

MIT Open Access Articles

Information-based Active SLAM via topological feature graphs

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Mu, Beipeng et al. "Information-based Active SLAM via topological feature graphs." 2016 IEEE 55th Conference on Decision and Control December 2016, Las Vegas, NV, USA, Institute of Electrical and Electronics Engineers (IEEE), December 2016. © 2016 IEEE

As Published: <http://dx.doi.org/10.1109/cdc.2016.7799127>

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Persistent URL: <https://hdl.handle.net/1721.1/123811>

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Terms of use: Creative Commons Attribution-Noncommercial-Share Alike



Information-based Active SLAM via Topological Feature Graphs

Beipeng Mu¹ Liam Paull² Ali-akbar Agha-mohammadi³ John Leonard² Jonathan How¹

Abstract—Active SLAM is the task of actively planning robot paths while simultaneously building a map and localizing within. It is challenging in that the feasibility of paths will depend on future observations along the path, and the complexity of the problem grows quickly with the size of the map and the length of robot trajectory. This work proposes a Topological Feature Graph (TFG) representation of the map that scales well and develops an active SLAM algorithm with it. The TFG enables a unified quantification of exploration and exploitation gains with a single entropy metric and hence 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. Results from hardware experiment show that the proposed approach achieves better accuracy than the traditional grid-map based approaches 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 re-recognize parts of the map when it returns to them, referred to as loop closing, then it can significantly reduce its mapping and localization error for a bounded environment. The problem of active SLAM focuses on designing robot trajectories to explore an environment and minimize the error.

The quality of the resulting map is necessarily dependent on the trajectory that the robot takes through the environment. The problem of planning this trajectory is referred to as *active SLAM* and is non-trivial because the robot must trade-off the benefits of exploring new areas and revisiting explored areas to reduce the sensor location uncertainty. Previous work in this area usually involves two types of actions that are categorized via their purposes: *exploration actions* are used to guide the robot towards unexplored regions of the map [1], and *exploitation actions* are used to drive the robots towards already explored regions of the map for map refinement. Selecting the right metric to quantify the benefits of these actions is a challenging problem. A common choice for such metric is entropy reduction. For example, the seminal work

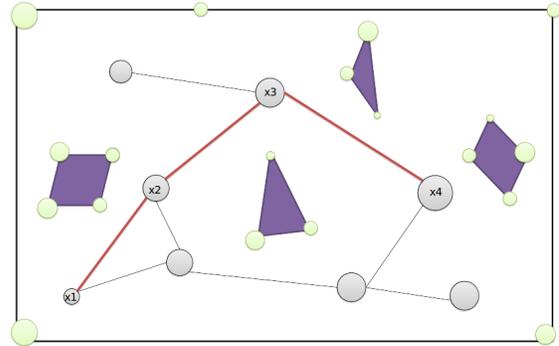


Fig. 1: Simultaneous planning, localization and mapping problem – purple polygons represent obstacles, green circles represent features with their size denoting uncertainties in feature estimates. The problem is to find milestones (gray circles) of robot poses, and plan a trajectory (red line) that minimize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

of Bourgault et al. [2] formulates the problem as a trade-off between information gain about the map and entropy reduction over the robot pose:

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

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. [3] use a Rao-Blackwellized particle filter (RBPF) to represent the state of the robot 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 [4], the D-optimality criterion [5], and the Kullback-Leibler divergence [6] have also been proposed recently. These two information gains are computed separately and maintaining the balance between 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, [7], [8] optimizes the entire robot trajectory by iteratively computing

¹Laboratory for Information and Decision Systems, MIT, 77 Mass Ave, Cambridge, MA, USA {mubp, jhow}@mit.edu

²Computer Science and Artificial Intelligence Laboratory, MIT, 77 Mass Ave, Cambridge, MA, USA, {lpau11, jleonard}@mit.edu

³Qualcomm Research, 5775 Morehouse Drive, San Diego, CA, USA, aliagha@qualcomm.com

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. Information quantification over the OG map representation carries over known shortcomings of bad scalability and robustness [9]. 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 graph is hard because features do not offer obstacle information. Therefore, the robot cannot check if a path is feasible. If both feature estimates and obstacle information are given *a priori*, there exist many planning algorithms that can generate paths for robots to accomplish different goals [10]–[14]. This paper proposes the first, to our knowledge active SLAM approach that plans robot paths to directly learn a feature-based representation. Rather than formulating the problem as an area coverage over the grid map [15], we set it up as entropy reduction over the map features subject to a budget constraint for exploration. Since the feature estimates and pose trajectory are necessarily correlated, we can remove the pose reduction objective from the traditional objective function (1) and directly optimize over the map quality.

When features and robot poses are modeled as joint Gaussian variables, we can directly quantify the information gain of new data on all variables with the same entropy metric. When the robot moves to a frontier of the mapped area, it can potentially observe new features. The observed features and new features are measured with a unified information metric, therefore we can balance between exploitation and exploration automatically. The feature-based graph model is sparser than grid maps, thus scales much better and enables real-time robot state estimation and path planning. Furthermore, it models the environment with dependent features and robot poses rather than with i.i.d. binary cells. Therefore, when the robot returns to a visited place and closes a loop, it can correct long-term drift and propagate the changes to all existing features and poses through the dependencies between variables.

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 pose, and purple polygons represent obstacles. The planning problem is then to find milestones (i.e., discrete set of poses) and a trajectory connecting the milestones that can minimize the 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. And finally, a path planning algorithm 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 with a real robot demonstrating that it achieves better accuracy than the traditional grid-map based approaches while requiring orders of magnitude less computation and memory resources.

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 $\mathbf{L} = \{L_1, L_2, \dots, L_M\}$. The exact locations of these features are not known *a priori* and need to be established by the robot. When moving in the environment, the robot’s trajectory is a sequence of poses $\mathbf{X}_T = \{X_0, X_1, \dots, X_T\}$, 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 first is the odometry change between two consecutive poses o_t with probability model $p(o_t|X_t, X_{t-1})$. The second is a measurement between the current pose and landmarks $z_t = \{z_{t,1}, \dots, z_{t,M}\}$. When a feature is not observable, $z_{t,i}$ is defined to be null. The corresponding probability model of z_t is $p(z_t|X_t, \mathbf{L})$. Given priors on \mathbf{L} , the joint posterior of \mathbf{X} and \mathbf{L} given the observations $\mathbf{o} = \{o_1, \dots, o_T\}$ and $\mathbf{z} = \{z_1, \dots, z_T\}$ is:

$$p(\mathbf{X}, \mathbf{L} | \mathbf{o}, \mathbf{z}) \propto p(\mathbf{L}) \prod_t p(o_t | X_t, X_{t-1}) \prod_i p(z_i | X_t, \mathbf{L}). \quad (2)$$

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

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

A factor graph is a sparse representation of (2). Each node is a random variable (feature pose L_i or robot pose X_t). And a factor is an observation of the relative pose between two variables (o_t or z_t). With this representation, (3) can be solved by readily available graph-SLAM algorithms/packages such as g2o, iSAM or GTSAM [16], [17].

Typically, the problem is 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. Consider that robots are typically constrained in computation/memory, the trajectory should be planned in such a way that the resource should be spent on gathering information that is directly related to the robot’s goal. In focus in this paper is incrementally build a map of the environment, therefore information gain is defined as entropy reduction only on variables representing features.

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

Problem 1. Active Focused Planning: Design control commands $\mathbf{u}_T = \{u_1, u_2, \dots, u_T\}$, such that the robot follows a

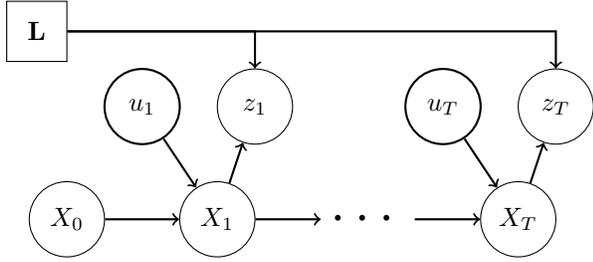


Fig. 2: **Active Focused Planning.** X_t represents robot poses, \mathbf{L} represents environment features. The goal is to design control policies $u_{1:T}$ to maximize information gain over feature belief \mathbf{L} .

trajectory that the obtained odometry $\mathbf{o} = \{o_1, \dots, o_T\}$ and feature measurements $\mathbf{z} = \{z_1, \dots, z_T\}$ can maximize some information metric $f(\cdot)$ over the belief of map features \mathbf{L} :

$$\begin{aligned} \max_{\mathbf{u}_{1:T} = \{u_1, \dots, u_T\}} \quad & \sum_{t=0}^T f(\mathbf{L} | o_{1:t}, z_{1:t}) \\ \text{s.t.} \quad & q(\mathbf{u}) \leq c \\ & o_t = g(X_{t+1}, X_t, u_t) \\ & z_t = h(X_t, \mathbf{L}) \end{aligned} \quad (4)$$

where $f(\cdot)$ is a metric on the information of map features \mathbf{L} given observations $o_{1:t}$ and $z_{1:t}$, for example, entropy $H(\cdot)$ is a widely used metric. $q(\cdot)$ is a measure of control cost, in the case of finite time horizon, $q(\mathbf{u}_T) = T$. Function $o_t = g(X_{t+1}, X_t, u_t)$ 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

The first challenge encountered with feature based path planning is obstacle representation. One important reason that the use of grid-map representation has been the main choice for active-SLAM is that a grid-based map contains all the necessary information for planning paths. A feature-based representation, although much sparser, lacks information about free/occupied space or 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 scenario, obstacles then can be represented as lines or polygons in 2D space¹. Relying on the fact that features are usually on the surface or corner of obstacles, we propose the representation of *Topological Feature Graph* (TFG). A TFG is a graph $\mathcal{G} = \{\mathbf{L}, E\}$, with its vertices representing features, and edges representing

obstacles. More specifically, if two features are connected by an edge, then these two features belong to the same flat obstacle surface, and thus not traversable by the robot.

These edges can be learned from either a depth image, a laser scan or even sequences of images [18]. 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 great memory savings. Second, the map complexity can easily adapt to various complexities in the environment. Instead of using equal sized cells at all places, TFG can model more features in more 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, thus lead to significant computation saving in path planning.

B. Sequential Optimization and Observation Point

Recall that our goal is to plan robot controls to get maximal information of the environment, formulated in Problem 1. Notice that solving Problem 1 in batch is hard in general, because at any time t , observations beyond t is not available, thus planning controls u_t, \dots, u_T will require modeling future observations and taking into account all possible outcomes, which is typically intractable.

To solve this problem, split Problem 1 into T stages and optimize the one-step-ahead control at each stage.

Problem 2. Incremental Active Planning At stage t , given odometry history $o_{1:t} = \{o_1, \dots, o_t\}$ and feature measurement history $z_{1:t} = \{z_1, \dots, z_t\}$, find the control u_t such that the information on map features \mathbf{L} is maximized:

$$\begin{aligned} \max_{u_t} \quad & f(\mathbf{L} | o_{1:t}, z_{1:t}) \\ \text{s.t.} \quad & o_t = g(X_{t+1}, X_t, u_t), \quad \tau = 1, \dots, t \\ & z_\tau = h(X_\tau, \mathbf{L}), \quad \tau = 1, \dots, t \end{aligned} \quad (5)$$

Laplacian Approximation Note the observations history $o_{1:t}$ and $z_{1:t}$ can be summarized into a posterior distribution of \mathbf{L}, X_t at time t and the maximal likelihood values of X_t^* and L_t^* can be obtained by standard SLAM solvers

$$\mathbf{X}_t^*, \mathbf{L}_t^* = \arg\max p(\mathbf{X}_t, \mathbf{L} | o_{1:t}, z_{1:t}) \quad (6)$$

$$= \arg\max p(\mathbf{L}) \prod_{\tau=1}^t p(o_\tau | X_\tau, X_{\tau-1}) p(z_\tau | X_\tau, \mathbf{L})$$

Now linearize the posterior $p(\mathbf{X}_t, \mathbf{L} | o_{1:t}, z_{1:t})$ at the MAP values $(\mathbf{X}_t^*, \mathbf{L}_t^*)$ to obtain the Laplacian approximation (i.e.,

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

approximate the pdf with its two first moments):

$$\mathbf{X}_t, \mathbf{L}_t \sim \mathcal{N}(\mathbf{X}_t^*, \mathbf{L}_t^*; \Sigma) \quad (7)$$

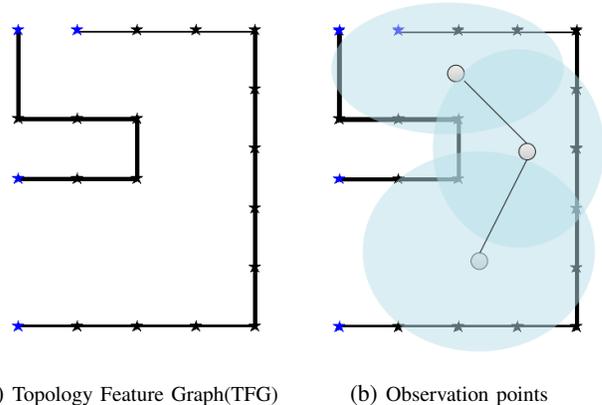
where Σ denote the joint covariance matrix of \mathbf{X}_t and \mathbf{L} , which is computed from Hessian matrix of posterior $p(\mathbf{X}_t, \mathbf{L} | o_{1:t}, z_{1:t})$ over \mathbf{X}_t and \mathbf{L} . With the approximate distribution of $\mathbf{X}_t, \mathbf{L}_t$, and associated connectivity 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 .

Observation Point 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 through 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.

Quantifying information along a path is not easy either, because information gain on later parts of the path will depend on early observations along the path. Therefore, the complexity will grow exponentially with the path length. To solve the information quantification problem in real-time, define an observation point as a point in space, at a vicinity of which the robot will stay, spin around to explore the environment. 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. This exploration will enable the robot to obtain an accurate measure of the local part of the overall environment.

Observation points also provide a way to segment the overall map and sparsify the underlying SLAM factor graph: robot accurately maps the environment at observations points, thus measurements between two observation points contains less information compared to those at observation 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 marginal out robot poses between two observations points, and the SLAM factor graph will become a joint graph of partial graphs at observation points. In this way, the complexity of the SLAM factor graph only scales with the number of observation points, but not the number of robot poses, and leads to great benefits in scalability. Furthermore, paths are generated with respect to current estimate of feature locations. If measurements along a path are used to update feature estimates, new loop closures may cause shifts of feature 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.

Problem 3. Active Planning based on Observation Points At stage t , given odometry history $o_{1:t} = \{o_1, \dots, o_t\}$ and



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

Fig. 3: Topology Feature Graph(TFG) and observation points. Vertex (star) representing features, edges (black line) representing obstacle surfaces, blue stars represent features at frontier, gray balls represent observation points. At each observation point, the robot stays and spins around to find out an accurate estimate of local features (stars). Blue regions illustrate local maps associated with observation points.

feature measurement history $z_{1:t} = \{z_1, \dots, z_t\}$, find the best observation point \hat{X}_{t+1} such that the information on map features \mathbf{L} is maximized:

$$\begin{aligned} & \max_{\hat{x}_{t+1}} f(\mathbf{L} | o_{1:t}, z_{1:t}, \hat{z}_{t+1}) \\ \text{s.t. } & o_t = g(X_{t+1}, X_t, u_t), \quad \tau = 1, \dots, t \\ & z_\tau = h(X_\tau, \mathbf{L}), \quad \tau = 1, \dots, t \\ & z_{t+1} = h(\hat{X}_{t+1}, \mathbf{L}) \end{aligned} \quad (8)$$

where \hat{z}_{t+1} represents the expected feature observations at \hat{X}_{t+1}

To avoid the robot jumping between two frontiers that are far away, we constrain the observation point search to be within a certain radius of the robot's current location. When there are no frontiers or good observation points, the search radius will be increased incrementally until the robot finds new frontiers or good observation points.

Compute Expected Information Gain If entropy $H()$ is used as the metric for information, then Problem 2 can be solved by the following procedure. Given TFG_t , sample observation points in robot's reachable space, compute information gain on each of the observation points, select the one gives maximal improvement of the map information, then plan a path to that point. The following states how the information gain over observation points can be computed, the next section presents path planning.

The entropy maximization can be stated as follows:

$$\max_{\hat{X}_{t+1}} H(\mathbf{L} | \hat{X}_{t+1}, TFG_t) \quad (9)$$

For simplicity the subscript t is dropped in the following, but it should be noted that this analysis is based on TFG_t . From (7), the joint covariance matrix of features and robot poses is Σ . Further split this matrix into 3 parts: the robot poses, the features that have been observed at least once, and

the features that are not observed:

$$\Sigma = \begin{bmatrix} \Sigma_r & \Sigma_{ro} & 0 \\ \Sigma_{or} & \Sigma_o & 0 \\ 0 & 0 & \Sigma_u \end{bmatrix} \quad (10)$$

where Σ_r , Σ_o and Σ_u are marginal covariance matrix of robot poses, observed features and unobserved features. Especially, for previously unobserved features, Σ_u is the prior covariance matrix, which is set to be diagonal and same for each unobserved feature in this case. Assume a density of features in the environment, the number of features is computed as the density times the size of the frontier at a particular observation point.

Since the goal here is to estimate feature positions, take the marginal covariance matrix on focused variables: $\text{diag}(\Sigma_o, \Sigma_u)$, where $\text{diag}(\cdot)$ concatenates its input matrices in a block diagonal format. The corresponding block-wise marginal information matrix on features is:

$$\Lambda_{\mathbf{L}}^t = \begin{bmatrix} \Lambda_o & 0 \\ 0 & \Lambda_u \end{bmatrix} \quad (11)$$

At observation point x , the robot will obtain measurements of features associated with the local map. The new posterior becomes

$$p(\hat{X}_{t+1}, L | o_{1:t}, z_{1:t}, \hat{z}_{t+1}) \\ \sim p(X_t, L | o_{1:t}, z_{1:t}) p(\hat{z}_{t+1} | \hat{X}_{t+1}, \mathbf{L}) \quad (12)$$

The corresponding posterior factor graph at $t + 1$ is the factor graph at t plus new terms $p(\hat{z}_{t+1} | \hat{X}_{t+1}, \mathbf{L})$. Assume the MAP estimate X_t^* , \mathbf{L}^* in (3) takes the same value after incorporation new data z_{t+1} , then the linearization point at time $t+1$ is the same as that at t for Laplacian approximation. And the new information matrix new robot pose \hat{X}_{t+1} , $\Lambda_{\mathbf{L}}^{t+1}$ would be the original information matrix without \hat{X}_{t+1} , $\Sigma_{\mathbf{L}}$, plus some new terms coming from factors $p(\hat{z}_{t+1} | \hat{X}_{t+1}, \mathbf{L})$:

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

where,

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

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

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

Note the element in A and H is 0 if the corresponding feature is not observable at observation point \hat{X}_{t+1} . Then

the incremental change in the information objective $H(\cdot)$ is:

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

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

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

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

$$= \log \left| I + \begin{bmatrix} \Sigma_o A_o & 0 \\ 0 & \Sigma_u A_u \end{bmatrix} \right| \\ + \log \left| I - B^{-1}H^T \begin{bmatrix} (I + \Sigma_o A_o)^{-1} & 0 \\ 0 & (I + \Sigma_u A_u)^{-1} \end{bmatrix} H \right| \\ = \log |I + \Sigma_o A_o| + \log |I + \Sigma_u A_u| \\ + \log \left| I - B^{-1}H_o^T (I + \Sigma_o A_o)^{-1} H_o - B^{-1}H_u^T (I + \Sigma_u A_u)^{-1} H_u \right| \quad (18)$$

When a feature is not observed previously, the prior variance Σ_u is typically large, therefore $(I + \Sigma_u A_u)^{-1} \approx 0$. 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:

$$dH \approx \log |I + \Sigma_o A_o| + \log |I - B^{-1}H_o^T (I + \Sigma_o A_o)^{-1} H_o| \\ + n_x \log |I + \sigma_u a_u| \\ = \log |I + \Sigma_o A_o - H_o B^{-1} H_o| + n_x \log |I + \sigma_u a_u| \\ = dH_o + dH_u \quad (20)$$

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.

Equation (20) indicates that the information gain on an observation point can be naturally split into two parts, the first part dH_o is the information gain obtained by re-observing and improving known features, and dH_u in the information gain on exploring new features. In experiment, n_x is computed by using a predefined feature density in the environment, times the size of frontier at observation point x . This way, our information metric gives a natural balance between exploration and exploitation: if there are big frontiers with potential to discover more new features, the robot will pick observation points to explore frontiers.

If there are none or only small frontiers, the robot might go to visited places to improve existing feature estimates. This trade-off will be illustrated with figures in section IV-A.

C. Path Planning

Path planning is not the focus of this paper, thus we do not go into details of the methods used in our work. At a high level, we have already obtained a set of collision-free samples in Section III-B, therefore the path planner will only compute connectivity and cost between these samples, and form a probabilistic roadmap (PRM). Trajectory to the next best observation point is generated by computing a minimum cost path on PRM. The cost of an edge between two sample points involve two factors:

- Length of the link, reflects the distance needs to be traveled thus control costs.
- Collision probability.

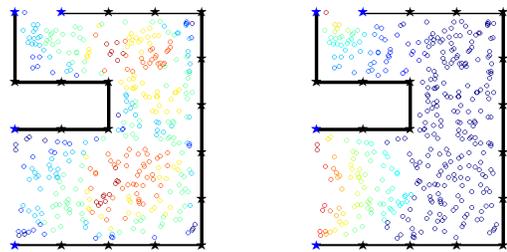
Computing exact collision probability of a given path is a computationally expensive procedure. However, exploiting the fact that a collision check for a point using a TFG representation can be carried out analytically, enables expensive methods such as Monte Carlo for real-time collision evaluation. However, 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.

At any point x along an edge, assume the robot position uncertainty is a Gaussian distribution with covariance Σ . Let c denote the Mahalobinas distance to the closest obstacle $c(x) = (x - x_o)\Sigma^{-1}(x - x_o)$, then the probability is approximated as $P_c(x) \sim \exp\{-c(x)\}$. If assume $\Sigma = \sigma^2 I$ for all x , then $P_c(x)$ can be further simplified to $P_c(x) = \exp\left\{-\frac{\|x-x_o\|^2}{\sigma^2}\right\}$. With our TFG representation, computing $\|x-x_o\|^2$ will reduces to computing distance between point-line, line-line distances, which can be achieved trivially.

IV. EXPERIMENTS

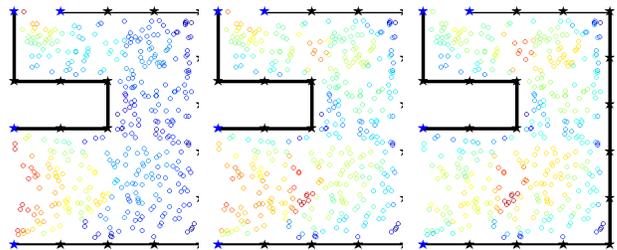
A. Information Measures

First we illustrate how the new proposed framework can automatically balance between exploration and exploitation. Figure 4 shows an example scenario, black lines represent obstacles, stars represent features with blue color denoting frontier. Circles are samples in the free space, with their color representing the information gain. Red means high while blue means low. Figure 4a gives the information gain on observed features, it can be seen that the total information gain is biggest at places that can potentially observe the most 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 have higher exploration gains compared with samples inside observed space. Assume a fixed new feature density, the bigger the frontiers, the more number of new features can potentially be observed, thus the bigger the information gain will be.



(a) Info gain by observing existing features (b) Info gain by observing new features at frontiers

Fig. 4: Information gain with observing existing features or obtaining new features. Black line (obstacle) and star(feature) represent the TFG. Blue star indicate frontier features. Color of circles represent information gain on samples.



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

Fig. 5: Total information gain with different unseen feature density. When robot expects to see many features beyond frontiers, information gain at frontiers are high. Otherwise, the robot prefers spots that can observe most number of features in visited places.

Sum up the exploitation and exploration information, Figure 5 gives the total information gain under high/medium/low prior variance on new features. When prior variance on new features is high, observing a new feature will give big information gains, thus the exploration terms 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 features does not add much information, and the robot will prefer to revisit places with observed features and improve estimates of them.

B. Hardware

The new framework is tested in an indoor space with turtlebot, using a computer with limited resources. The hardware specification is listed in Table I. This set of hardware can easily achieved by modern on-board systems. The focus of this paper is not on feature detection or data association, thus april tags [19] are used as features in the indoor-space. Figure 7 gives some example views about the environment. Figure 6 shows how the robot progressed in the environment. 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, black dots are features. The red dot is the robot's current position and the red lines is its planned trajectory. It can be seen that instead of greedily picking the biggest frontier, the robot sometimes went back to relative

TABLE I: Hardware Specification

Robot	turtlebot (kobuki base)
Processor	Intel Core i3 duel @2.3GHz
RAM	4GB
Operating System	Ubuntu 14.04

well-explored regions to close loops and improve the existing map.

C. Result comparison

Because the problem was set up in a fundamentally different way than for most of the existing literature, it is difficult to provide a direct quantitative comparison of the algorithm's performance against previous techniques. In this paper, we provide a qualitative evaluation over two other ways: (1) generate a TFG by asking an expert operator to drive the same robot in the environment. and (2) generate a grid-map by post-processing the same trajectory as TFG-based active SLAM with the same computer. Because registering laser scans and updating particles is computationally expensive, 30 particles were used and the data processing was slowed down 20 times to obtain a reasonable map. To further test the robustness of the algorithm, we also disturbed the robot pose by about one meter in the middle of the experiment.

Figure 8 shows the maps generated by three different methods. The proposed TFG-based active SLAM method (Figure 8a) well recovers the space even with disturbance. The map by human-operated data (Figure 8b) captures the basic structure, but missed some obstacles on the wall and some surfaces on the obstacles in the middle of the room. This is mainly because human operators do not have a metric model of the environment, and cannot tell if some place is well explored. Finally, there is significant distortion in grid maps (Figure 8c). And after the disturbance, the map completely drifted. Laser scans do not use any features in the environment as landmarks, thus once the odometry is drifted, it will very hard to correct even the robot comes back to the original place.

Table II further compares the resources required by TFG-based active SLAM and grid maps. The numbers give the order of computation/memory cost. The new framework improves map representation, map updates, localization, and path planning by factors of 10-100 times. It is better for long-term operation as it is robust to disturbances and can close loops when robot revisit a place. However, the new representation relied on good feature detection and data association mechanisms, which can be quite difficult in unstructured environment.

V. CONCLUSION

This paper contributed to the problem of active simultaneously 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,

TABLE II: Resource Requirement

	TFM	grid-map
Map Size	20m×20m	30m×20m
# of Variables	100 light memory	10 ⁵ heavy memory
Map Update	isam 10ms	thresholding 1s
Localization	closed form instantaneous	scan registration 1s
Path Planning	sampling 10ms	A* 10ms
Info Gain	closed form 100ms	trajectory based 1s
Loop Closure	easy	hard
Long-term Map Improvement	easy	hard
Exploit vs Explore	easy	hard
Operation in Unstructured Environments	hard	easy
Feature Detect	usually hard	easy

*number in the table indicate orders of magnitude

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.

Feature work includes extending the algorithm to visual feature/object detection and association, and extend the TFG to work with 3D cases.

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] 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.
- [2] 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.
- [3] 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.
- [4] 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.
- [5] 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.
- [6] 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.
- [7] J. Vallv and J. Andrade-Cetto. Active Pose SLAM with RRT*. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015.
- [8] 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.

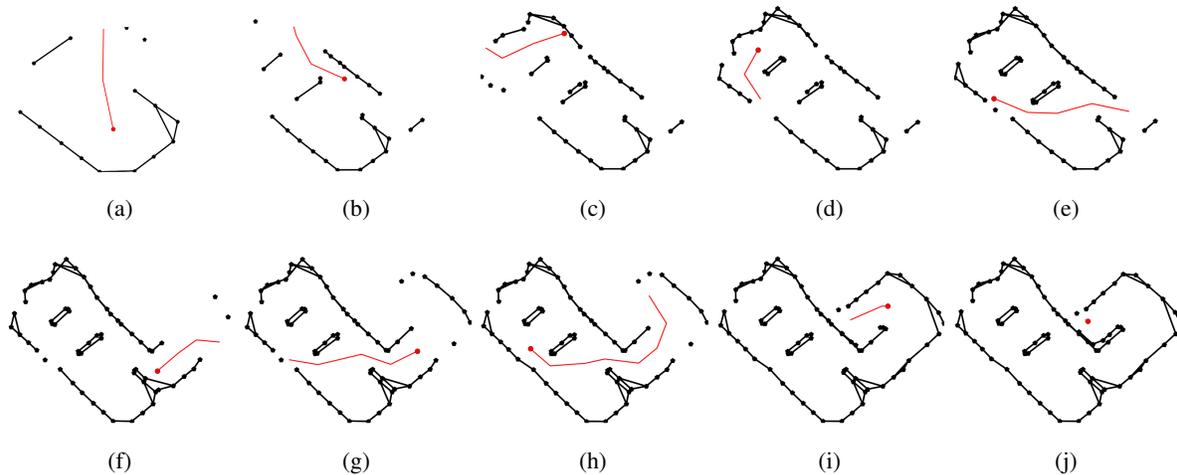


Fig. 6: Robot path and TFG in hardware experiment. Black stars represent apriltags, black lines represent obstacles. Red circle represents robot's current location, and red line represents robot's planned trajectory. The robot started with a partial map, then gradually picked up the frontiers and expand the map to cover the space.

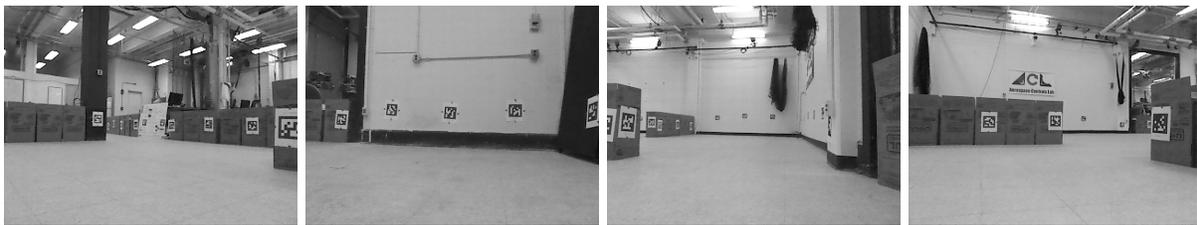


Fig. 7: Views of the space. An GPS-denied indoor environment with april tags as features.

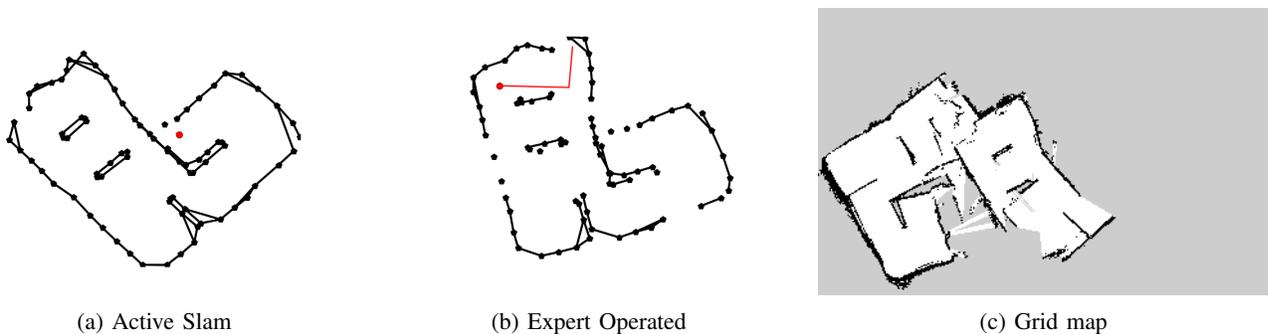


Fig. 8: Comparison of different policies. TFG-based active SLAM method well recovers the space even with disturbance. The map by human-operated data captures the basic structure, but missed some obstacles on the wall and some surfaces on the obstacles in the middle of the room. Grid map has significant distortion and completed drifted after the disturbance.

- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] Samuel Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, 2009.
- [14] 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.
- [15] 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.
- [16] g2o: A general framework for graph optimization. <https://openslam.org/g2o.html>.
- [17] 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.
- [18] Sudeep Pillai and John Leonard. Monocular SLAM Supported Object Recognition. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.
- [19] Fast odometry from vision. <http://april.eecs.umich.edu/wiki/index.php/AprilTags>.