Robust and Efficient Robotic Mapping

Summary of 2008 MIT PhD Thesis

Edwin Olson

1 Introduction

Mobile robots are dependent upon a model of the environment for many of their basic functions. Locally accurate maps are critical to collision avoidance, while large-scale maps (accurate both metrically and topologically) are necessary for efficient route planning. Solutions to these problems have immediate and important applications to autonomous vehicles, precision surveying, and domestic robots.

Building accurate maps can be cast as an optimization problem: find the map that is most probable given the set of observations of the environment. However, the problem rapidly becomes difficult when dealing with large maps or large numbers of observations. Sensor noise and non-linearities make the problem even more difficult— especially when using inexpensive (and therefore preferable) sensors.

This thesis describes an optimization algorithm that can rapidly estimate the maximum likelihood map given a set of observations. The algorithm, which iteratively reduces map error by considering a single observation at a time, scales well to large environments with many observations. The approach is particularly robust to noise and non-linearities, quickly escaping local minima that trap current methods. Both batch and online versions of the algorithm are described.

In order to build a map, however, a robot must first be able to recognize places that it has previously seen. Limitations in sensor processing algorithms, coupled with environmental ambiguity, make this difficult. Incorrect place recognitions can rapidly lead to divergence of the map. This thesis describes a place recognition algorithm that can robustly handle ambiguous data.

We evaluate these algorithms on a number of challenging datasets and provide quantitative comparisons to other state-of-the-art methods, illustrating the advantages of our methods.

2 Loop Closing

2.1 Problem Statement

Without the ability to recognize previously-visited places, the position uncertainty of a robot increases without bound due to the ceaseless accumulation of dead-reckoning error. Place

recognitions serve as constraints on the motion of the robot, allowing a correction of its dead-reckoning error. In the mapping context, place recognition is called "loop closing": if the robot drives in a loop and back to its starting position (and recognizes it), the robot "closes" the loop. The quality and accuracy of the map is a function of the quantity and quality of the loop closures.

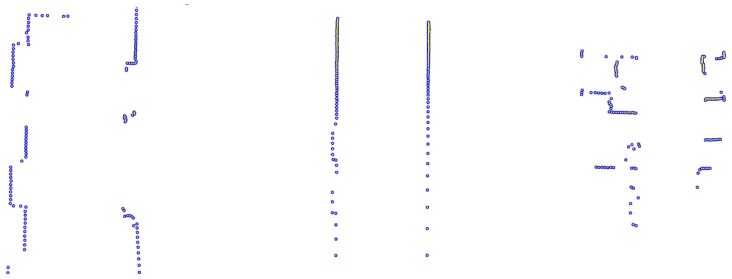

Figure 1: Three laser scans from the CSAIL dataset. Some environments present rich alignment cues (left: an elevator lobby). Incorrect matches arise from both spartan areas (middle: a corridor) and from repetitive/cluttered areas (right: office cubicles).

Loop closing is difficult for several reasons. Algorithms must be robust to modest changes in the environment caused by things like moving chairs or humans. Sensing limitations play a critical role. The data produced by laser scanners, for example, is metrically accurate (see Fig. 1), but it is often difficult to distinguish one room from another: indoor environments tend to be composed of sets of straight walls and corners whose appearances are similar.

Camera-based systems, like those using the SIFT feature detector, provide a richer description of the environment (able to distinguish different posters on a wall, for example), but are still susceptible to ambiguity. Different offices, for example, may contain the same type of chair, and different outdoor environments contain the same types of street signs. The length of these feature vectors has a major impact on both total memory usage and the time required to index and search for similar features. Consequently, it is desirable to use shorter descriptors (such as PCA-SIFT). Reducing the descriptor size generally increases the error-rate of matching: matching algorithms that are robust to higher error rates are thus very desirable.

In a map-building context, place recognition is known as "loop closing", reflecting the common case in which a robot travels around a large loop and returns to its original position (thus "closing" the loop). However, any place recognition adds an edge to the pose graph, creating a new cycle— the robot does not need to physically travel in a circle in order for a loop closure to occur.

In robotic mapping applications, place recognition has generally been cast in terms of explicitly associating landmarks with previously observed landmarks. This process is called "data association". Data association on one (or just a few) features at a time is highly susceptible to errors, and data-association errors can cause catastrophic divergence of the map.

In this thesis, we describe an approach that attempts to close loops by performing a number of simple pose-to-pose matches. Given a number of naive matches ("hypotheses"), our goal is to identify those that are correct. A *set* of these hypotheses has an interesting property: the correct hypotheses all agree with each other (since there is only one true

configuration), whereas the the incorrect hypotheses tend to be incorrect in different ways (and thus do not agree with each other). Our algorithm exploits this property by computing the subset of hypotheses that is most self-consistent. In comparison to the exponential cost of performing data association amongst N features, the cost of our approach is $O(N^2)$ in the number of hypotheses. Our approach is similar to CCDA, except that our notion of consistency is not limited to discrete boolean quantities.

Our approach does not require landmarks to be tracked, and thus does not require covariance information about the landmarks or a data association algorithm. All that is required is the ability to compute possible rigid-body transformations between two sensor scans. Our approach can be viewed as a outlier-rejection step, extending previous pose-to-pose methods.

A major advantage of our proposed method is that it computes the "second-best" set of mutually-consistent hypotheses as well. When using RANSAC or Joint Compatibility Branch and Bound (JCBB), the second-best solution is generally a trivial variation on the best solution: it is thus not very informative. In contrast, the second-best solution reported by our new approach is *orthogonal* to the best solution– i.e., represents a substantially different interpretation of the data. Our method allows the quality of these two solutions to be quantitatively compared, allowing ambiguous solutions to be safely discarded. (If the data can be equally well explained in two different ways, it is not safe to trust either explanation.) In these ambiguous cases, our method will occasionally reject hypotheses that are correct. Rejecting correct hypotheses has an impact on the quality of the final map (since some information is lost), but it does not result in the sort of rapid divergence that accepting an incorrect hypothesis can cause.

2.2 Approach summary

Given a set of N hypotheses, we compute the pair-wise consistency for each pair, yielding an $N \times N$ "consistency" matrix A. We now wish to find the subset of hypotheses that is maximally self-consistent. In other words, we wish to find a subset of hypotheses whose pair-wise consistency is, on average, the greatest.

A hypothesis set can be viewed as a graph, with each hypothesis represented as a node. The (weighted) adjacency matrix of this graph is the pair-wise consistency matrix A.

Our solution is based on our previous work, an algorithm called "Single Cluster Graph Partitioning" (SCGP) . SCGP is a graph partitioning method that attempts to find a single cluster of well-connected nodes, rejecting other nodes.

Let v be an $N \times 1$ *indicator vector* such that $v_i = 1$ if the i^{th} hypothesis should be accepted, and $v_i = 0$ otherwise. The sum of the compatibilities between the accepted hypotheses is $v^T A v$, and the number of accepted hypotheses is $v^T v$. Thus, the average pair-wise consistency $\lambda(v)$ for a subset of hypotheses v is simply:

$$\lambda(v) = \frac{v^T A v}{v^T v} \tag{1}$$

2.3 Results

We evaluated our loop-closing method on both vision-based and laser-based datasets. Our vision data set is a 1.9km trajectory comprised of three complete loops in an outdoor urban

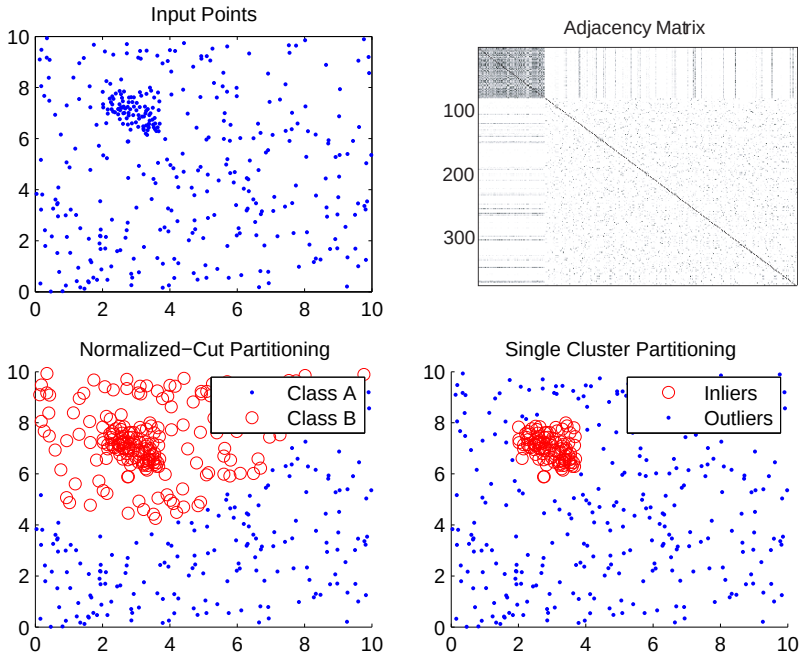

Figure 2: SCGP versus Normalized Cuts. Given a set of (x, y) points, each representing a hypothesis (top left), the goal is to identify the subset of hypotheses that forms the maximally self-consistent cluster. The adjacency matrix is plotted such that the desired points have small indices: their higher-than-average pair-wise consistency is clearly visible (top right). Our method, SCGP, produces the desired segmentation (lower right). Normalized-Cuts, also based on spectral clustering, uses a different clustering metric which results in a poor solutions on this type of problem.

environment. Our lidar-based data sets include standard map building benchmark datasets, as well as several new real and synthetic datasets.

Loop closure algorithms are difficult to evaluate quantitatively: there are no standardized datasets for this purpose. The benchmark datasets that we have processed are composed of raw sensor data, but because sensor processing methods vary between researchers, the resulting sets of hypotheses are not the same. Naturally, the quality and reliability of the sensor processing (and hypothesis-generating) systems has huge impact on the difficulty of the loop-closing problem.

We address this problem in two ways. First, we evaluate the performance of our system repeatedly on the same dataset, with the quality of hypotheses purposefully degraded to varying degrees. This allows us to determine the point (if any) at which our algorithm can no longer produce useful output. Second, our hypothesis datasets have been made available so that other researchers can perform a direct comparison to our method.

3 Graph Optimization

3.1 Problem Statement

In this section, we describe our approach to optimizing pose graphs. The nodes of a pose graph represent particular places in the environment: these nodes can represent physical

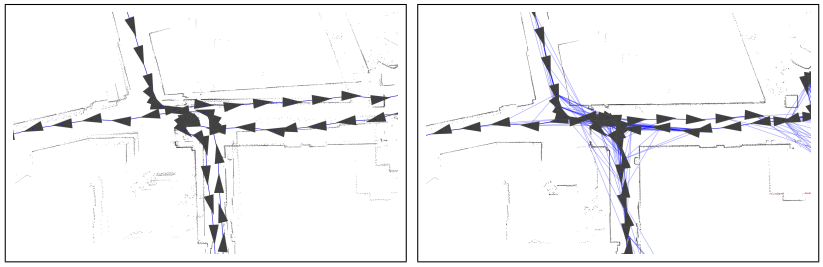

Figure 3: CSAIL loop closure close-up. Before loop-closing (left), laser scans show significant discrepancies. After recognition (right), not only are the discrepancies resolved, but the topological relationships are discovered.

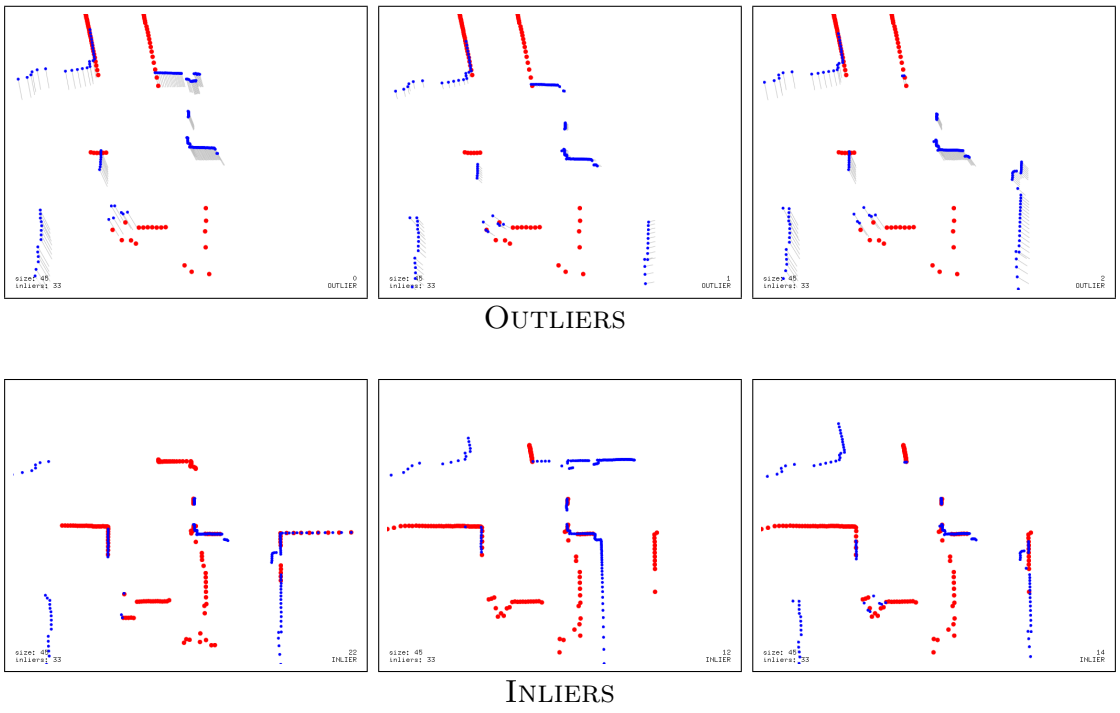

Figure 4: Filtered loop-closure hypotheses. Twelve representative hypotheses from a hypothesis set of size 45 are displayed. The outliers (top) and inliers (bottom) were automatically labelled using Single-Cluster Graph Partitioning. The alignment errors (derived from the posterior) are shown as gray lines: these errors are apparent in the outlier set.

things (such as landmarks that have been sensed by the robot) or can represent the current or previous position of the robot. The edges of a pose graph relate the positions of two nodes and describe a cost as a function of the deviation from their preferred configuration. Pose graph optimization is the task of computing positions for each node such that the edges have minimum cost.

This optimization problem is critically important: the lowest-cost solution represents the best (maximum likelihood) map of the world. Whether a robot is avoiding obstacles, planning routes to some goal, or is explicitly trying to map an environment, the robot is dependent upon the fast and reliable optimization of the pose graph.

However, pose graph optimization is difficult for three central reasons:

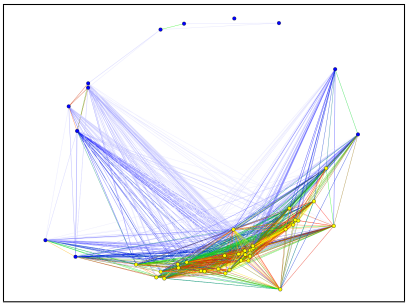

Figure 5: Adjacency graph for a set of 45 hypotheses. Each node represents a pose-to-pose hypothesis: the inlier set is indicated by yellow nodes (towards the bottom), with outliers in blue. Brightly-colored edges indicate greater pair-wise compatibility. The distance between two nodes is roughly proportional to the joint probability of the two hypotheses.

- The state space is large. The size of the state vector is proportional to the number of nodes in the graph; a problem with thousands of state elements is *small*.

- The constraints are non-linear. As a result, the cost surface contains optimization hazards like local minima and valleys.

- The initial estimate is often poor. Most robots have meager dead-reckoning accuracy—as a result, the initial configuration of the pose graph can be far away from the best configuration.

This thesis describes a new optimization method that is both fast and robust. It can rapidly find the best configuration of nodes, and is robust to optimization hazards that trap existing methods. Our method also has low memory requirements and is relatively easy to implement.

Our method adapts Stochastic Gradient Descent (SGD) , commonly used in training artificial neural networks, to the problem of pose graph optimization. SGD has not previously been applied to map optimization, but has been used for localization.

We use a novel state-space representation, solving for the global-relative motions between poses, rather than solving for the absolute positions of the nodes. This representation allows faster reductions of error at each iteration, while a specialized data structure allows each iteration to be computed very quickly.

We demonstrate that our approach is not only very fast (competitive with or faster than other state-of-the-art methods), but is also significantly more robust: it finds correct solutions in cases that other methods get stuck in local minima.

The basic version of the algorithm operates in "batch" mode; we also describe an incremental version that is well-suited to online operation. We introduce the idea of spatially-varying learning rates as a way of controlling the impact of new loop closures. It allows stable parts of the graph to continually refine their solution while simultaneously allowing volatile parts of the graph to quickly incorporate new information.

3.2 Approach Summary

Our approach is motivated by a simple intuition. Consider a pose graph constructed by a robot as it travels around a large loop. The trajectory of the robot is periodically sampled, with each sample point represented by a pose (a "node") and connected to its predecessor by an estimate of the robot's motion (an "edge"). Note that every edge has an uncertainty associated with it, so it is possible to compute its χ^2 error.

Suppose the robot returns to its starting place and recognizes it; this adds an additional edge between the first and last pose, creating a cycle (see Fig. 6). This edge is commonly called a "loop closure", since it closes a large loop of nodes.

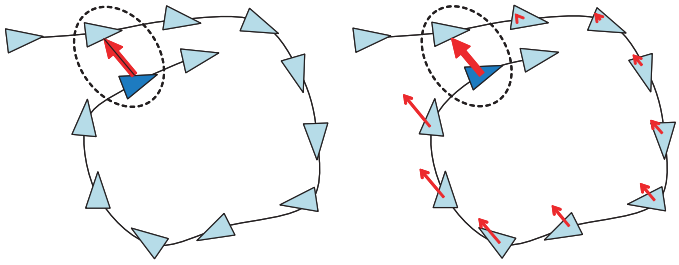

Figure 6: Map Optimization Intuition. The χ^2 error of a graph with a poorly-satisfied constraint (left, red arrow) can be reduced by distributing small adjustments around the nodes in the loop (right).

The choice of state representation has a large impact on the performance of our optimization algorithm. Picking a state space in which the state variables echo the structure of the underlying random process results in significant performance improvements.

We propose a new state space representation that approximates the cumulative motion of the robot, but eliminates the projective effects of purely relative representations. We call it the "global incremental" state space representation, x_{incr} : the state vector contains the difference between the successive global positions. In the absence of rotations ($\theta_i = 0$), it is identical to the relative state space. With this state space representation, there are no projective effects (the position of a node is a simple sum of global motions, not the composition of rigid-body transformations), and so updates are more stable.

Consider the χ^2 error of a single constraint i, evaluated at the current state estimate plus d:

$$\chi_i^2 = (J_i d - r_i)^T \Sigma^{-1} (J_i d - r_i) \tag{2}$$

The gradient is:

$$\frac{\partial \chi_i^2}{\partial d} = 2 J_i^T \Sigma_i^{-1} J d - 2 J_i^T \Sigma_i^{-1} r_i \tag{3}$$

Evaluated at the current state estimate, where $d = 0$, gives just:

$$\frac{\partial \chi_i^2}{\partial d} = -2 J_i^T \Sigma_i^{-1} r_i \tag{4}$$

At time step t, let the learning rate λ be $1/t$. The gradient descent step moves in the opposite direction of the gradient, and can be written as:

$$d = \lambda 2 J_i^T \Sigma_i^{-1} r_i \tag{5}$$

Algorithm 1 Stochastic Gradient Descent Algorithm

$\lambda = 1/3$
while not converged **do**
  select a constraint i at random
  compute J_i and r_i at the current state estimate.
  $d = 2\lambda J_i^T \Sigma_i^{-1} r_i$
  $x = x + d$
  $\lambda = \lambda/(\lambda + 1)$
end while

At each iteration, Stochastic Gradient Descent (see Alg. 1) reduces the error of a single constraint (often unintentionally increasing the error of other constraints). In early iterations, it will take large steps in order to reduce the residual, which causes the state estimate to jump around the state space. This behavior allows SGD to escape local minima. Intuitively, while many constraints may be satisfied at a local minimum, there are generally one or more constraints that remain poorly satisfied. These ill-satisfied constraints will cause a comparatively large jump away from that local minimum. Conversely, near the global minimum, constraints tend to be better satisfied: this means smaller jumps. In short, ill-satisfied constraints cause the state estimate to jump around, but the state estimate tends to "stick" near the global minimum once it is found.

3.3 Online/Incremental Approach

In order to extend our method to the online case, we allow different parts of the graph to have different learning rates. When a new constraint is added to the graph, the learning rate can then be selectively increased. This allows unaffected parts of the graph to retain their low-error configurations while volatile parts of the graph are allowed to rapidly find a new equilibrium.

Specifically, instead of a single global learning rate λ, we will give each node i its own learning rate Λ_i. These learning rates represent the volatility of each node in the graph. The key contribution is a set of rules describing how these learning rates should be adjusted in order to accommodate new information. The batch algorithm can be described as a special case of the incremental algorithm: in the absence of new observations, the learning rates Λ_i will all equal the learning rate λ of the batch algorithm.

3.4 Results

P5K20K is a synthetically generated dataset with 5000 nodes and 20000 edges. This gives it a fairly high average node degree, which, with the fairly large size of the state vector, makes it is a challenging optimization problem (see Fig. 7 and Fig. **??**).

The Cholesky factorization is fairly slow on this problem due to the fairly high average node degree and relatively large sensing range. The result is that the Cholesky factor is an order of magnitude more dense than on the CSW dataset: the information matrix has a similar density (0.086%), but the Cholesky factor has a density of 1.96% (versus 0.166% on

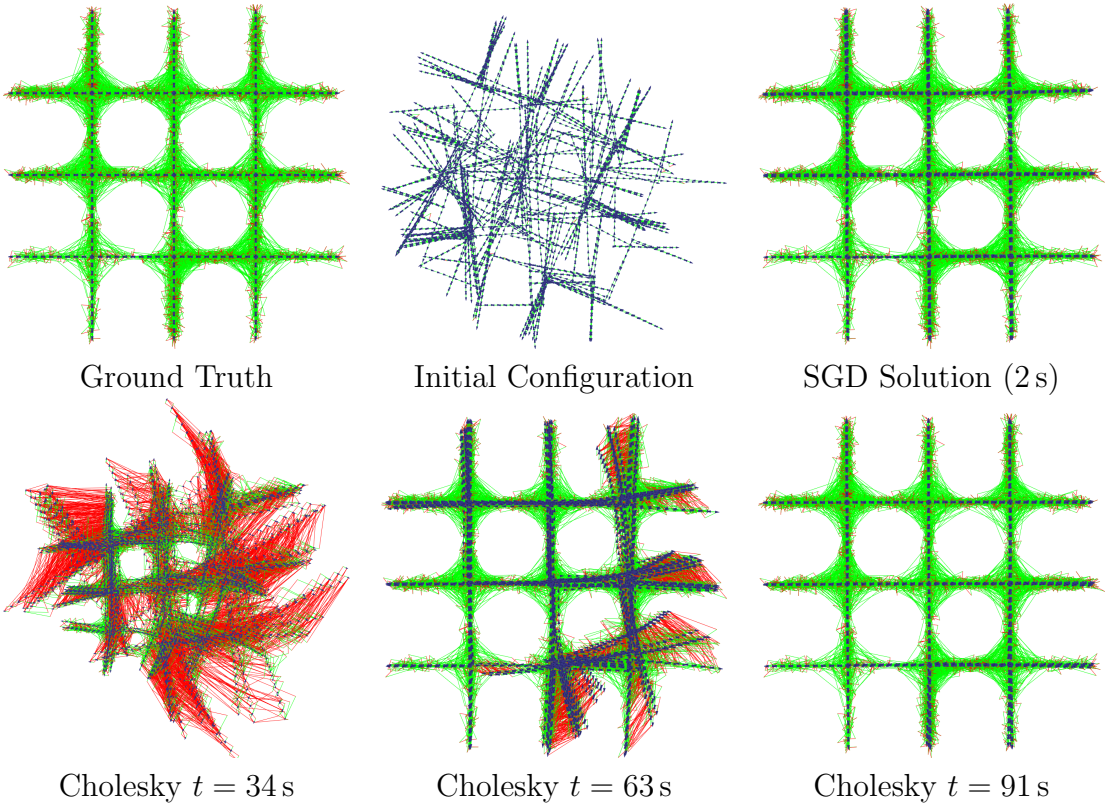

Figure 7: P5K20K Graphs. The true configuration (top left) and an odometry-derived initial configuration (top middle) are shown. The configuration resulting from three successive iterations of Cholesky Decomposition are shown (bottom, left to right). After 91 seconds, its solution is comparable to our solution (top right), which was obtained in 2 seconds.

the CSW dataset).

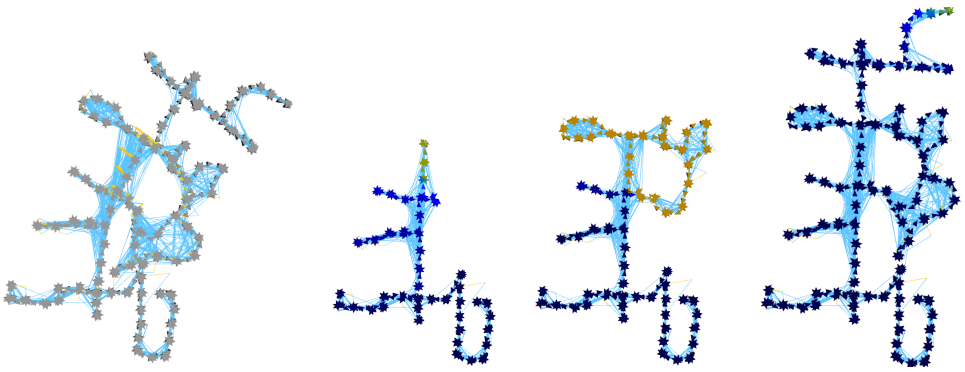

Figure 8: Incremental Processing of Freiburg dataset. The open-loop graph (top-left) is incrementally optimized; the state of the graph is shown at two intermediate configurations and the final configuration. The colors used in the maps indicate the learning rates Λ_i, which are also plotted on the bottom as a function of pose number; earlier parts of the graph are clearly insulated from learning rate increases affecting poses closer to the robot. When closing large loops (middle-left figure), the learning rate is increased over a larger portion of the graph.

Compared to other algorithms, our proposed method is substantially more robust to

initialization error. Specifically, our method can find the global minimum even when other methods get stuck in local minima.

This robustness can be measured. The likelihood of an algorithm getting stuck in a local minimum increases with the noise of the observations in the graph: noisy observations are more likely to cause the algorithm to erroneously step in a sub-optimal direction. We can test the robustness of an optimization algorithm by generating a large number of synthetic datasets with varying degrees of noise, and counting the number of graphs that each algorithm successfully solves. We detect failure by comparing the normalized χ^2 to 1.0, its expected value at the global minimum: convergence to significantly larger values indicates a local minimum.

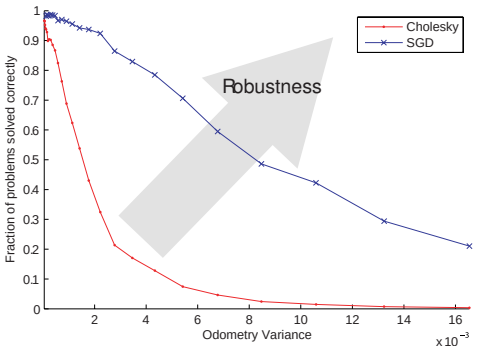

Figure 9: Convergence Robustness. On problems with significant noise and poor initial estimates, our algorithm finds the global minimum much more often than a Cholesky-Decomposition based method.

4 Conclusion

This thesis has presented new algorithms methods for both loop closing and map optimization. We show how relatively easy-to-generate loop-closure hypotheses can be filtered so that only correct loop-closure hypotheses remain. Our approach is based on the spectral properties of the pair-wise consistency matrix. A key feature of our approach is that the ratio of the first and second eigenvalues can be interpreted as a confidence metric, allowing perceptually ambiguous sensor data to be detected. In this thesis, we simply reject ambiguous hypothesis sets; a natural direction for future work would be to resolve ambiguous hypothesis sets by collecting additional data.

The second major contribution of this thesis is a new family of map optimization algorithms. These algorithms are based on stochastic gradient descent, which improves the state estimate by considering a single constraint at a time. Stochastic Gradient Descent has previously used in machine learning (particularly in the training of neural networks), but we showed how the method could be adapted to the map optimization problem. Our resulting algorithm rapidly explores the state space, giving it two key advantages: it escapes local minima that trap existing methods, and it can rapidly find solutions near the global minimum. This thesis describes both a batch and online version of the algorithm and evaluates the algorithms on a variety of real and synthetic data sets.