. Kaess. Square root SAM: Simultaneous location and mapping via square root information smoothing. *International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[21] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *Robotics, IEEE Transactions on*, 24(6):1365–1378, Dec. 2008.

[22] Nicholas Carlevaris-Bianco, Michael Kaess, and Ryan M. Eustice. Generic node removal for factor-graph slam. *IEEE Robotics and Automation Society*, 30(6), September 2014.

[23] G. Huang, M. Kaess, and J. Leonard. Consistent sparsification for graph optimization. In *Mobile Robots (ECMR), 2013 European Conference on*, pages 150–157, Sept 2013.

[24] Mladen Mazuran, Wolfram Burgard, and Gian Diego Tipaldi. Nonlinear factor recovery for long-term SLAM. *International Journal of Robotics Research*, 2015.

[25] H. Kretzschmar and C. Stachniss. Information-theoretic compression of pose graphs for laser-based SLAM. *The International Journal of Robotics Research*, 31(11):1219–1230, 2012.

[26] V. Ila, J.M. Porta, and J. Andrade-Cetto. Information-based compact pose SLAM. *Robotics, IEEE Transactions on*, 26(1):78–93, Feb 2010.

[27] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research (JMLR)*, 9:235–284, June 2008.

[28] M. Beinhofer, J. MÃijller, and W. Burgard. Effective landmark placement for accurate and reliable mobile robot navigation. *Robotics and Autonomous Systems*, 61(10):1060 – 1069, 2013. Selected Papers from the 5th European Conference on Mobile Robots (ECMR 2011).

[29] M. Dymczyk, S. Lynen, M. Bosse, and R. Siegwart. Keep it brief: Scalable creation of compressed localization maps. In *Intelligent Robots and Systems*

(IROS), 2015 IEEE/RSJ International Conference on, pages 2536–2542, Sept 2015.

[30] S. Frintrop and P. Jensfelt. Attentional landmarks and active gaze control for visual SLAM. *Robotics, IEEE Transactions on*, 24(5):1054–1065, Oct 2008.

[31] A. Kim and R. Eustice. Active visual slam for robotic area coverage: Theory and experiment. *The International Journal of Robotics Research*, 2014.

[32] S. Hochdorfer and C. Schlegel. Landmark rating and selection according to localization coverage: Addressing the challenge of lifelong operation of slam in service robots. In *Intelligent Robots and Systems (IROS), 2009. IEEE/RSJ International Conference on*, pages 382–387, Oct 2009.

[33] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (slam) problem. In *Robotics and Automation (ICRA), 2000. IEEE International Conference on*, volume 2, pages 1009–1014, 2000.

[34] Zhang S., Xie L., and M. Adams. Entropy based feature selection scheme for real time simultaneous localization and map building. In *Intelligent Robots and Systems (IROS). 2005 IEEE/RSJ International Conference on*, pages 1175–1180, Aug 2005.

[35] H. Strasdat, C. Stachniss, and W. Burgard. Which landmark is useful? learning selection policies for navigation in unknown environments. In *Robotics and Automation (ICRA), 2009 IEEE International Conference on*, pages 1410–1415, May 2009.

[36] R. Lerner, E. Rivlin, and I. Shimshoni. Landmark selection for task-oriented navigation. *Robotics, IEEE Transactions on*, 23(3):494–505, June 2007.

[37] P. Sala, R. Sim, A. Shokoufandeh, and S. Dickinson. Landmark selection for vision-based navigation. *Robotics, IEEE Transactions on*, 22(2):334–349, April 2006.

[38] Huang Y. and K. Gupta. Collision-probability constrained PRM for a manipulator with base pose uncertainty. In *Intelligent Robots and Systems (IROS), 2009 IEEE/RSJ International Conference on*, pages 1426–1432, Oct 2009.

[39] B. Luders. *Robust Sampling-based Motion Planning for Autonomous Vehicles in Uncertain Environments*. PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, May 2014.

[40] L. Blackmore, M. Ono, and B. Williams. Chance-constrained optimal path planning with obstacles. *Robotics, IEEE Transactions on*, 27(6):1080–1094, dec. 2011.

[41] A. Agha-mohammadi, S. Chakravorty, and N. Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *The International Journal of Robotics Research*, 33(2):268–304, 2014.

[42] S. Prentice and N. Roy. *The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance*, volume 28, pages 1448–1465. Sage Publications, Inc., Thousand Oaks, CA, USA, November 2009.

[43] H. Kurniawati, T. Bandyopadhyay, and N. Patrikalakis. Global motion planning under uncertain motion, sensing, and environment map. *Autonomous Robots*, 33(3):255–272, 2012.

[44] Alain L. and Nadine L. Safe task planning integrating uncertainties and local maps federations. *The International Journal of Robotics Research*, 19(6):597–611, 2000.

[45] Jur van den Berg, Pieter Abbeel, and Kenneth Y. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research*, 30(7):895–913, 2011.

[46] T. A. Vidal-Calleja, A. Sanfeliu, and J. Andrade-Cetto. Action selection for single-camera slam. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(6):1567–1581, Dec 2010.

[47] Ruben Martinez-Cantin, Nando Freitas, Eric Brochu, José Castellanos, and Arnaud Doucet. A bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27(2):93–103, 2009.

[48] Frédéric Bourgault, Alexei A. Makarenko, Stefan B. Williams, Ben Grocholsky, and Hugh F. Durrant-Whyte. Information based adaptive robotic exploration. In *IEEE International Conference on Intelligent Robots and Systems*, 2002.

[49] H. Carrillo, I. Reid, and J.A. Castellanos. On the comparison of uncertainty criteria for active SLAM. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2080–2087, May 2012.

[50] L. Carlone, Jingjing Du, M.K. Ng, B. Bona, and M. Indri. An application of Kullback-Leibler divergence to active SLAM and exploration with particle filters. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 287–293, Oct 2010.

[51] J. VallvÃĺ and J. Andrade-Cetto. Active Pose SLAM with RRT*. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015.

[52] R. Valencia, J.V. Miro, G. Dissanayake, and J. Andrade-Cetto. Active pose SLAM. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1885–1891, Oct 2012.

[53] C. Leung, Shoudong Huang, and G. Dissanayake. Active slam in structured environments. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1898–1903, May 2008.

[54] G. Grisetti, R. KuÌĹmmerle, C. Stachniss, and W. Burgard. A Tutorial on Graph-Based SLAM. *Intelligent Transportation Systems Magazine, IEEE*, 2(4):31–43, winter 2010.

[55] Sudeep Pillai and John Leonard. Monocular SLAM supported object recognition. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[56] Shuran Song, Linguang Zhang, and Jianxiong Xiao. Robot in a room: Toward perfect object recognition in closed environments. *CoRR*, abs/1507.02703, 2015.

[57] Nikolay Atanasov, Menglong Zhu, Kostas Daniilidis, and George Pappas. Semantic localization via the matrix permanent. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[58] R.F. Salas-Moreno, R.A. Newcombe, H. Strasdat, P.H.J. Kelly, and A.J. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1352–1359, June 2013.

[59] J. Civera, D. Galvez-Lopez, L. Riazuelo, J.D. Tardos, and J.M.M. Montiel. Towards semantic SLAM using a monocular camera. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1277–1284, Sept 2011.

[60] N. Sunderhauf and P. Protzel. Towards a robust back-end for pose graph SLAM. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1254–1261, May 2012.

[61] Matthew C Graham and Jonathan P How. Robust simultaneous localization and mapping via information matrix estimation. In *Position, Location and Navigation Symposium-PLANS 2014, 2014 IEEE/ION*, pages 937–944. IEEE, 2014.

[62] Edwin Olson and Pratik Agarwal. Inference on networks of mixtures for robust robot mapping. *The International Journal of Robotics Research*, 32(7):826–840, 2013.

[63] Prabhakar Agarwal, Gian Diego Tipaldi, Luciano Spinello, Cyrill Stachniss, and Wolfram Burgard. Robust map optimization using dynamic covariance scaling. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 62–69. IEEE, 2013.

[64] Yasir Latif, CÃľsar Cadena, and JosÃľ Neira. Robust loop closing over time for pose graph SLAM. *The International Journal of Robotics Research*, 2013.

[65] B. Mu, A. Agha-mohammadi, L. Paull, M. Graham, J. How, and J. Leonard. Two-stage focused inference for resource-constrained collision-free navigation. In *Robotics Science and Systems*, Rome, Italy, July 2015.

[66] Beipeng Mu, Liam Paull, Ali akbar Agha-mohammadi, Jonathan How, and John Leonard. Two-stage focused inference for resource-constrained minimal collision navigation. *IEEE Transactions on Robotics*, 2016(under review).

[67] Beipeng Mu, Matthew Giamou, Liam Paull, John Leonard, and Jonathan P. How. Information-based active slam via topological feature graphs. In *IEEE Conference on Decision and Control (CDC)*, December 2016.

[68] Beipeng Mu, Shih-Yuan Liu, Liam Paull, John Leonard, and Jonathan P. How. Slam with objects using a nonparametric pose graph. In *Intelligent Robots and*

Systems (IROS), 2016 IEEE/RSJ International Conference on, Oct 2016(submitted).

[69] C. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st edition, 2007.

[70] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002.

[71] C. E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423, 1948.

[72] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999.

[73] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, January 2008.

[74] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[75] T. Steiner, G. Huang, and J. Leonard. Location utility-based map reduction. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 479–486, May 2015.

[76] D. Rosen, Huang G., and J. Leonard. Inference over heterogeneous finite-/infinite-dimensional systems using factor graphs and Gaussian processes. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1261–1268, May 2014.

[77] N. E. Du Toit and J. W. Burdick. Robot motion planning in dynamic, uncertain environments. *IEEE Transactions on Robotics*, 28(1):101–115, Feb 2012.

[78] Jur van den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012.

[79] N. Carlevaris-Bianco and R.M. Eustice. Generic factor-based node marginalization and edge sparsification for pose-graph SLAM. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5748–5755, May 2013.

[80] D. Bertsekas. *Dynamic Programming and Optimal Control*, volume I–II. Athena Scientific, PO Box 391, Belmont, MA 02178, 2007.

[81] H. Johannsson, M. Kaess, M. Fallon, and J.J. Leonard. Temporally scalable visual SLAM using a reduced pose graph. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 54–61, May 2013.

[82] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, May 2011.

[83] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, CIRA '97, pages 146–, Washington, DC, USA, 1997. IEEE Computer Society.

[84] L. Carlone and D. Lyons. Uncertainty-constrained robot exploration: A mixed-integer linear programming approach. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1140–1147, May 2014.

[85] Jason L. Williams, John Iii, and Alan S. Willsky. Performance guarantees for information theoretic active inference. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*, volume 2, pages 620–627. Journal of Machine Learning Research - Proceedings Track, 2007.

[86] G.E. Newstadt, B. Mu, D. Wei, J.P. How, and A.O. Hero. Importance-weighted adaptive search for multi-class targets. *Signal Processing, IEEE Transactions on*, 63(23):6299–6314, Dec 2015.

[87] M. Pavone, K. Savla, and E. Frazzoli. Sharing the load. *Robotics Automation Magazine, IEEE*, 16(2):52–61, June 2009.

[88] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1):34–46, 2007.

[89] Matan Keidar, Eran Sadeh-Or, and Gal A Kaminka. Fast frontier detection for robot exploration. In *Advanced Agent Technology*, pages 281–294. Springer, 2011.

[90] Eagle S. Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *Int. J. Rob. Res.*, 30(4):407–430, April 2011.

[91] I. Dryanovski, R.G. Valenti, and Jizhong Xiao. Fast visual odometry and mapping from rgb-d data. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2305–2310, May 2013.

[92] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[93] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[94] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In C.j.c. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2553–2561. 2013.

[95] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[96] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 2155–2162, Washington, DC, USA, 2014. IEEE Computer Society.

[97] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. Similarity Matching in Computer Vision and Multimedia.

[98] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, June 2005.

[99] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[100] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[101] Thomas Ferguson. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[102] Brian Kulis and Michael I. Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 513–520, New York, NY, USA, 2012. ACM.