Toward SLAM on Graphs

Avik De, Jusuk Lee, and Noah J. Cowan

Department of Mechanical Engineering, Johns Hopkins University
{avik.de,jsl,ncowan}@jhu.edu

Abstract: We present an algorithm for SLAM on planar graphs. We assume that a robot moves from node to node on the graph using odometry to measure the distance between consecutive landmark observations. At each node, the robot follows a branch chosen at random, without reporting which branch it follows. A low-level process detects (with some uncertainty) the presence of landmarks, such as corners, branches, and bumps, but only triggers a binary flag for landmark detection (i.e., the robot is oblivious to the details or "appearance" of the landmark). Under uncertainties of the robot's odometry, landmark detection, and the current landmark position of the robot, we present an E-M-based SLAM algorithm for two cases: (1) known, arbitrary topology with unknown edge lengths and (2) unknown topology, but restricted to "elementary" 1- and 2-cycle graphs. In the latter case, the algorithm (flexibly and reversibly) closes loops and allows for dynamic environments (adding and deleting nodes).

1 Introduction

Navigation of inexpensive mobile robots with limited computational capabilities, imprecise sensing, and crude odometry presents a number of interesting challenges. Here, we approach the problem of Simultaneous Localization and Mapping (SLAM, cf. [11]) in this setting. We assume that as a robot moves through the environment, a low level control algorithm allows the robot to follow physical structures (walls, doorways, etc.). These physical structures are presumed to give rise to a natural graph structure where nodes of the graph are intermittent features detected by the robot's sensory system, including doorways, corners, or bumps on the wall (Fig. 1). This sensory system was inspired by an artificial antenna [18] from which the tactile feedback received is close range, intermittent, and sparse. This means that the robot needs only run its mapping and localization algorithm occasionally (when a feature is detected), but that only "corridor-like" environments with walls are considered in this paper. Here, the observation is simply the odometrically measured dis-

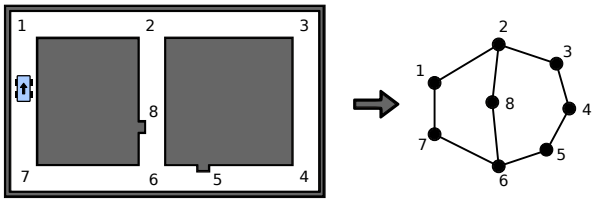

Fig. 1. An illustration of how an environment with landmarks is treated purely as a graph by the robot.

tance traversed since the last detected node, and we allow for the possibility of missing nodes or detecting false positives.

In short, we propose a solution to the SLAM problem given sparse sensory data (binary and intermittent) and a low dimensional state space. The topological approach lets us abstract the application (for example an indoor environment) from the basics of the algorithm.

Almost all existing SLAM approaches use some statistical technique due to the inherent uncertainty in noisy robot motion and/or observation. Csorba [8] developed the theory behind a modified Extended-Kalman-Filter (EKF) based SLAM algorithm; the EKF method has been improved and used extensively such as in [13]. By the nature of an EKF, the motion model and observation noise are independent: the sensory noise is a function of the sensor physics, and is independent of the robot's motion noise. By contrast, our approach considers the motion and observation models as deeply related: the observation model is the "time to collision" for the motion model. In this paper, we use a Wiener process motion model which gives rise to an Inverse Gaussian sensory model; however this method can work with a variety of movement and sensory models as long as there is a probability density function describing the observations and an estimator, such as maximum likelihood, for its parameters.

Monte Carlo or particle filter approaches like FastSLAM [20] were designed to be computationally efficient, mapping up to thousands of landmarks while using the EKF for landmark location estimation. In our framework of intermittent observations, such a huge number of landmarks would be rare.

We used an E-M (Expectation–Maximization) based mapping approach which has been explored previously by others using various approaches. Unlike Thrun [24] who considered a discrete brute-force minimization of a cost function over a grid-based map, we considered our map lengths to be continuous and derive a formula for an approximate solution. GraphSLAM [25] optimizes a specially constructed graph with robot poses and landmarks as nodes to get the map posterior and is meant to work "off-line." We considered an "on-line" approach where the robot dynamically builds the map as it receives observations. Another approach uses a Kalman filter [23] which

attempts to keep track of the full states *in between* nodes (or observations) by taking the limit of a "dummy" observation variance to infinity. Utilizing the fact that our sensory information (edge length) is only available on every node contact and is directly related to the motor noise, we posed the problem to recovering the topological state of the robot; this has the added benefit of being more robust to problems such as loop closure.

Most SLAM implementations represent the map as a metric object but several researchers have taken a topological approach. Choset and Nagatani [7] treat the higher dimensional robot navigation space as a topology by using a Generalized Voronoi Graph (GVG) and perform localization using graph matching. Our simple sensor models generate sparse data that lends itself well to graph representation, and we attempt to simultaneously map and localize the robot on the graph using only odometry and landmark detection without appearance information. In [21], the authors demonstrate mapping using Bayesian methods and a prior over graphs. To search over the space of graphs, they use Monte Carlo sampling by starting with a random topology, proposing a modification based on a proposal distribution and then picking the new one if it improves a pre-set cost function. Bailey [3] proposes a graph theoretic approach to data association. A recent approach [12] performs SLAM on graphs using "energy" of the graph as a metric for choosing the best fitting topology and Extended Information Filter (EIF) for mapping. For all of these "model selection" issues, like data association and evaluating how good a topological fit is, we use an information theoretic criterion which rationalizes selection based on entropy considerations. A few methods [17, 19] combine both metric and topological information, composed of local feature-based metric maps connected by edges in a topology, using Kalman Filter or FastSLAM based methods. Our approach does not require metric mapping because we estimate lengths in the map as edge parameters.

The main contribution made by this paper is a multi-part algorithm that solves SLAM on planar graphs (assuming "elementary" 1- or 2-cycle topologies) including a novel loop closure approach using a model selection criterion (Sections 3–5). We verify our results and test applications of the algorithm through numerical experiments (Section 6), and address unsolved problems and opportunity for improvement (Section 7).

2 Preliminaries

2.1 Notation

We represent the topology of landmarks as a graph G with N nodes and M edges. We denote the set of all nodes as $\mathcal{X} = \{1, \ldots, N\}$. Each node $i \in \mathcal{X}$ has degree κ_i [14] and a landmark detection probability q_i which is assumed

to be known *a priori*.[1] The edge lengths between adjacent nodes are denoted by $\theta = \{\theta_1, \ldots, \theta_M\}$, and the robot's estimates are $\hat{\theta} = \{\hat{\theta}_1, \ldots, \hat{\theta}_M\}$.

As mentioned in Section 1, our SLAM algorithm runs at discrete instances $k \in \mathbb{Z}$ where each increment in k occurs when a landmark is detected. We use $x_k \in \mathcal{X}$ to denote the *node* position of the robot in the graph at instance k. The robot's (odometry) observation y_k is a variable representing the distance traveled between the events of detecting nodes at instances $k-1$ and k. The history of odometry observations is denoted $y_1^k \equiv \{y_1, \ldots, y_k\}$.

Let $\mathcal{S}$ denote the discrete set of states the robot can be in where $s_k \equiv (x_k, e_k) \in \mathcal{S}$ and e_k is the "entry" edge to node x_k. Note that $|\mathcal{S}| = \sum_{i=1}^{N} \kappa_i$. This choice of state takes into account the position and orientation of the robot in G at any instance k. We also denote the transition made by the robot at time k as t_k, the corresponding observation as y_k, the start state as $\mathrm{L}(t_k)$, and the end state as $\mathrm{R}(t_k)$. Relating to our existing notation, $\mathrm{R}(t_k) = s_k = \mathrm{L}(t_{k+1})$. Let the number of edges included in a transition be the "length" of the path $|t|$.

2.2 Odometry Measurement Error: Inverse Gaussian

When using Bayes' rule or performing parameter estimation we need to analytically express the posterior likelihood of an observation $P_\theta(y)$. This expression can be thought of as the distribution over the first passage time to a fixed distance in a random walk. This distribution is known in the literature as the Inverse Gaussian (IG) or Wald distribution [6].

We assume the robot's motion in between nodes to be a Wiener process with a variance σ^2 and a constant and strictly positive drift velocity v. Then the distribution of the first passage time is a probability density function [6]:

$$\mathcal{N}^{-1}(\tau; \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi\tau^3}} \exp\Big(- \frac{\lambda(\tau-\mu)^2}{2\mu^2\tau}\Big), \tag{1}$$

where $\mu = L/v$, $\lambda = L^2/\sigma^2$, $\tau = y_k/v$ (passage time) and L is the actual edge length associated with the observation y_k. For our purposes it makes more sense to write the pdf as a function of y_k, v and L:

$$P_\theta(y_k) \sim \mathcal{N}^{-1}(y_k; L, v, \sigma^2) = \frac{L v^{3/2} y_k^{-3/2}}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y_k - L)^2 v}{2\sigma^2 y_k}\right). \tag{2}$$

For this distribution, $P(y_k < 0) = 0$, and as noise decreases the shape looks more and more Gaussian.

[1] In practice, we would estimate q_i by many trials of having the robot run along landmarks and obtaining $\hat{q}_i = \frac{\#(\text{detected } i)}{\#(\text{passed } i)}$ where detections are triggered when the antenna deflects above a certain pre-set threshold (see Discussion).

3 Mapping with Known, Arbitrary Topology

Mapping an unknown environment along with localization make up the SLAM problem. We treat mapping first, and then localization in Section 5.

The "map" in our case consists of the graph G and its associated edge lengths $\theta(G) = \{\theta_1, \ldots, \theta_M\}$. In this section we assume G is known (and can be a general graph without any restrictions) and we find an estimate $\hat{\theta}(G)$:

$$\hat{\theta}(G) = \arg\max_{\theta} P(y_1^k|\theta; G). \tag{3}$$

In Section 4 we address the problem when G is unknown.

3.1 Perfect Data Association

When each observation can be perfectly associated to an edge length, we can do a simple ML estimate of the parameter. For this section, let θ_i be the edge length associated with the observations y_1^k. Then

$$\hat{\theta}_i = \arg\max_{\theta_i} \left(\ln P(y_1^k|\theta_i)\right).$$

Using (2) and maximizing the likelihood function above gives $\hat{\theta}$ as a function of the sample harmonic mean $\langle y \rangle$:

$$\hat{\theta}_i = \frac{1}{2}\left(\langle y\rangle + \sqrt{\langle y\rangle^2 + 4\sigma^2\langle y\rangle/v}\right), \text{ where } \frac{k}{\langle y\rangle} = \sum_{i=1}^{k}\frac{1}{y_i}. \tag{4}$$

3.2 Imperfect Data Association: E-M Approach

When data association is not deterministic (for example if the robot does not perfectly detect nodes), the observed data y_1^k are "incomplete" and we cannot directly maximize $P_\theta(y_1^k)$. The E-M algorithm [9] introduces "hidden variables" z_1^k which are chosen such that $P_\theta(y_1^k, z_1^k)$ or the "complete data" is specifiable by some distribution.

The E step computes an expectation (and effectively averages over) the hidden variables z_1^k and lets us maximize a likelihood $P_\theta(y_1^k)$:

$$\begin{aligned} Q(\theta, \theta') &= E_{Z|Y;\theta'}\left[\ln P_\theta(y_1^k, z_1^k)\right] \\ &= \sum_{l=1}^{k}\sum_{z_l}(\ln P_\theta(y_l|z_l) + \ln P_\theta(z_l))P_{\theta'}(z_l|y_1^k). \end{aligned} \tag{5}$$

The natural choice of each hidden variable is $z_l = t_l$, but this raises the issue that the space of paths t is infinite and the sum seems intractable. This problem can be solved by breaking up the sum by the possible length of the path:

$$\sum_t f(t) = \sum_{s\in\mathcal{S}} \sum_{t:\mathrm{R}(t)=s} f(t) = \sum_{s\in\mathcal{S}} \left(\sum_{\substack{t:\mathrm{R}(t)=s,\\ |t|=1}} f(t) + \sum_{\substack{t:\mathrm{R}(t)=s,\\ |t|=2}} f(t) + \dots \right).$$

The infinite sum above is suited for a breadth-first search (BFS) which is a tree traversal technique. Any planar graph G can be expanded to an infinite tree if we allow nodes to appear multiple times in this tree, by looking at connectivities in the incidence matrix of G. We can make an approximation and truncate this tree at $|t|_{\max}$ levels deep, making the set of possible paths (denoted $\mathcal{T}$) finite and the sum above computable.

By keeping track of detection probabilities q_i of each node in the tree and the sum of the edge lengths from the root node, the BFS can tell us $P(t)$ and $P(y_l|t)$. Since we only care about transitions with non-zero probability, we let $|\mathcal{T}|$ be the number of paths with $P(t) > 0$.

E-Step

We use the "forward" and "backward" algorithms from Hidden Markov Model (HMM) theory to compute $P_{\theta'}(t_l|y_1^k)$ efficiently. Let

$$c_j^{(l)} \triangleq P_{\theta'}(t_l|y_1^k) = \frac{\alpha_{l-1}(\mathrm{L}(t_l))P(t_l)P(y_l|t_l)\beta_{l+1}(\mathrm{R}(t_l))}{\sum_{s'}\alpha_k(s')}, \tag{6}$$

where j indexes into the (finite) set of paths returned by BFS, and

$$\alpha_l(s) = \sum_{t:\mathrm{R}(t)=s} P(y_l|t)P(t)\alpha_{l-1}(\mathrm{L}(t)) \tag{7}$$

$$\beta_l(s) = \sum_{t:\mathrm{L}(t)=s} P(y_l|t)P(t)\beta_{l+1}(\mathrm{R}(t)), \tag{8}$$

with $\alpha_0(s) = P(S_0 = s)$ and $\beta_{k+1}(s) = 1 \ \forall \ s \in \mathcal{S}$.[2]

M-Step

We can define a $M \times |\mathcal{T}|$ matrix $\mathbf{D}$ with $d_{ij} = \begin{cases} 1 & \text{if path } j \text{ contained edge } i, \\ 0 & \text{otherwise.} \end{cases}$

Also let L_j be the length of path t_j. To maximize we take the first derivative, and that gives us

$$\begin{aligned} \frac{\partial Q}{\partial \theta_i} &= \sum_{l=1}^{k} \sum_{j:d_{ij}=1} c_j^{(l)} \frac{\partial}{\partial \theta_i} \ln P_\theta(y_l|t_j) = \sum_{l=1}^{k} \sum_{j:d_{ij}=1} c_j^{(l)} \frac{\partial}{\partial L_j} \ln P_\theta(y_l|t_j) \\ &= \sum_{l=1}^{k} \sum_{j:d_{ij}=1} c_j^{(l)} \left(\frac{1}{L_j} + \frac{v}{\sigma^2} - L_j \frac{v}{\sigma^2 y_l} \right), \end{aligned} \tag{9}$$

[2] These computations are standard in HMM literature and derivations should be easily found in texts such as [15].

by the chain rule. For $\frac{\partial L_j}{\partial \theta_i}$, we know $L_j = \sum_{i:d_{ij}=1} \theta_i$ where the condition $d_{ij} = 1$ ensures that the derivative is not zero, and we assume that each L_j passes through θ_i no more than once.

Using all the θ_i we get M quadratic equations in M variables which are hard to solve analytically. Gradient ascent methods may fail because the likelihood function (5) may have local maxima as shown by Fig. 4.

We can get an analytical solution by making the following approximation. If we assume that the edge lengths are large compared to the motion model noise, the first term in the sum in (9) can be ignored.[3] Then we get

$$\frac{\partial Q}{\partial \theta_i} = \frac{v}{\sigma^2} \sum_{l=1}^{k} \sum_{j:d_{ij}=1} c_j^{(l)} \left(1 - \frac{L_j}{y_l}\right). \tag{10}$$

After manipulating the sums, we get a linear system of equations; if we let $U_j = \sum_{l=1}^{k} c_j^{(l)}$, $V_j = \sum_{l=1}^{k} \left(c_j^{(l)}/y_l\right)$, $\mathbf{A} = \{a_{ij}\}, a_{ij} = \sum_{m:d_{im}d_{jm}=1} V_m$, $\mathbf{b} = (b_1, \ldots, b_M)^T, b_i = \sum_{m:d_{im}=1} U_m$, and $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)^T$, we can solve for the estimates of θ by solving $\mathbf{A}\boldsymbol{\theta} = \mathbf{b}$.

Given reasonable odometry, such as would be expected with a wheeled robot on an office floor, we expect this approximate solution to be fairly close to the real solution, and, when using gradient ascent, within the region of convergence of the true solution. In future work we will establish this for our particular experimental platform.

4 Mapping with Unknown Topology (1- or 2-Cycles)

The space of planar graphs is large, so we restrict our attention to a specific, narrow class of topologies, and enumerate all possible topologies from that class—"elementary" planar graphs consisting *only* of one or two cycles, without leaves or self-loops. The former is clearly defined as a "cycle graph" in literature, and the latter is a union of two cycle graphs (at an edge chain or single vertex) which is connected.

4.1 Model Selection Using Information Theory

To select the most parsimonious model $\hat{G}$ to fit the data y_1^k we use the Akaike Information Criterion (AIC) [1] which is defined as

$$\mathrm{AIC} = 2\mathcal{K} - 2\ln(\mathcal{L}), \tag{11}$$

[3] To achieve (10), we assume $\frac{\sigma^2}{vL_j} \ll 1$. In our numerical trials (Section 6.1) this ratio is in the order of 10^{-3}. The ratio depends on the chosen robot's dynamics and may be determined by performing random trials or characterizing the dynamics [5].

where $\mathcal{K}$ is the number of model parameters (it is M in our case because there are M edges in G) and $\mathcal{L}$ is the maximized likelihood (5) for that G. A lower AIC indicates a better model. Now the algorithm picks $\hat{G}$ as

$$\hat{G} = \arg\min_{G} \left(2M(G) - 2 \ln P(y_1^k | G, \hat{\theta}(G)) \right), \tag{12}$$

where $\hat{\theta}(G)$ was found in the previous section.

4.2 1-Cycle Graphs

For simple graphs with 1 cycle we are just estimating M, and that completely specifies the graph. This is equivalent to the problem of "closing the loop," because the robot must make a decision about when it has re-visited the start node.

Our mapping procedure finds length estimates according to (3) for a preset range of M (for implementation feasibility), and then uses (12) to find $\hat{G}$.

4.3 2-Cycle Graphs

Graphs with two cycles and other more complex graphs contain *branches*, or nodes with $\kappa_i > 2$. We use this nomenclature because if we call one edge the "entry edge," there are more than 1 "exit edges," only one of which the robot will use to exit the node.

In our simple framework, we will ignore all control input and assume that the robot picks from the exit edges uniformly at random. This can easily be incorporated into the BFS mentioned before in the calculation of $P(t)$.

Enumerating 2-Cycle Topologies

One type of 2-cycle topology has two nodes of degree 3 which are connected by 3 edge chains. We will refer to these edge chains as *superedges*.

For a given total number of edges M, the problem reduces to finding the number of edges in each superedge, or the number of ways in which M can be written as a sum of 3 positive integers ($p+1$ positive integers for a p-cycle graph) which is the number of solutions to $\gamma_1 + \gamma_2 + \ldots + \gamma_{p+1} = M, \gamma_i > 0$, which is the same as the number of solutions to $\phi_1 + \phi_2 + \ldots + \phi_{p+1} = M - (p+1), \phi_i = \gamma_i - 1, \phi_i \geq 0$.

This last problem is almost the same as that of finding "partitions of an integer" which has been studied extensively such as in [2]. In the absence of a simple formula for the number of partitions, we can provide a very conservative upper bound using the "stars and bars" combinatorial argument [4] which will include solutions that are permutations of one another. The upper bound is $\binom{M-1}{p}$ solutions. To actually find these partitions a very simple recursive

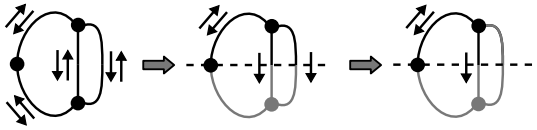

Fig. 2. Eliminating symmetric start states in a 2-cycle graph. (1) Draw a plane of symmetry between the two nodes of degree 3. (2) Discard all starting states lying on edges completely in the lower half. (3) For edges being cut by the symmetry plane, discard one of the two starting states on that edge. (4) If more than one superedge has the same number of edges, ignore duplicates.

function that enforces $\phi_i \leq \phi_{i-1}$ or similar to eliminate permuted solutions can be implemented as a computer program.

The other type of 2-cycle graph can be thought of being two distinct 1-cycle graphs joined together at a degree 4 node. These can be enumerated by finding the ways in which M can be expressed as a sum of 2 integers each greater than 1. We disallow a superedge of length 1 because that would imply presence of a self-loop.

The Start State Problem

The 1-cycle graphs has a symmetry that any choice of starting position $P(x_0 = x) = \delta(x, x^\star),\ x^\star \in \mathcal{X}$ would yield a correct map up to a cycling of the edges. However, for more general graphs, this is not true. One way to circumvent this problem is to use a uniform distribution for $P(s_0)$, which is the most general and intuitive but also less optimal for the algorithm because at the start localization results will be poor.

An alternative approach which we take is to enumerate all the possible distinct starting states for a given topology while taking symmetries into account. The number of elements in this set is usually much smaller than $|\mathcal{S}|$; a visual demonstration of how this elimination occurs is shown in Fig. 2. A start state s_0 can be represented completely by an edge and a direction (with the edge being e_0 and the direction being towards x_0) and is represented by an arrow in the figure. Then we perform E-M calculations for each of those starting states and pick the one that gives maximum likelihood. We use a similar method as in (3) but maximize for θ and s_0 together:

$$\hat{\theta}(G) = \arg\max_{\theta, s_0} P(y_1^k | \theta, s_0; G). \tag{13}$$

5 Localization

The localization problem asks to find a distribution over the current state s_k given the history of observations y_1^k. If we use E-M for mapping as described

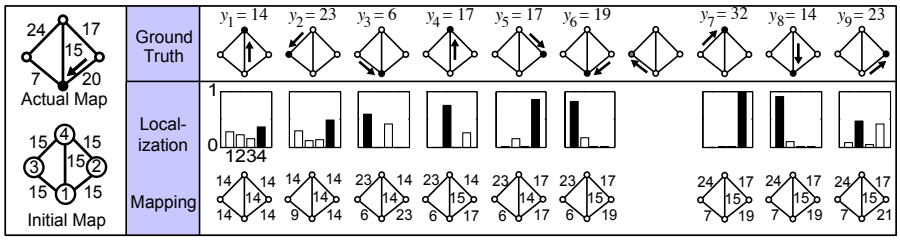

Fig. 3. The first few iterations from a numerical trial of the E-M mapping algorithm for imperfect detection. The filled circles in the "ground truth" row are the true position of the robot and the filled bars in the localization bar plots are the peaks of the pmf $P(x_k)$. Between observations 6 and 7, the robot failed to detect a node.

in Section 3.2, localization is performed implicitly in the E-step. After the computation of (6),

$$P(s_k|y_1^k) = \sum_{t:R(t)=s_k} P_{\theta'}(t|y_1^k). \tag{14}$$

If we are not using E-M (such as with perfect association in Section 3.1) or if we want to explicitly perform localization, we can use the following:

$$P(s_k|y_1^{k-1}) = \sum_{t:\mathrm{R}(t)=s_k} P(t)P(\mathrm{L}(t)|y_1^{k-1}) \qquad \text{(prediction)} \tag{15}$$

$$P(s_k|y_1^k) \propto \left(\sum_{t:\mathrm{R}(t)=s_k} P(y_k|t)P(t) \right) P(s_l|y_1^{k-1}) \qquad \text{(update)}\ . \tag{16}$$

These equations alternately carry out a time-update (before observation y_k) and measurement-update (after observation y_k) and proceed iteratively.

6 Numerical Tests

6.1 Mapping Lengths with E-M

Figure 3 shows a simple trial run of the algorithm in Section 3.2 in which the robot attempts to map the edge lengths and localize in a known topology with imperfect association of observations with edges. We used a uniform distribution for $P(s_0)$, and the initial length guesses are shown in the leftmost column of the figure. We chose $|t|_{\max} = 5$ for this trial (and others in this section), and for this choice $\sum_j c_j^{(l)}$ was 1 without need for renormalization.

At every "branch point" the exit edge is picked uniformly at random and node detection is imperfect, resulting in imperfect data association. The given map has a symmetry so that if it is, for example, "flipped" about the vertical

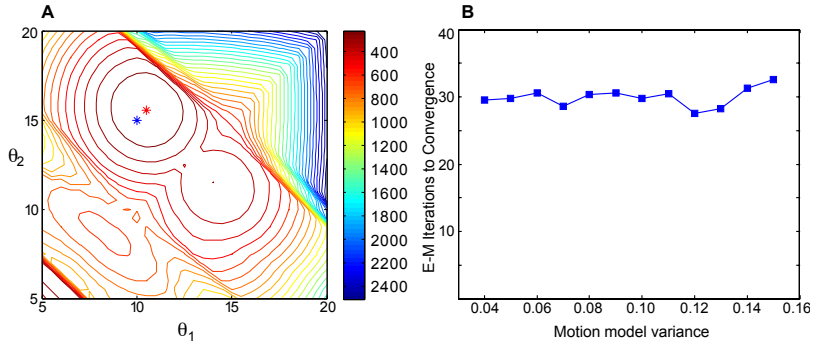

Fig. 4. A: Contours of the likelihood function (5) showing multiple maxima. The blue star is the true value of the parameters and the red star is the solution found using the "noise approximation" in (10). The red star is close to the peak of the global maximum showing that the approximation was valid. B: When the E-M algorithm finds the correct solution, the number of E-M iterations required in total is roughly independent of the motion noise.

or horizontal, we still get the same graph. The ground truth and developed maps have been aligned for the reader's convenience.

It is evident that localization results are poor initially because of the developing map and the initial uniform distribution, but it performs better after a few iterations. After each iteration localization is performed iteratively from s_0 using (15) and (16), so the localization results may be drastically different from the previous belief after the map is updated. We can only hope to get good localization results after the developed map is perfect.

Figure 4A justifies the assumption made in (10) by showing that for sufficiently small noise, the peak of the approximated likelihood function (red star) closely matches that of the exact function (blue star) given in (9). As noise is increased, the red star diverges from the blue star and at each E-M iteration, the maximum obtained by solving (10) will not necessarily maximize the true likelihood function. Since the maximization is key to convergence of E-M, we suspect that sufficiently large noise will cause E-M to fail and degrade the performance of our algorithm.

The figure also shows the presence of local extrema in the likelihood function making a gradient ascent method difficult. This trial was for a simple 2-edge cycle so that the likelihood function can be visualized.

In general for E-M, no bounds can be given on the rate of convergence, but Fig. 4B empirically shows that the total number of E-M iterations required to find the solution is roughly independent of the motion noise, which is important from an implementation point of view.

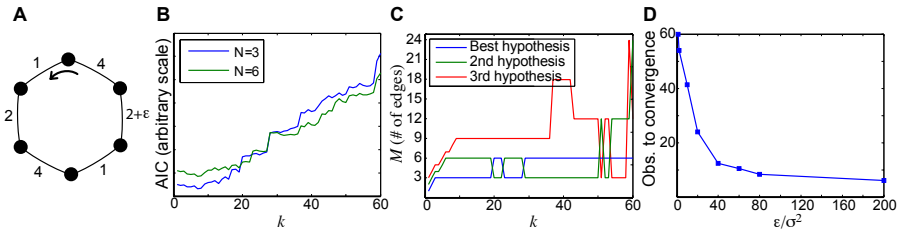

Fig. 5. An illustration of loop re-opening using the AIC model selection criterion. A: The simple (but almost ambiguous) map given to the robot. B: The AIC of the hypotheses for a 3-edge and 6-edge cycle evolving over time. C: The top three hypotheses at any period during the same trial. D: The average convergence time to the correct hypothesis as a function of ϵ/σ^2.

6.2 Loop Closure (and Re-Opening)

With respect to "closing the loop," Fig. 5 demonstrates the advantage of using the method described in Section 4.1: the robot can modify a loop closure decision as long as we do not discard the history of observations. In this trial the robot is performing SLAM in an unknown topology with imperfect association.

The robot is given a simple cycle map of six edges with the first three edges almost of equal length to the last three (with one of the three being perturbed by an ϵ). It initially picks $M = 3$ as the best hypothesis after the trial starts, but after going around more times and receiving more observations it corrects its hypothesis and chooses $M = 6$ as the most parsimonious model. This behavior can be explained at a higher level by the following argument.

When the robot does not have much information about the map, the "perturbation" can be attributed to noise, so that the lower order model is sufficiently good at predicting the observations. From the asymptotic normality of ML estimators [10], we know that $\mathrm{var}(\hat{\theta})$ falls as $1/\sqrt{k}$. So as k gets bigger the lower order model gives a much poorer fit to the data than the higher order model.

The time required for the algorithm to decide to favor the higher order (but correct) model over the lower order model depends on the perturbation, and this "convergence time" is plotted in part (D). The minimum number of observations to support a $M = 6$ model is six, and so there is a horizontal asymptote at 6 as ϵ gets larger and larger. The motion model variance was $\sigma^2 = 0.05$ for these trials.

6.3 Dynamic Environment

Another problem which pertains to model selection for us is the problem of "disappearing landmarks" where a previously existing landmark is taken away, and the similar problem of "new landmarks" where a newer landmark is

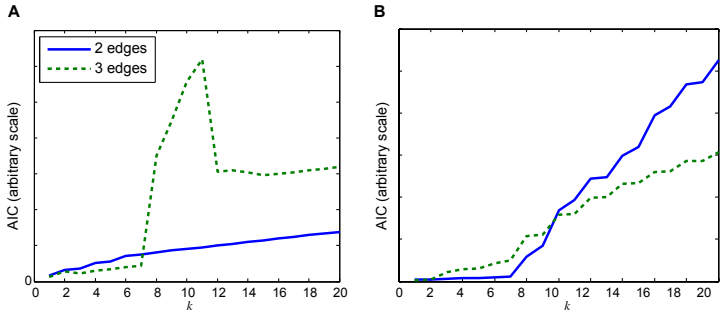

Fig. 6. The A: "new landmark" problem where a node is added to a 2-edge cycle map and B: "disappearing landmark" problem where a node is deleted from a 3-edge cycle, as handled by our algorithm. In both the plots, the change in the environment occurs at iteration 7. For the disappearing landmarks problem, one possibility is that the robot attributes the missing landmark to imperfect detection and continues to favor the higher-order model (which helps in case the landmark *re*-appears). In this trial we set detection probabilities $q_i \approx 1$ so that this did not happen.

inserted in the environment. By comparing the AIC of hypotheses which have M close to the best model the robot can make "soft" or modifiable decisions about the nature of its environment (Fig. 6).

Note in Fig. 6A that the 2-edge model adjusts its length estimates to considerably lower its AIC around iteration 12, but it still cannot match the 3-edge model. In Fig. 6B it takes about three iterations for the 2-edge hypothesis to beat the 3-edge hypothesis.

6.4 Topology Enumeration and Selection: 2-Cycle Graphs

We used the AIC again to help the robot pick the best topology according to (12). Due to calculation costs, the robot searched only the space of 2-edge-

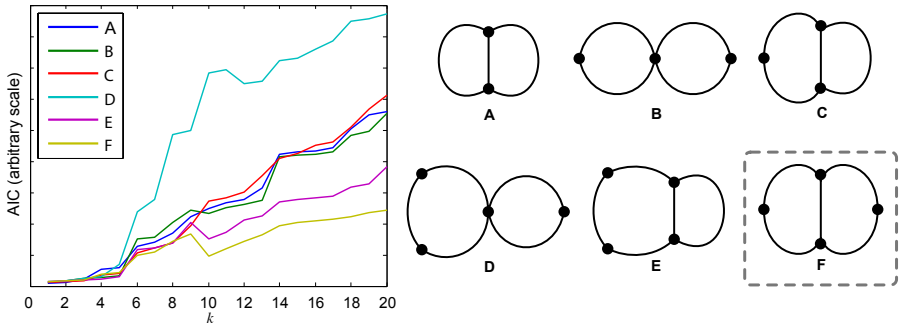

Fig. 7. The robot picks the best (2-cycle) topology fit for some observations using AIC. The true topology was "F," i.e. the rightmost one in the second row.

connected planar graphs with two cycles, having $3 \leq M \leq 5$. The candidate topologies are shown in the right half of Fig. 7. For each of these "edge-partition" hypotheses, there were a number of possible starting states to be taken into account (see Section 4.3). The robot assumed perfect detection for this particular trial.

7 Discussion

This paper presents an algorithm to perform SLAM on elementary graphs. We divide the mapping into two parts, known and unknown topology. Our algorithm solves the former in the most general case using E-M, and presents techniques of finding the topology from a very small subset of planar graphs. We pick this subset to be graphs with one or two cycles, but this could be easily extended to other families of graphs as long as they can be parametrized and enumerated.

This method, while still targeting simple environments and using minimal sensory information, can address some well known SLAM problems [11]. We present numerical experiments demonstrating how the algorithm solves the "loop closure" problem without requiring appearance information in a non-parametric way by using an information theoretic model selection criterion. We also present numerical experiments illustrating the solution to the problem of dynamic environments by maintaining multiple hypotheses.

To apply our framework to the real world, several issues would have to be addressed. First, we need a better way to characterize various landmarks and their corresponding detection probabilities. Although the antenna sensor is capable of capturing finer details of a landmark, we have chosen to use only sparse sensor input (a binary flag for detection) which gives rise to problems such as "misses" and "false positives" (Section 6.3) which correspond to the sensor signal being below or above (resp.) a pre-set threshold. This approach would take many trials to give an accurate detection probability and the detection probability for one landmark would most likely be different from the rest of the landmarks. In the future, we plan to incorporate a richer sensor model of the antenna to (a) correct "misses" or "false positives" for cases in which the signal is close to the threshold, (b) predict q_i based on how "close" the signal magnitude is to the threshold the first time a landmark is observed, and (c) incorporate landmark appearances in our algorithm which may make data association simpler in more complex environments.

Second, some parts of the algorithm, such as maintaining multiple hypotheses for a large number of possible topologies as well as enumerating to find graph structures, require large computations. Methods of graphical inference from data [16] as well as approximate methods that take assumptions about the structure, such as the "topology improvement algorithm" in [22], should be explored. We plan to compute the complexity bounds of our al-

gorithm and compare the performance with our approach to other existing approaches such as EKF-SLAM in large environments.

A natural extension to the algorithm presented here would be to devise a method of mapping a general planar graph. Our algorithm targets indoor 1-dimensional environments (corridors), but we imagine using our algorithm with GVG's [7] to map higher dimensional environments; extensions to the algorithm to handle "leaves" will naturally have to be made. Furthermore, with the framework in this paper, we can perform SLAM on any parameter (e.g., landmark appearance) which can be associated with edges on a graph that has states as nodes, as long as we can define a probability distribution for observations and an estimator of its parameters.

Acknowledgments

The authors would like to thank the anonymous reviewers for helpful comments on elucidating approximations and the rationale behind our approach, and also Donniell Fishkind for helpful discussions on graph theory. They would also like to thank Nick Keller.

References

1. H. Akaike. A new look at the statistical model identification. *IEEE Trans. Autom. Control*, 19(6):716–723, 1974.
2. G. E. Andrews. *The theory of partitions.* Addison-Wesley, Reading, MA, USA, 1976.
3. T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments.* PhD thesis, Department of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, 2002.
4. A. T. Benjamin and J. Quinn. *Proofs that Really Count: The Art of Combinatorial Proof (Dolciani Mathematical Expositions).* The Mathematical Association of America, August 2003.
5. J. Borenstein and L. Feng. Measurement and correction of systematic odometry errors in mobile robots. *IEEE Trans. Robot. Autom.*, 12:869–880, 1996.
6. R. S. Chhikara and J. L. Folks. *The inverse Gaussian distribution: theory, methodology, and applications.* Marcel Dekker, Inc., New York, NY, USA, 1989.
7. H. Choset and K. Nagatani. Topological simultaneous localization and mapping (slam): Toward exact localization without explicit localization. *IEEE Trans. Robot. Autom.*, 17:125–137, 2001.
8. M. Csorba. *Simultaneous Localisation and Map Building.* PhD thesis, Department of Engineering Science, University of Oxford, 1997.
9. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
10. W. DuMouchel. On the asymptotic normality of the maximum-likelihood estimate when sampling from a stable distribution. *Annals of Statistics*, 1(5):948–957, 1973.

11. H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robot. Autom. Mag.*, 13(2):99–108, 2006.
12. J. Folkesson and H. I. Christensen. Closing the loop with graphical SLAM. *IEEE Trans. Robot.*, 23(4):731–741, 2007.
13. M. W. M. Gamini Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. Robot. Autom.*, 17(3):229–241, 2001.
14. F. Harary. *Graph theory*. Addison-Wesley, Reading, MA, USA, 1969.
15. F. Jelinek. *Statistical methods for speech recognition*. MIT Press, Cambridge, MA, USA, 1997.
16. M. I. Jordan, editor. *Learning in Graphical Models*. Cambridge MA: MIT Press, 1999.
17. K. Kouzoubov and D. Austin. Hybrid topological/metric approach to SLAM. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 872–877, April 2004.
18. J. Lee, S. N. Sponberg, O. Y. Loh, A. G. Lamperski, R. J. Full, and N. J. Cowan. Templates and anchors for antenna-based wall following in cockroaches and robots. *IEEE Trans. Robot.*, 24(1):130–143, Feb. 2008.
19. B. Lisien, D. Morales, D. Silver, G. Kantor, I. Rekleitis, and H. Choset. Hierarchical simultaneous localization and mapping. *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1:448–453, Oct. 2003.
20. M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.
21. A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *IEEE Trans. Robot.*, 22(1):92–107, 2006.
22. L. Shi, A. Capponi, K. H. Johansson, and R. M. Murray. Network lifetime maximization via sensor trees construction and scheduling. In *Third International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks*, Annapolis, MD, USA, Jun 2008.
23. B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry. Kalman filtering with intermittent observations. *IEEE Trans. Autom. Control*, 49:1453–1464, 2004.
24. S. Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous Robots*, 15(2):111–127, 2003.
25. S. Thrun and M. Montemerlo. The graph SLAM algorithm with applications to large-scale mapping of urban structures. *Int. J. Robot. Res.*, 25(5–6):403–429, 2006.