2016 IEEE 55th Conference on Decision and Control (CDC)
ARIA Resort & Casino
December 12-14, 2016, Las Vegas, USA

Information-based Active SLAM via Topological Feature Graphs

Beipeng Mu[1] Matthew Giamou[1] Liam Paull[2] Ali-akbar Agha-mohammadi[3]
John Leonard[2] Jonathan How[1]

***Abstract*— Exploring an unknown space and building maps is a fundamental capability for mobile robots. For fully autonomous systems, the robot would further need to actively plan its paths during exploration. 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). Existing work has focused on planning paths with occupancy grid maps, which do not scale well and suffer from long term drift. This work proposes a Topological Feature Graph (TFG) representation that scales well and develops an active SLAM algorithm with it. The TFG uses graphical models, which utilize independences between variables, and enables a unified quantification of exploration and exploitation gains with a single entropy metric. Hence, it facilitates a natural and principled balance between map exploration and refinement. A probabilistic roadmap path-planner is used to generate robot paths in real time. Experimental results demonstrate that the proposed approach achieves better accuracy than a standard grid-map based approach while requiring orders of magnitude less computation and memory resources.**

I. Introduction

The exploration of an unknown space is a fundamental capability for a mobile robot, with diverse applications such as disaster relief, planetary exploration, and surveillance. In the absence of a global position reference (e.g., GPS) the robot must simultaneously map the space and localize itself within that map, referred to as SLAM. If a mobile robot is able to successfully recognize parts of the map when it returns to them, referred to as loop closure, then it can significantly reduce its mapping and localization error. The problem of active SLAM focuses on designing robot trajectories to actively explore an environment and minimize the map error.

Previous work has been done on designing trajectories to reduce robot pose uncertainty when the map is known or there exists a global position reference [1]–[5]. There also exists work that maximizes myopic information gain on the next action with partially known maps [6], [7]. However, in this work, the goal is to build a map of an unknown environment thus the robot needs to plan its path and perform SLAM at the same time (active SLAM) with a focus on *global* map quality. Active SLAM is non-trivial because the robot must trade-off the benefits of *exploring* new areas and *exploiting* visited areas to close loops [8].

[1]Laboratory for Information and Decision Systems, MIT, 77 Mass Ave, Cambridge, MA, USA {mubp, mgiamou, jhow}@mit.edu
[2]Computer Science and Artificial Intelligence Laboratory, MIT, 77 Mass Ave, Cambridge, MA, USA, {lpaull, jleonard}@mit.edu
[3]Qualcomm Research, 5775 Morehouse Drive, San Diego, CA, USA, aliagha@qualcomm.com

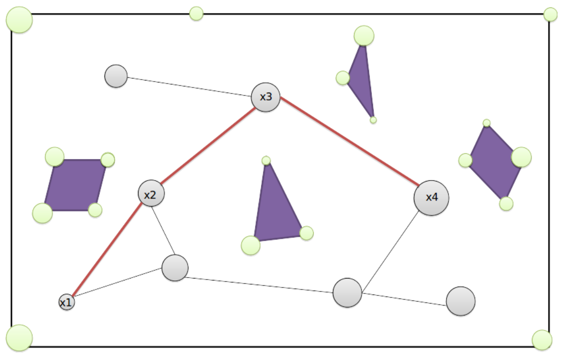


Fig. 1: Simultaneous planning, localization and mapping problem – purple polygons represent obstacles, green circles represent features with their size denoting uncertainties in pose estimates. The problem is to find milestones (gray circles) of robot poses, and plan a trajectory (red line) that minimizes feature uncertainties.

Selecting the right metric to quantify the benefits of exploration and exploitation is fundamental to active SLAM but challenging. For example, the seminal work of Bourgault et al. [9] formulates the problem as a trade-off between information gain about the map and entropy reduction over the robot pose:

$$u^* = \max_u w_1 I_{SLAM}(x,u) + w_2 I_{OG}(x,u) \tag{1}$$

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. [10] 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 [11], the D-optimality criterion [12], and the Kullback-Leibler divergence [13] 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 features in the map. These approaches have 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 feature-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, [14], [15] optimizes 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. [16] optimizes robot trajectory with features in a structured environment for also maintains an OG for collision check. Information quantification over the OG map representation carries over known shortcomings of bad scalability and robustness [17]. 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.

In a feature-based representation, features are explicitly maintained in the graph and give a layout of the map. However, active-SLAM on feature-based graphs is hard because features do not offer obstacle information, which is crucial for the robot to check path feasibility. This paper proposes the first, to our knowledge active SLAM approach that plans robot paths to directly optimize a global feature-based representation without any underlying OG representation. Rather than formulating the problem as area coverage over an OG map [18], we set it up as entropy reduction over the map features subject to a budget constraint. Since the feature estimates and pose trajectory are necessarily correlated, we can remove the pose uncertainty from the traditional objective function (1) and directly optimize over the map quality.

Figure 1 shows an example scenario. The locations of features 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 feature uncertainties.

In summary, there are three primary contributions. First, a feature-based topology graph is proposed to represent the map of features as well as obstacles in an efficient way. Second, a feature-focused information metric is developed to quantify uncertainties in the map, in both visited and unvisited places. Finally, a path planning algorithm is presented that uses the feature-based topological graph to enable the robot to actively explore with the objective of directly reducing the uncertainty of the map. The approach is tested in a Gazebo simulated environment, as well as in a real world environment with a turtlebot.

II. Problem Statement

Assume that there exists a library of static features that can be uniquely identified as landmarks to localize the robot in the environment, denoted as $\mathbf{L} = \{L_1, L_2, \cdots L_M\}$. Notice that the number of features present in the environment could be less than M, and is not known a priori. The exact locations of the present features 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 feature measurement z_t is a measurement between the current pose and features $z_t = \{z_{t,1}, \cdots, z_{t,M}\}$. When a feature is not observable, $z_{t,i}$ is defined to be null. The corresponding probability model of z_t is $p(z_t|X_t, \mathbf{L})$.

A factor graph is a sparse representation of the variables. Each node represents either a feature L_i or a robot pose X_t. Let $p(\mathbf{L}) = \prod_{i=1}^{M} p(L_i)$ denote the prior for features. Each factor is a feature prior $p(L_i)$, an odometry o_t or a feature measurement $z_{t,i}$. 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 $\mathbf{z} = \{z_1, \cdots, z_T\}$:

$$p(\mathbf{X}, \mathbf{L}|\mathbf{o}, \mathbf{z}) \propto p(\mathbf{L}) \prod_{t=1}^{T} p(o_t|X_t, X_{t-1}) p(z_t|X_t, \mathbf{L}). \quad (2)$$

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

$$(\mathbf{X}^*, \mathbf{L}^*) = \arg\max_{\mathbf{X},\mathbf{L}} p(\mathbf{X}, \mathbf{L}|\mathbf{o}, \mathbf{z}) \quad (3)$$

With factor graph representation, (3) can be solved by readily available graph-SLAM algorithms/packages such as g2o, iSAM or GTSAM [19], [20].

The problem has traditionally solved by manually operating the robot in the environment to gather a dataset first and then optimize the map in a batch update. In this work, 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 focus of this paper is to incrementally build a map of the environment, therefore information gain is defined as entropy reduction only on variables representing features.

Shannon entropy [21] is a measure of uncertainty in a random variable x thus widely used as information metric. Let $p(x)$ denote the probability distribution of x, then entropy $H(x)$ is defined as $H(x) = \sum_x p(x) \log p(x)$ for discrete variables and $H(x) = \int p(x) \log p(x) dx$ for continuous variables.

Denote the control command at time t as u_t, and let $\mathbf{u}_T = \{u_1, \cdots, u_T\}$. The active focused planning problem is summarized as follows.

Problem 1. ***Active Focused Planning:*** *Design control commands $\mathbf{u}_T = \{u_1, u_2, \cdots, u_T\}$, such that the robot follows a trajectory that the obtained odometry $\mathbf{o} = \{o_1, \cdots, o_T\}$*

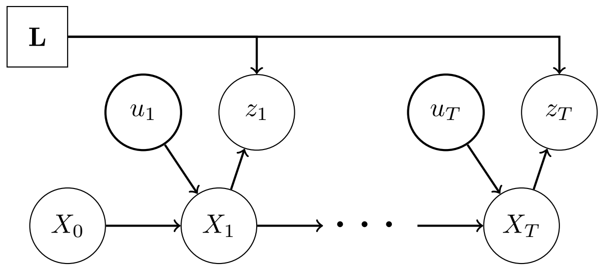


Fig. 2: **Active Focused Planning**. The robot uses landmark measurements z_t and odometry to design a control policy $u_{1:T}$ to maximize information gain over environmental features $\mathbf{L}$.

and feature measurements $\mathbf{z} = \{z_1, \cdots z_T\}$ *can minimize the entropy* $H(\cdot)$ *over the belief of map features* $\mathbf{L}$*:*

$$\begin{aligned} \max_{\mathbf{u}_T=\{u_1,\cdots,u_T\}} \quad & H(\mathbf{L}|\mathbf{o}, \mathbf{z}) \\ \text{s.t.} \quad & q(\mathbf{u}_T) \le c \\ & o_t = g(X_t, X_{t-1}, u_t) \\ & z_t = h(X_t, \mathbf{L}) \\ & t = 1, \cdots, T \end{aligned} \tag{4}$$

where $q(\cdot)$ is a measure of control cost, in the case of finite time horizon, $q(\mathbf{u}_T) = T$. Function $o_t = g(X_t, X_{t-1}, u_t)$ describes the odometry measurement model and $z_t = h(X_t, \mathbf{L})$ is the feature measurement model. Fig. 2 presents a graphical model of this problem. X_t represents robot poses, $\mathbf{L}$ represents environment features. The goal is to design control policies $u_{1:T}$ to maximize information gain over feature belief $\mathbf{L}$.

III. Method

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

In the current paper, we assume the robot is a ground robot that operates in 2D space[1]. Relying on the fact that features are usually on the surface or corner of obstacles, we propose the *Topological Feature Graph* (TFG) representation. A TFG is a graph $\mathcal{G} = \{L, E\}$, with its vertices representing features and edges representing obstacles. More specifically, if two features are connected by an edge, then these two features belongs to the same flat obstacle surface and the edge is not traversable[2].

[1]Extension to 3D scenarios can be achieved by triagularizing obstacle surfaces and is left to future work

[2]Features 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

These edges can be learned from either a depth image, a laser scan or even sequences of images [22]. The robot first segments the depth map or laser scan into several components representing different obstacle surfaces, then checks if two features detected belong to the same component. If so, the robot creates an edge between these two features. This idea is illustrated in Figure 3a.

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 features in cluttered/narrow spaces and less features 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 feature 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.

B. Sequential Optimization

Recall that our goal is to plan robot controls that gain maximal information from the environment as formulated in Problem 1. Notice that solving Problem 1 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, we instead split Problem 1 into T stages, optimize a goal point at each stage, and use a separate path planner to generate controls.

Problem 2. ***Incremental Active Planning*** *At stage t, given odometry history* $\mathbf{o}_t = \{o_1, \cdots, o_t\}$ *and feature measurement history* $\mathbf{z}_t = \{z_1, \cdots, z_t\}$*, find the next goal point* $\widehat{X}_{t+1}$ *such that the entropy on map features* $\mathbf{L}$ *is minimized:*

$$\begin{aligned} \widehat{X}_{t+1} = \min_{X_{t+1}} \quad & H(\mathbf{L}|\mathbf{o}_t, \mathbf{z}_t, \hat{z}_{t+1}) \\ \text{s.t.} \quad & o_t \ = g(X_\tau, X_{\tau-1}, u_\tau), \quad \tau = 1, \cdots, t \\ & z_\tau \ = h(X_\tau, \mathbf{L}), \quad \tau = 1, \cdots, t \\ & \hat{z}_{t+1} = h(\hat{X}_{t+1}, \mathbf{L}) \end{aligned} \tag{5}$$

Laplacian Approximation Let $p(\mathbf{L})$ denote a prior of the features, then 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:

$$\begin{aligned} \mathbf{X}_t^*, \mathbf{L}_t^* =& \text{argmax } p(\mathbf{X}, \mathbf{L}|\mathbf{o}_t, \mathbf{z}_t) \\ =& \text{argmax } p(\mathbf{L}) \prod_{\tau=1}^{t} p(o_\tau|X_\tau, X_{\tau-1}) p(z_\tau|X_\tau, \mathbf{L}) \end{aligned} \tag{6}$$

Laplacian approximation is a Gaussian approximation for probability of $\mathbf{X}$ and $\mathbf{L}$. The mean is its MAP values $(\mathbf{X}_t^*, \mathbf{L}_t^*)$ and its information matrix Λ is the second moment:

$$\mathbf{X}, \mathbf{L}|\mathbf{o}_t, \mathbf{z}_t \sim \mathcal{N}(\mathbf{X}_t^*, \mathbf{L}_t^*; \Lambda^{-1}) \tag{7}$$

$$\begin{aligned}\Lambda =& \frac{\partial^2 \log p(\mathbf{L})}{\partial(\mathbf{X},\mathbf{L})^2} + \sum_{\tau=1}^{t} \frac{\partial^2 \log p(o_\tau)}{\partial(\mathbf{X},\mathbf{L})^2} + \sum_{i=1}^{M} \frac{\partial^2 \log p(z_\tau)}{\partial(\mathbf{X},\mathbf{L})^2} \\ =& \begin{bmatrix} \Lambda_f & \Lambda_{fr} \\ \Lambda_{rf} & \Lambda_r \end{bmatrix}\end{aligned} \tag{8}$$

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

$$H(\mathbf{L}|\mathbf{o}_t, \mathbf{z}_t) = -\frac{1}{2}\log|\Lambda_{\mathbf{L}}| + \text{constant} \tag{9}$$

Further with the associated connectivity edges between features, 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.

C. Expected Information Gain

Quantifying the exact information gain from X_t to $\widehat{X}_{t+1}$ is challenging because it involves discretizing the trajectory from X_t to $\widehat{X}_{t+1}$ into a sequence of robot poses, 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 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 features at the goal point. The observation would be some layout of a subset of the local features. As shown in Figure 3b, gray balls denote observation points, and the blue circle indicate the set of features it can observe at those observation points.

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, feature measurements along the path are only used to localize the robot, but are not used to update feature 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.

Notice that $\widehat{X}_{t+1}$ is in continuous $\mathbb{R}^2$ space. Different $\widehat{X}_{t+1}$ would give different combination of observability of features, thus solving problem 2 exactly would be hard.

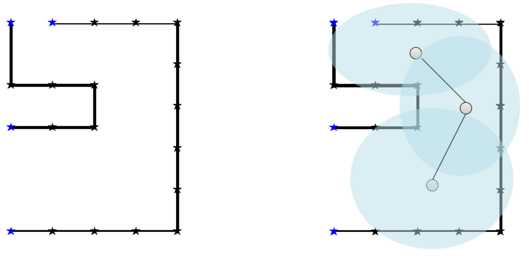

(a) Topology Feature Graph(TFG) (b) Goal points

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

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.

The maximal entropy reduction problem can be stated as follows:

$$\begin{aligned}\widehat{X}_{t+1} &= \arg\max_{X_{t+1}} \quad dH(\mathbf{L}|X_{t+1}, TFG_t) \\ &= \arg\max_{X_{t+1}} \quad H(\mathbf{L}|TFG_t) - H(\mathbf{L}|X_{t+1}, TFG_t)\end{aligned} \tag{10}$$

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

$$dH(X_{t+1}|X_{t+1}, TFG_t) = dH_o + dH_u \tag{11}$$

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

Proof can be found in appendix VI. Theorem 1 indicates that the information gain on a goal point can be split into two parts: the first part dH_o is the information gain obtained by re-observing and improving known features, and dH_u is the information gain from exploring new features. In our experiments, n_x is computed by using a predefined feature density in the environment multiplied by the size of a frontier at observation point X_{t+1}.

D. Frontier Detection

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

1) Compute features the robot expects to observe
2) Sort the features according to their orientation relative to the robot

3) If two consecutive feature are not connected to any neighbors, they represent a frontier.

With this frontier detection approach, we have a uniform information metric for both observed features and unobserved features 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 features, 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 feature estimates.

E. Path Planning

In Section III-B, 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 travelled and thus the control costs.
- Collision penalty.

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.

IV. Experiments

A. Information Measures

We first illustrate how the proposed framework can balance exploration and exploitation. Figure 4 shows an example scenario: black lines represent obstacles, stars represent features 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 4a displays the information gain on observed features: the total information gain is largest at samples that can potentially observe the greatest number of features. On the other hand, Figure 4b shows the information gain on new features. The samples closer to frontiers will have a chance to observe new features, thus they have higher exploration gains than samples further from frontiers. Assuming a fixed new feature density, larger frontiers offer the potential to observe more new features and therefore nearby samples have greater exploration information gain. Figure 4 shows the total exploration and exploitation information gain.

B. Simulation

We compared our framework with a nearest-frontier exploration algorithm [8] using the Gazebo simulator. The frontier exploration simulation used the popular GMapping [23] system for localization and mapping and used wavefront frontier detection [24] to identify frontiers.

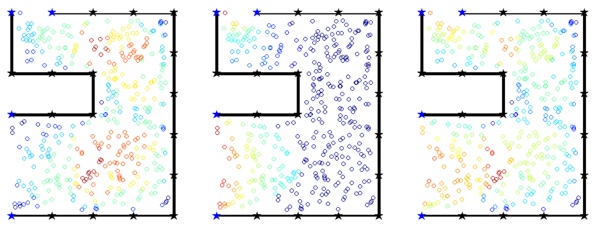

(a) Exploitition (b) Exploration (c) Total

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

The simulated TurtleBot receives noisy odometry, laser scans, and feature measurements. Table I contains the simulation parameters. Figure 5a displays a screenshot of the simulated environment (simulated april tags are spaced roughly one meter apart along the walls).

Figures 5b and 5c 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 features and thus maintain the shape of the building in its map.

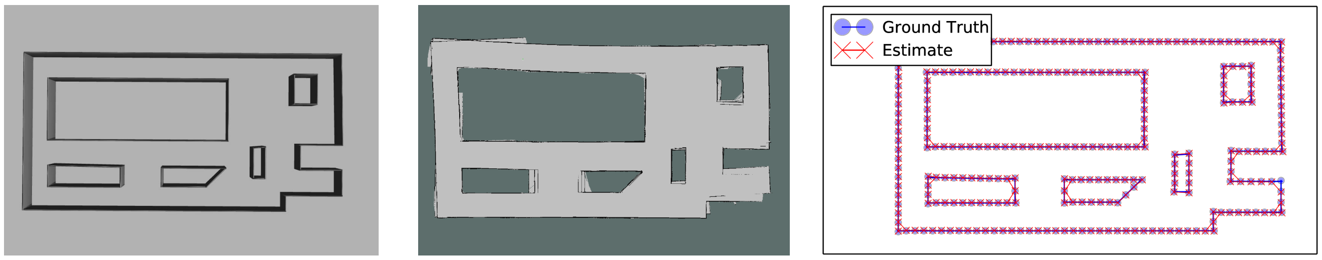
Figure 7 compares the robot pose error of nearest-frontier and TFG active SLAM over 4 runs. The solid lines represent error mean and shades represent error range. TFG active SLAM consistently has significantly less error in its robot pose estimates, especially in position. Figure 6 compares the map coverage with time spent exploring. In TFG active SLAM, the robot balances exploration with loop closing and is thus slightly slower in covering the whole space when compared with greedy frontier exploration. Table II 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.

TABLE I: 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

C. Hardware

The new framework is tested in an indoor space with the TurtleBot platform, using a computer with specifications listed in Table III. The computational resources used are readily available in many modern on-board systems. The focus of this paper is not on feature detection or data



(a) Gazebo simulation environment (b) frontier exploration with grid map (c) active SLAM via TFG

Fig. 5: 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 features and have much more accurate maps.

TABLE II: Simulation Performance Comparison

	TFG Active SLAM	grid map frontier
No. of variables	274	800000
CPU idle time	75%	0%
running 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

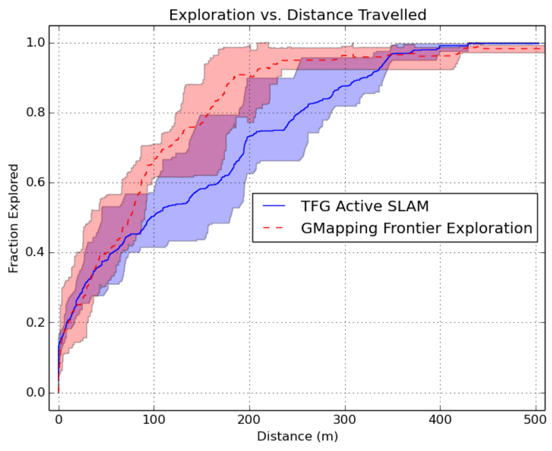


Fig. 6: Map coverage vs distance travelled. TFG builds an accurate map while exploring and is thus slightly slower than greedy nearest-frontier exploration.

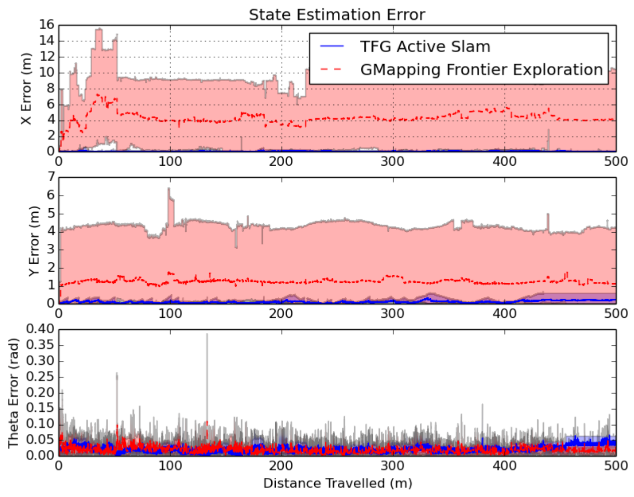


Fig. 7: Robot pose error over multiple runs. Solid line repsents mean and shade represent range. TFG has consistently smaller errors.

TABLE III: Hardware Specification

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

association, thus april tags [25] are used as features in the indoor space. Figure 9 gives some example views of the environment. Figure 8 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 features. The red dot is the robot's current position and the red lines are its planned trajectories.

V. Conclusion

This paper contributed to the problem of active simultaneous localization and mapping (SLAM) in three ways:

1) proposed a topological feature graph (TFG) that extends point estimates in SLAM to geometry representation of the space.
2) An information objective that directly quantifies uncertainty of a TFG. It captures correlations between robot poses and features in the space in a unified framework, thus new feature observations can help close loops and reduce uncertainties on observed features. The exploration and exploitation naturally comes out of the framework for a given feature density.
3) An efficient sampling-based path planning procedure within the TFG, which enables active SLAM.

Future work includes extending the algorithm to visual feature/object detection and association, and extending the TFG to work with 3D active SLAM.

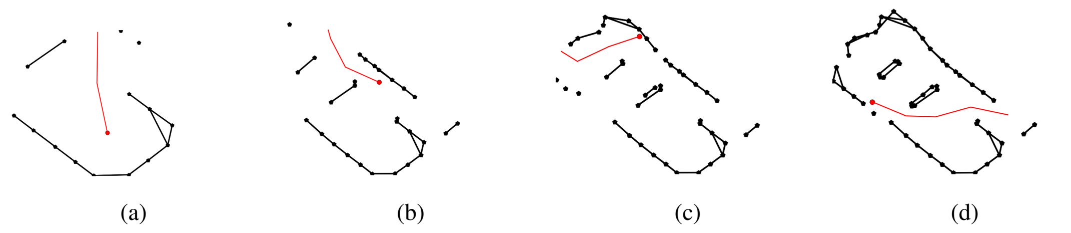
VI. Appendix

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 features 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 feature, therefore we



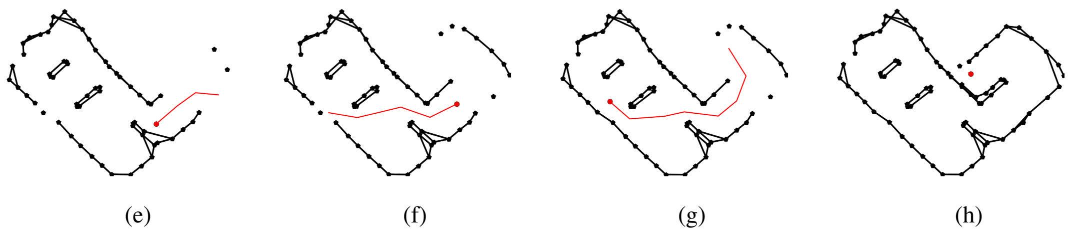


Fig. 8: Robot path and TFG in hardware experiment. Black stars represent april tags, and black lines represent obstacles. The red circle represents the robot's current location, and the red line represents robot's planned trajectory. The robot started with a partial map, then gradually picked up the frontiers and expanded the map to cover the space.

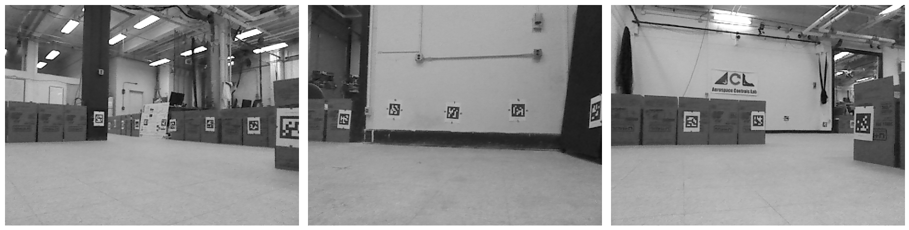


Fig. 9: Views of the space. An GPS-denied indoor environment with april tags as features.

only look at the marginal information matrix on features:

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

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

$$\begin{aligned} & p(\mathbf{o}_t, \mathbf{z}_t, \hat{z}_{t+1}; \widehat{X}_{t+1}, \mathbf{X}, \mathbf{L}) \\ & \quad \sim p(\mathbf{o}_t, \mathbf{z}_t; \mathbf{X}, \mathbf{L}) p(\hat{z}_{t+1} | \widehat{X}_{t+1}, \mathbf{L}) \end{aligned} \tag{12}$$

The corresponding factor graph is the factor graph at t plus new feature measurements $p(\hat{z}_{t+1}|\widehat{X}_{t+1}, \mathbf{L})$. Using the same ML values X_t^*, $\mathbf{L}_t^*$ in (3), the new information matrix $\Lambda_{\mathbf{L}}^{t+1}$ would be the original information matrix $\Lambda_{\mathbf{L}}$, plus some new terms coming from factors $p(\hat{z}_{t+1}|\widehat{X}_{t+1}, \mathbf{L})$:

$$\Lambda_{\mathbf{L}}^{t+1} = \begin{bmatrix} \Lambda_o & 0 & 0 \\ 0 & \Lambda_u & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} A_o & 0 & H_o \\ 0 & A_u & H_u \\ H_o^T & H_u^T & B \end{bmatrix} \tag{13}$$

where

$$\begin{aligned} B &= B_o + B_u \\ \begin{bmatrix} A_o & H_o \\ H_o^T & B_o \end{bmatrix} &= \frac{\partial^2 p(z_{t+1}|\widehat{X}_{t+1}, \mathbf{L})}{\partial(\mathbf{L}_o, \widehat{X}_{t+1})^2} \\ \begin{bmatrix} A_u & H_u \\ H_u^T & B_u \end{bmatrix} &= \frac{\partial^2 p(z_{t+1}|\widehat{X}_{t+1}, \mathbf{L})}{\partial(\mathbf{L}_u, \widehat{X}_{t+1})^2} \end{aligned} \tag{14}$$

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

$$\begin{aligned} \Lambda_{\mathbf{L}}^{t+1} &= \begin{bmatrix} \Lambda_o + A_o & 0 \\ 0 & \Lambda_u + A_u \end{bmatrix} - HB^{-1}H^T \\ H &= \begin{bmatrix} H_o \\ H_u \end{bmatrix} \end{aligned} \tag{15}$$

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

$$\begin{aligned} dH &= -\log|\Lambda_{\mathbf{L}}^{t}| + \log|\Lambda_{\mathbf{L}}^{t+1}| \\ &= \log\left|\begin{bmatrix} \Lambda_o + A_o & 0 \\ 0 & \Lambda_u + A_u \end{bmatrix} - HB^{-1}H^T\right| \\ &\quad - \log\left|\begin{bmatrix} \Lambda_o & 0 \\ 0 & \Lambda_u \end{bmatrix}\right| \end{aligned} \tag{16}$$

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

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

Extract the first term to get

$$\begin{aligned} dH &= \log\left|I + \begin{bmatrix} \Sigma_o A_o & 0 \\ 0 & \Sigma_u A_u \end{bmatrix}\right| \\ &\quad + \log\left|I - \begin{bmatrix} I + \Sigma_o A_o & 0 \\ 0 & I + \Sigma_u A_u \end{bmatrix}^{-1} HB^{-1}H^T\right| \end{aligned} \tag{17}$$

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

$$= \log \left| I + \begin{bmatrix} \Sigma_o A_o & 0 \\ 0 & \Sigma_u A_u \end{bmatrix} \right| \tag{18}$$

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

$$= \log |I + \Sigma_o A_o| + \log |I + \Sigma_u A_u| \tag{19}$$

$$+ \log \left| I - B^{-1} H_o^T (I + \Sigma_o A_o)^{-1} H_o - B^{-1} H_u^T (I + \Sigma_u A_u)^{-1} H_u \right|$$

When a feature 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 feature, $\log |I + \Sigma_u A_u| = n_x \log |I + \sigma_u a_u|$, and we have the following approximation:

$$\begin{aligned} dH \approx & \log |I + \Sigma_o A_o| + \log \left| I - B^{-1} H_o^T (I + \Sigma_o A_o)^{-1} H_o \right| \\ & + n_x \log |I + \sigma_u a_u| \\ = & \log |I + \Sigma_o A_o - H_o B^{-1} H_o| + n_x \log |I + \sigma_u a_u| \\ = & dH_o + dH_u \end{aligned} \tag{20}$$

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

ACKNOWLEDGMENTS

This research is supported in part by ARO MURI grant W911NF-11-1-0391, ONR grant N00014-11-1-0688 and NSF Award IIS-1318392.

REFERENCES

[1] Samuel Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, 2009.

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

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

[4] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *International Journal of Robotics Research (IJRR)*, 33(2):268–304, 2014.

[5] Jur van den Berg, Pieter Abbeel, and Kenneth Y. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *I. J. Robotic Res.*, 30(7):895–913, 2011.

[6] 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.

[7] 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.

[8] 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.

[9] F. Bourgault, A.A. Makarenko, S.B. Williams, B. Grocholsky, and H.F. Durrant-Whyte. Information based adaptive robotic exploration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 540–545, 2002.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

[15] 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.

[16] 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.

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

[18] 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.

[19] g2o: A general framework for graph optimization. `https://openslam.org/g2o.html`.

[20] 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.

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

[22] Sudeep Pillai and John Leonard. Monocular SLAM Supported Object Recognition. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[23] 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.

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

[25] Fast odometry from vision. `http://april.eecs.umich.edu/wiki/index.php/AprilTags`.