# Performing Inference over Topometric Maps

**Victoria Preston** [*1] **Christopher Bradley** [*1]

## Abstract

Autonomous navigation necessitates building *maps* to represent configurations of the world to plan robot agent trajectories. Generally, these maps are constructed from prior mission data or while the agent is exploring. Traditional methods use LiDAR and similar sensors to create dense structure maps; however this poses a significant and expensive data association challenge. Using sparse detections of structure in an environment (e.g., corners) can greatly reduce the scale of the data association problem, and yield a useful semantic representations for planning known as topometric maps. This project examines a Bayesian inference method based on Metropolis-Hastings Markov-Chain Monte Carlo (MH-MCMC) for extracting the most likely topometric map from a series of noisy landmark detections first proposed by Ranganathan et al. (Ranganathan & Dellaert, 2004) called Probabilistic Topological Maps (PTMs). We analyze the performance of the PTM MH-MCMC sampler, and propose an extension of the work with respect to a specific type of landmark detection.

## 1. Introduction

Imagine finding an office in the Stata Center: as you search, you'll be keeping track of what rooms you've seen and where they were, what intersections you came across, and what decisions you made at those intersections in order to backtrack. For robotic agents, this task is known as the simultaneous localization and mapping (SLAM) problem. Advancements in SLAM for robot agents have generally produced dense *feature maps* in which the metric positions of structural features in an environment are observed with sensors like LiDAR. Multiple sensor measurements are built into a coherent map though *data association* which includes sophisticated filtering and smoothing techniques.

There is significant interest in generating useful map representations with camera images alone; Tesla openly holds an anti-LiDAR, pro-camera stance and recent work shows that a trained neural network and camera images can effectively replace LiDAR (Wang et al., 2018). For small-scale navigation challenges (e.g., office robots, industry campuses), vision-only techniques in SLAM have generally relied on a process called Bundle-Adjustment, akin to a large, non-linear least-squares problem. While impressive benchmarks have been reached for self-localizing within a given map, the actual maps produced by dense feature tracking remain difficult to plan with. Building map representations also suffer from scaling challenges, including memory allocation for search in the data structures and complexity of "fixing" data association errors.

Alternatives to structure-based map building are generally referred to as *landmark SLAM*. These SLAM systems eschew visual features and instead rely on detections of sparse "landmarks" for constructing topological maps of an environment. What these landmarks actually are isn't necessarily important to the algorithms that track them, and can be anything from actual beacons with range information, to physical objects detected via an object detector. A useful map from these sparse detections will encode not only the metric information about these landmarks (i.e., position), but also semantic information that may be of significance to the environment (i.e., a "doorway" label). One example would be detecting corners in a polygonal environment to generate an efficient structure graph.

Outside of this course, we've been considering the general problem of generating sparse navigable maps with image data *online* (while the agent navigates in real-time). The goal of this project is to examine the use of Bayesian estimation in generating topological maps from image-based landmark detections by close examination of the work on *Probabilistic Topological Maps* (PTMs) (Ranganathan & Dellaert, 2004; 2005; Ranganathan et al., 2006; Ranganathan, 2008). We present our specific problem in Sec. 2, our implementation of PTMs in Sec. 3, our own extension which we call covPTMs in Sec. 5, and discuss the outcomes of the project in Sec. 7.

[*]Equal contribution [1]Department of Aeronautics and Astronautics, MIT, Cambridge, Massachusetts, USA. Correspondence to: Victoria Preston <vpreston@mit.edu>, Christopher Bradley <cbrad@mit.edu>.

*Proceedings of 6.435*, Cambridge, 2019. 

## 2. Project Background

We're keen to answer questions about robot navigation in polygonal environments (e.g., indoors, urban environments) solely using camera data. In particular, we ask: can landmarks, in the form of *corners* in polygonal environments be sufficient input to infer the underlying structure of an environment? Corners are particularly interesting to us because the perspective in which a corner is viewed can imply the location of walls (viewing a "closed" corner implies two walls, viewing an "open" corner implies that there is one wall and a gap/discontinuity). For this project, we made use of a simulator and *learned corner detector* which were produced in conjunction with research outside of the course. We present it in brief to explain our data generation method and highlight what modifications we made to complete this project.

In order to replicate the problem of a physical robot navigating in the real world, we designed a simulator that integrates with the UNITY game engine to imitate navigation in a polygonal world Fig. 1. Our robot is simply a mobile panoramic camera that travels along a user-specified trajectory through an environment composed only of ground, sky, and walls.

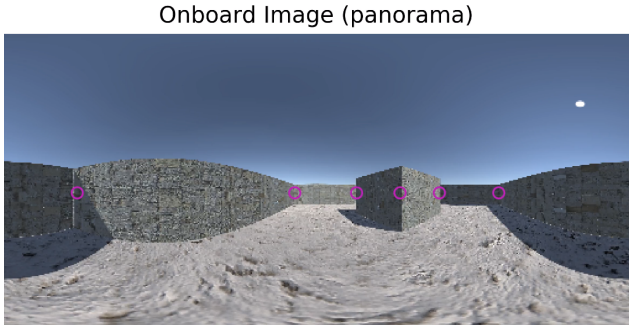



*Figure 1.* Screencap from the simulator showing the panoramic view. Detected corners are highlighted with purple circles.

As the robot traverses its path, it assumes that its position perfectly tracks with the path it was given. The ground truth position of the robot is influenced by additive Gaussian noise, which causes growing error between the robot's believed state and the true state. This drifting pose estimate generally adds to the challenge of data-association in map-building.

The output of the panoramic camera is fed into a convolutional neural network (CNN) to identify corners (defined as the intersection of two walls) within sight of the robot. The output of this "landmark detector" is a set of range and bearing estimates from the robot to every vertex in the polygonal environment within direct line of sight. As many detections are made each over the course of a mission, we sort raw detections from the sensor into "coherent" vertex groups by measuring the Mahalanobis distance between a raw detection and the mean/covariance of a vertex group. When detections are sorted into existing groups, the mean location and covariance of that group is updated via an Extended Kalman Filter (EKF).

In a world with perfect sensing, the groups formed using this filtering technique would be exact representations of every corner in the environment. However, measurements generated by the sensor are noisy or occasionally faulty and the robot pose estimate is also noisy. The robot may also view the same corners at different times in a trajectory. In combination, this results in many false-positive corner detections which we would like to merge into coherent vertices in the structure graph. Ultimately our data analysis goal is to *find the most likely set of vertices for a structure graph, given a set of noisy vertex detections*.

## 3. Probabilistic Topological Maps

The method that we emulate in this project to infer the most likely set of vertices from noisy detections is presented by Ranganathan and colleagues in several papers and a thesis, all of which we drew upon for this project (Ranganathan & Dellaert, 2004; 2005; Ranganathan et al., 2006; Ranganathan, 2008). Their method, Probabilistic Topological Maps (PTMs), focuses on designing a distribution over "topologies" (sets of landmark detections) and finding the most likely topology using a Metropolis-Hastings Markov-Chain Monte Carlo (MH-MCMC) sampler. In their work, landmarks are detected when the robot is "co-located" with the landmark site, and the detection is encoded with the vehicle's odometry and an image of the location (referred to as "appearance" data in the papers). We will primarily focus on odometry-based information about landmarks in this project.

The use of an inference framework is motivated by the idea of *landmark ambiguity*; as the robot navigates, the sensors may drift or landmarks may "look" indistinguishable. The latter problem is relevant in *loop closure* scenarios, in which the robot will actually revisit landmarks, but perhaps from a different vantage point or with accumulated position estimate error. Associating a new visit with an old visit has always been a core challenge in SLAM regimes. To choose the most likely map representation from a set of detections would generally require an exhaustive state space search, where the space of possible topologies is combinatorial in the number of detections. It also stands to reason that the posterior over possible topologies is intractable to compute directly, ultimately motivating an MCMC sampling regime.

To frame the sampler, a topology is ultimately defined as a set-partition. Landmark detections can be grouped together into sets. Each set represents a "true" landmark that will be

| Symbol | Definition |
|---|---|
| $N$ | Number of detections |
| $M$ | Number of distinct landmarks, $M \leq N$ |
| $Z$ | Set of detections, $Z = \{Z_i \| 1 \leq i \leq N\}$ |
| $S$ | Set of detections |
| $T$ | Topology, $T = \{S_j \| j \in [1, M]\}, \cup_{j=1}^{M} S_j = Z$ |
| $O$ | Landmark detection measurement |
| $X$ | Distinct landmark locations |

*Table 1.* Key notation for sampler derivation.

**Algorithm 1** Metropolis-Hastings Sampler

**Input:** topology $T_t$, data $Z^t$, number of samples $N$
**repeat**
  Propose new topology $T_t'$ with $Q(T_t'; T_t)$
  Calculate acceptance ratio

$$a = \frac{P(T_t'|Z^t)}{P(T_t|Z^t)} \frac{Q(T_t; T_t')}{Q(T_t'; T_t)}$$

  With probability $p = \min(1, a)$ accept $T_t'$ and $T_t \leftarrow T_t'$.
**until** $N$ samples drawn

put in the graph as a node. Edges in the graph are drawn based upon the order of landmark detections encapsulated in each graph node; therefore the sampling only serves to associate landmark detections. Key notation that will be used throughout this section is defined in Table 1.

### 3.1. MH-MCMC Sampler

In order to draw samples in the space of topologies, a MH-MCMC sampler is proposed, Algorithm 1. To calculate the acceptance ratio, both a *proposal distribution* and *posterior distribution* needs to be defined. The next sections give an overview of the definition of these distributions, summarized from the relevant literature.

#### 3.1.1. Proposal Distribution

The proposal distribution $Q(\cdot;\cdot)$ consists of two equally likely types of "moves": a *split* and a *merge*. The probability of any specific split or merge move is entirely a function of the number of sets in a topology, and the number of detections sorted into each set (effectively a matter of combinatorics). The pseudo-code for the proposal distribution is presented in Algorithm 2 where both the proposed topology $T'$ and the proposal ratio $r$ are returned.

A proposed merge move selects two sets at random within $T$ and puts the combined detections into a single new set. This reduces the number of sets in $T$ by one $(M-1)$. If $T$ is already composed of a single set, then no merge can occur. For a proposed split move, a random non-singleton set in $T$ is selected, and randomly split into two new sets. This increases the number of sets in $T$ by one $(M+1)$. If $T$ only contains singleton sets, no split can occur. To calculate the proposal ratio for either a merge or split, $r$ in Algorithm 2, we must consider the number of merges that can occur, $N_m$, the number of splits, $N_s$, and the number of ways a single set could be split. $N_m$ is the binomial coefficient $\binom{M}{2}$ where $M > 1$ and where in the merge move, $M$ is the number of sets in $T$ and in the split move, $M$ is the number of sets in $T'$. $N_s$, is the number of splits that are possible in a topology; for a merge move this is the number of non-singleton sets in $T'$, and in a split move this is the number of non-singleton sets in $T$. Finally, when we select a set to split, we must consider the number of ways that the chosen set could be split. This can be calculated as a Stirling number of the first kind, which we write as $\left\{ {X \atop Y} \right\}$.

**Algorithm 2** Proposal Distribution

**Input:** topology $T$
Choose with probability 0.5 a *merge* move, otherwise select *split* move
**if** merge move selected **then**
  **if** only one set in $T$ **then**
    **Return** $T, 1$
  **else**
    Randomly choose sets $R$ and $Q$ from $T$
    Let $P = R \cup Q$
    Set $T' = (T - R - S) \cup P$
    $r = N_m (N_s \left\{ {|P| \atop 2} \right\})^{-1}$
    **Return** $T', r$
  **end if**
**end if**
**if** split move selected **then**
  **if** only singleton sets in $T$ **then**
    **Return** $T, 1$
  **else**
    Randomly choose non-singleton set $P$
    Randomly split $P$ into two sets $R$ and $Q$
    Set $T' = ((T - P) \cup R) \cup Q$
    $r = N_m^{-1} N_s \left\{ {|P| \atop 2} \right\}$
    **Return** $T', r$
  **end if**
**end if**

#### 3.1.2. Posterior Distribution

The posterior distribution, $P(T|Z) \propto P(Z|T)P(T)$ (by Bayes Rule), ultimately requires a likelihood function on the observed detections given a topology and a prior on topologies. Functionally, Ranganathan et al. assume a non-informative uniform prior over the space of topologies; however in the thesis it is suggested that a Poisson prior

over the number of total landmarks could be used instead. In our emulation, we assume the non-informative uniform prior.

To find the likelihood of landmark detections given a topology, Ranganathan suggests that marginalizing over the distinct landmark locations is necessary:

$$P(Z|T) = \int_X P(Z|X,T)P(X|T) \tag{1}$$

which further requires that we place a prior over the landmark locations given the topology, and express the likelihood of the data with respect to both the landmark locations and the topology. The prior over landmark locations given a topology is expressed as a "penalty" function which encodes the idea that distinct landmarks will *not* lie close together:

$$-\log P(X|T) = \sum_{1 \le i < j \le N} f(||X_i - X_j||) \tag{2}$$

where $X_i$ and $X_j$ are not members of the same set in $T$, and $f(\cdot)$ is defined as a cubic function parameterized by a *maximum radius* and *maximum penalty* where any two points that fall within the maximum radius of each other will be penalized according to the cubic function with peak at the maximum penalty, and points that fall outside will receive no penalty (value 0).

The likelihood $P(Z|X,T)$ is used to capture two key ideas: distinct landmark locations according to the topology should not lie very far from the actual detections, and landmark locations that lie within the same set should lie close to one another. This is expressed as:

$$-\log P(Z|X,T) = \left(\frac{||X - X^{(o)}||}{\sigma_o}\right)^2 + \sum_{S \in T} \sum_{i,j \in S} \left(\frac{||X_i - X_j||}{\sigma_t}\right)^2 \tag{3}$$

where $\sigma_o$ and $\sigma_t$ are the error in observed landmark detections and estimated topology we will accept, respectively and $X^{(o)}$ is a vector of the detected landmark locations. The quantity $X$ is particularly key: $X$ is *not* the observed landmark locations, it is a vector that represents what the observed landmark locations *should* be based upon a given topology. It is not necessarily straightforward to estimate this vector, so it must be approximated. Ranganathan et al. use a nonlinear least-squares optimization algorithm (Levenberg-Marquardt) to find $X*$, which minimizes Equation 3. A Gaussian distribution is then proposed over $X*$:

$$Q(X|Z,T) = \frac{1}{\sqrt{|2\pi\Sigma|}} e^{-0.5(X-X*)^T \Sigma^{-1} (X-X*)} \tag{4}$$

where $\Sigma$ is a product of the least squares optimization. To ultimately approximate Equation 1, a Monte Carlo approximation (importance sampler) is used:

$$\int_X P(Z|X,T)P(X|T) \approx \frac{1}{R} \sum_{i=1}^{R} \frac{P(Z|X^{(i)},T)P(X^{(i)}|T}{Q(X^{(i)}|Z,T)} \tag{5}$$

where $R$ is the number of samples generated from $Q(X|Z,T)$, $X^{(i)}$ are the samples, and the calculation for the probability distributions follow from Eqn. 1, Eqn. 2, and Eqn. 4.

### 3.2. Implementation Practicalities

We implemented PTMs based on Ranganathan's work from scratch in Python 2.7. The only departure we made from the implementation described is in which minimization algorithm we use to find $X^*$: we substitute LM minimization with the BroydenFletcherGoldfarbShanno (BFGS) algorithm built-in to the Python library *scipy*.

There was some notation ambiguity throughout the paper. In particular, the second term of Equation 3 was never explicitly explained to reference the topological landmark locations; however we assume that to make the use of a nonlinear optimization regime worthwhile that this must be the case. The penalty function is also never explicitly defined, however is said to be a cubic function. We defined a quadratic penalty function of the form $-\frac{M(X^2-D^2)}{D^2}$ where $M$ is the maximum penalty to assign, and $D$ is the maximum radius.

We make special note that values for sampler parameters $\sigma_o$, $\sigma_t$, and $R$ are never given in the literature. We selected our values for these parameters based on qualitative performance for our illustrative scenarios, and performed a small parameter sweep over a few chosen values to get an idea of how they impacted sampler performance. We note the values that we use in Section 4.

## 4. Results

In this section we present some interesting results from a test environment that illustrates the capabilities of the sampler. First, we show results for the "perfect" data case. Then, we will add noise to our odometry and vertex detector and iterate through potential parameters

### 4.1. Performance on Perfect Data

To demonstrate that the sampler can converge to the correct topology given "perfect" data, we ran our simulator without

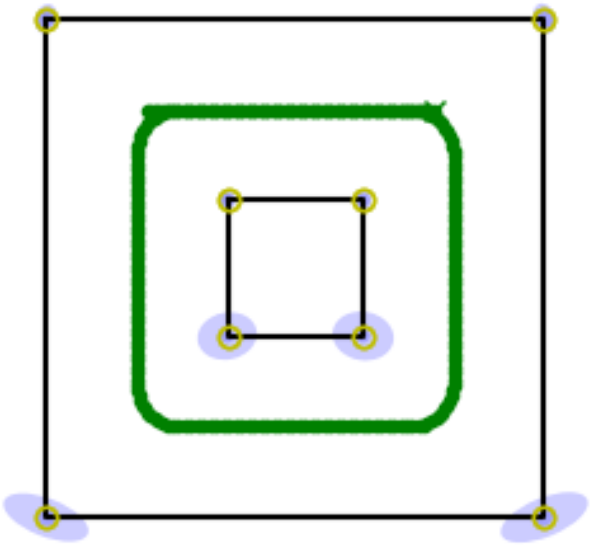


*Figure 2.* Here we show the output of the simulator on "perfect data." The locations of the vertices are given by the yellow circles, and the path of the robot is given by the green line, ending at the $x$. The walls are shown as black lines, and the covariances of the vertices are represented by the blue ellipses. Note that in this figure, each vertex was actually detected two separate times due to the robot's path, so at each vertex location, there are actually two vertices, perfectly overlapping each other so that the dimensionality of the topology before sampling is 16.

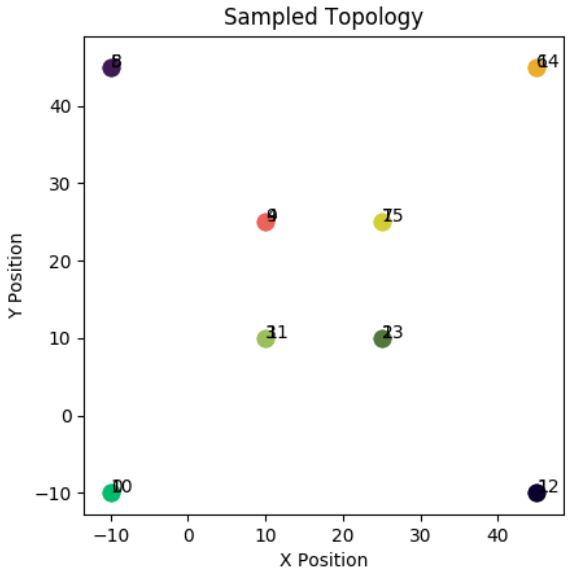



*Figure 3.* Most sampled topology for the "perfect" data. Each pair of vertices that share the same location are assigned to the same cluster (in the figure, this corresponds to the dots being the same color, though that is hard to see with the samples overlapping perfectly). Out of 1000 samples, this topology was sampled 535 times.

any noise in either the position of the robot or the position of the vertex from the detector. This map is shown in Fig. 2. We then ran the sampler on the resulting sixteen vertices, and since the simulation was noiseless, each vertex shared it's position exactly with one other. As expected, the most likely topology matches the ground truth, as shown in Fig. 3. Several runs of the sampler on this data with different parameter values for $\sigma_o$, $\sigma_t$, $R$, *maximum radius* and *maximum penalty* showed that the results were robust to changes in these values, as to be expected when the correct topology is obvious with no ambiguity. With proof of concept that the sampler worked on "perfect" data, the next step was to expand our analysis to more interesting cases.

### 4.2. Performance on Noisy Data

In order for our analysis to be consistent, we ran one simulation using the our noisy simulator, then performed a parameter sweep by running the sampler many times over the same resulting proposal topology for different configurations of parameters. The map can be seen in Fig. 4. The three most likely cluster maps, a histogram of the relative amount of samples for those maps, and a trace plot of the sample dimensionality from these simulations can be seen in Fig. 7. We generally start with a proposal distribution in which all detections are unique sets. We found that for certain values of these parameters the most-likely map that was returned was indeed what we knew to be the ground-truth map.

In practice, however, we found that the regime was extremely sensitive to the initialization of the parameters. The literature gave no guidance on the settings for these parameters, but did present an illustrative analysis on the settings for the max penalty value. Their work illustrated that this value had an impact on the total number of nodes in their final graphs. We found in our analysis that the maximum range value also had a non-trivial impact on the dimensionality of the output proposal: if the max range for the penalty function is too small, there was little incentive to merge vertices and the sampler rarely accepts merge proposals outside the radius. We similarly found that the noise parameters, $\sigma_o$ and $\sigma_t$ also impacted the rate of accepted merges: the lower the acceptable noise/error the more conservative the sampler was towards accepting merge moves. The converse was also found: the higher the acceptable noise/error the more likely that seemingly unrelated nodes would be acceptably grouped.

The full page figure demonstrates the sensitivity of parameter selection. First, we show the results of sampling using parameters that allow the sampler to converge to what we would consider the "correct" topology. From that starting point, we show the results of raising and lowering each of the parameter values. The last figure on the page shows what

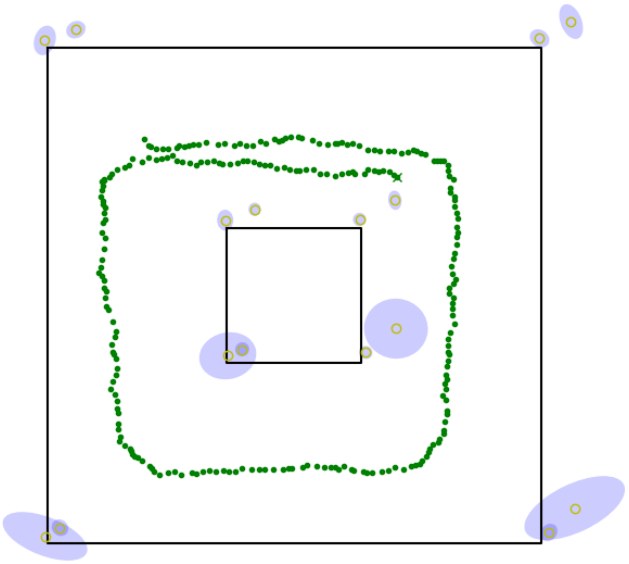


*Figure 4.* The result of the simulation run with noisy data. Here we see there are sixteen distinct vertices with differing levels of uncertainty. In reality, each ground truth vertex was seen twice, and should be grouped together. The sampler will sample different clusters, and hopefully converge to this topology.

happens if the initial proposed topology groups all vertices into one cluster, instead of having them all begin in their own cluster. Even with "good" parameter values, the sampler was not able to converge to the correct topology in this case, despite sampling 10,000 times. From a combinatorics perspective, this isn't particularly surprising: it is harder to choose a series of good splits that lead to a workable dimensionality than to simply try a fer alternate merges. At the limit of time, we would anticipate that the sampler could overcome an adversarial proposal topology. In general, the burn-in and mixing characteristics shown in the trace plots seem to indicate how certain parameter settings impact the ease of which the sampler can move through the distribution and could be indicative of mis-matched parameter settings to the input data.

## 5. Method Extension

In Ranganathan's work, it is generally assumed that the landmark detector triggers when directly interacting with a landmark. In our proposed problem, we detect our landmarks from afar, and we have the ability to estimate the measurement error in our range and bearing. In general, our landmark sensor has more accurate bearing, but noisy range seeking. A pathological example is shown in Fig. 5 where the gray ellipses show the covariance of our landmark detections, and the red stars show the estimated metric location of the landmark.

In this example, if we were to consider the metric locations alone, we may identify two clusters; one in the lower left and one in the upper right. However, the covariance implies that the "right" landmark associations in this case would be made along the angle of the covariance distribution. This is clearly a case where PTMs would fail without modification.

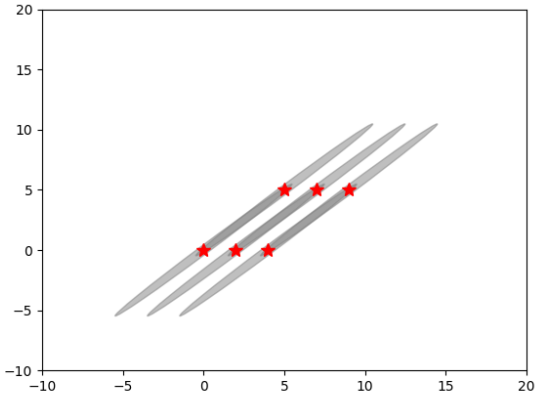
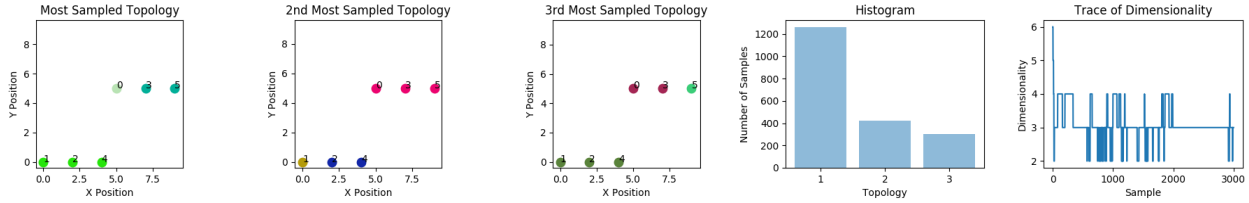



*Figure 5.* Detection example where the red stars indicate metric location and the gray ellipses represent covariance in the range estimate. Generally we can assume that our bearing estimate is quite good, so the "right" vertex associations would be along the angle of the covariance.

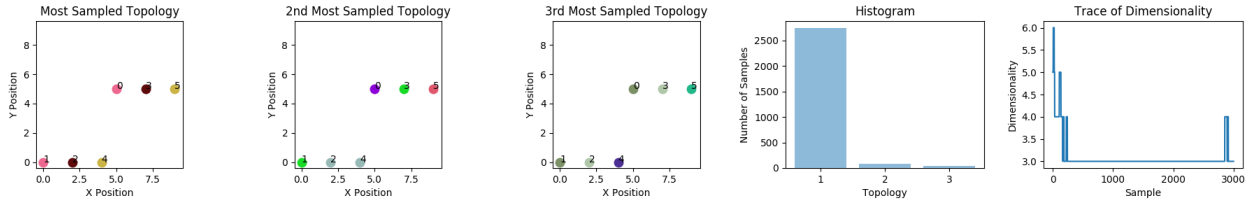
To include measurement variance, we modify the posterior distribution by re-examining Eqn. 3. In particular, the use of metric distance ought to be replaced. Mahalanobis distance, which is the distance between a point and a distribution, seems like a natural fit in this setting. Moreover, we also consider adding a term to the likelihood that reflects the "goodness" of a grouping with respect to the actual measurements by including Bhattacharyya distance, which is the distance between two distributions:

$$-\log P(Z|X,T) = \left(\frac{|m(X, X^{(o)}, \Sigma^{(o)})|}{\sigma_o}\right)^2 + \sum_{S\in T}\sum_{i\in S}\sum_{j\in S}\left(\frac{|m(X_i, X_j^{(o)}, \Sigma_j^{(o)})| + |b(X_i^{(o)}, X_j^{(o)}, \Sigma_i^{(o)}, \Sigma_j^{(o)})|}{\sigma_t}\right)^2 \tag{6}$$

where $m(\cdot,\cdot,\cdot)$ is a function which calculates the Mahalanobis distance between an input point $X$ and the distribution of the observed point $X^{(o)}$ with covariance $\Sigma^{(o)}$ and $b(\cdot,\cdot,\cdot,\cdot)$ is a function which calculates the Bhattacharyya distance between two input points and respective covariances. For completeness, we also change the calculation of the penalty function so that the distance measure is the Mahalanobis distance, rather than Euclidean distance.




(a) 3000 samples generated by the method presented in Ranganathan et al.




(b) 3000 samples generated by the method extension presented in Eqn. 6

*Figure 6.* Comparison of the the method presented in Ranganathan and the extension in this paper. The metric-only method tends to jump between associating the top right and bottom left groups, and other subsets. In contrast, the extended method in this paper strongly favors grouping the samples along the covariance lines.

A comparison of the sampler performance on the adversarial world is shown in Fig. 6. Generally, the outcome of the extension is to highlight potential limitations of a metric only approach.

## 6. Project Logistics

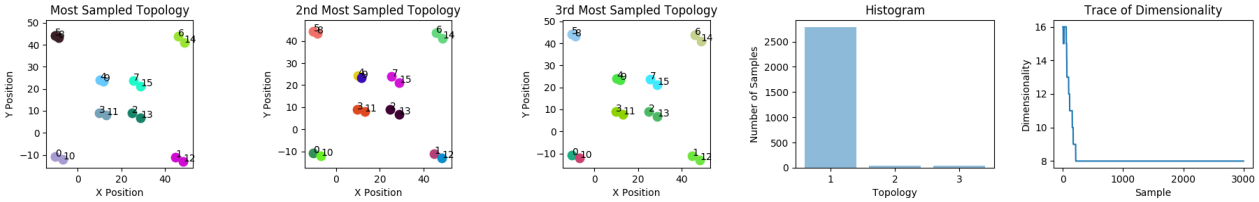
We've implemented everything in Python. Both members were involved with all aspects, but in particular Victoria generally implemented the sampler and extension, and Chris took lead in adapting the simulator and running diagnostics. Based upon the statement of objectives for the project, we met our goals of replicating the approach in the paper, expanding the analysis of the sampler, and extending the method to exploit covariance information. Stand-alone code (without simulator) for this project has been posted at cp-bradley/v_preston_c_bradley_6_435_project.git.

## 7. Discussion

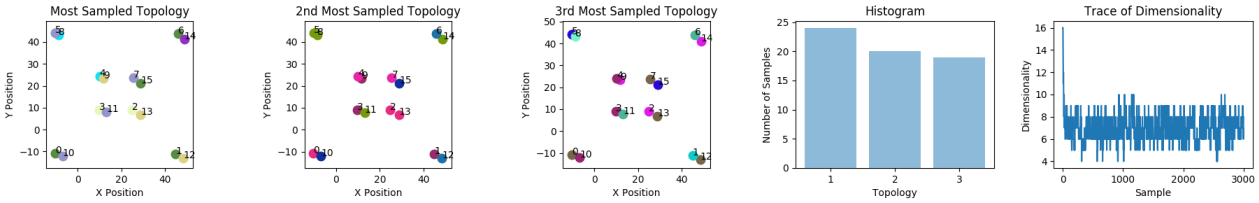
As was revealed over the course of this project, using PTMs on metric odometry data alone is useful for generating metric clusters, but fails to resolve data ambiguities or consider additional information that may be useful (such as measurement covariances). The extension we presented in Sec. 5 seeks to address one potential way of resolving metric ambiguity when only odometric information is available. In later works, Ranganathan et al. seek to address some ambiguities by adding a secondary observation type, image data, to the model (Ranganathan et al., 2006). By adding a term for processed image data, ambiguity in location proximity can be resolved by adding a term that penalizes when two landmark observations don't "look" similar. One case in which this is particularly powerful is when navigating in an indoor environment: though you may be metrically close to another landmark, perhaps there is a wall that physically occludes the two. These detections would be merged if we only considered metric information, but by "seeing" that there is a wall, we can prevent degenerate associations like this from occurring in the proposed topology.

On the outset of this project, we proposed to implement a Reversible-Jump MCMC (RJMCMC) method to compare against Ranganathan's sampler; however on further reading, we found that the derivation of RJMCMC for set partitions presented in Green's seminal paper (Green, 1995) was directly mirrored in the method proposed by Ranganathan et al. Interestingly, Green's work is not cited in any of the papers or the thesis, which was a missed opportunity to include more theoretical discussion about the sampler performance. What we show in this project is some initial evaluation of the sampler for this problem, in which we reveal that the sampler is incredibly sensitive to the parameter settings; indeed without a principled way of setting these parameters, the methodology is extremely limiting. In practice, we also experienced extremely variable outcomes based upon initialized topology proposal, indicating that certain parts of the distribution on topologies is exceedingly difficult to access, and somewhat at the mercy of random selections on splits and merges, which is a combinatorial process.

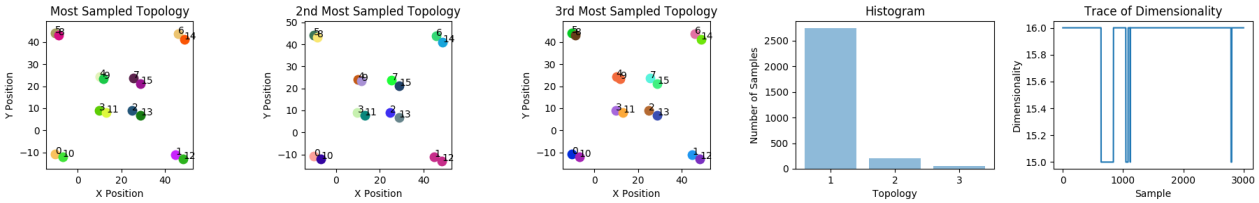
Looking forward, this project helped us thoroughly assess PTMs for our specific research question. While we will ultimately not elect to use the sampler as presented by Ranganathan, we hope to incorporate some of the extensions we explored in this project to our research.



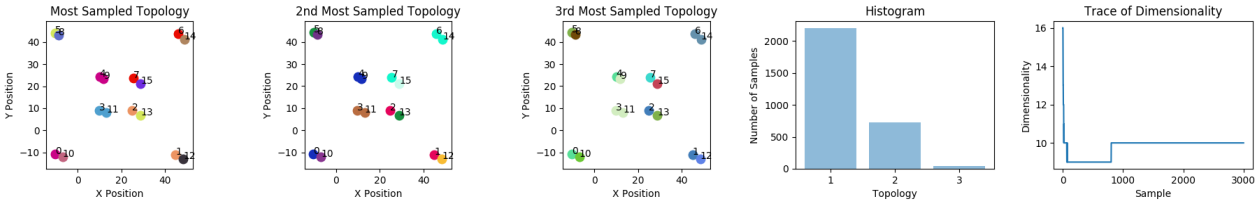

(a) Sampler converges to correct topology. Parameters: $\sigma_o = 1$, $\sigma_t = 1$, *maximum radius* = 10 and *maximum penalty* = 100



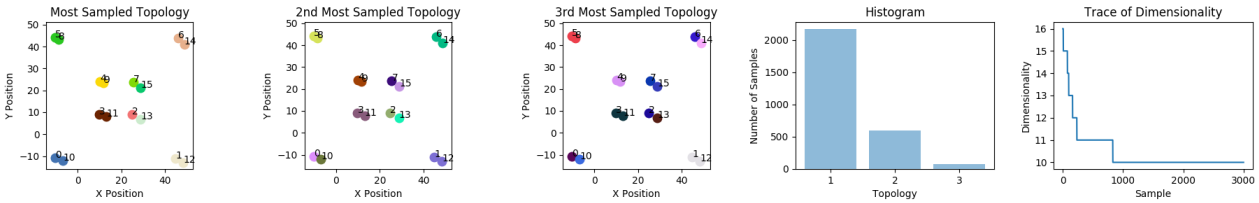

(b) Sampler groups vertices it shouldn't. Parameters: $\sigma_o = 1$, $\sigma_t = 1$, *maximum radius* = 50 and *maximum penalty* = 100



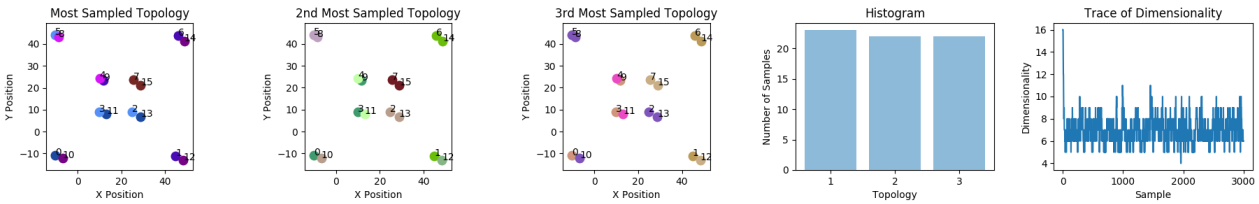

(c) Sampler doesn't want to group anything. Parameters: $\sigma_o = 1$, $\sigma_t = 1$, *maximum radius* = 1 and *maximum penalty* = 100



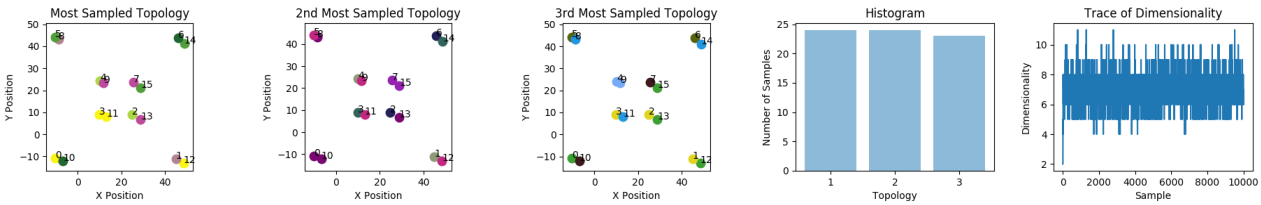

(d) Sampler gets trapped in local minimum. Parameters: $\sigma_o = 5$, $\sigma_t = 5$, *maximum radius* = 10 and *maximum penalty* = 100




(e) Penalty for leaving clusters unmerged is too low. Parameters: $\sigma_o = 1$, $\sigma_t = 1$, *maximum radius* = 10 and *maximum penalty* = 10




(f) High penalty value over-incentivises merging. Parameters: $\sigma_o = 1$, $\sigma_t = 1$, *maximum radius* = 10 and *maximum penalty* = 1000




(g) Same parameters as fig a, but initial proposed topology is all vertices clustered in one group. Dimensionality prevents convergence.

*Figure 7.* Data for different values of parameters from the same simulator run. Only the first set of parameters allow convergence to the correct topology. Demonstrates the importance of parameter tuning in this approach.

## References

Green, P. J. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

Ranganathan, A. *Probabilistic topological maps*. PhD thesis, Georgia Institute of Technology, 6 2008. An optional note.

Ranganathan, A. and Dellaert, F. Inference in the space of topological maps: An mcmc-based approach. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 2, pp. 1518–1523. IEEE, 2004.

Ranganathan, A. and Dellaert, F. Data driven mcmc for appearance-based topological mapping. Georgia Institute of Technology, 2005.

Ranganathan, A., Menegatti, E., and Dellaert, F. Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*, 22(1):92–107, 2006.

Wang, Y., Chao, W.-L., Garg, D., Hariharan, B., Campbell, M., and Weinberger, K. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. *arXiv preprint arXiv:1812.07179*, 2018.