SLAM Pose-graph Robustification via Multi-scale Heat-Kernel Analysis

Sayantan Datta
sayantan.datta@research.iiit.ac.in

Siddharth Tourani
tourani.siddharth@gmail.com

Avinash Sharma
asharma@iiit.ac.in

K. Madhava Krishna
mkrishna@iiit.ac.in

***Abstract*— The Simultaneous Localization and Mapping problem (SLAM) in robotics is typically modeled as a dyadic graph of relative pose measurements taken by the robot. The graph nodes store the values representing the absolute pose of the robot at a given point of time. An edge connecting two nodes represents robot movement and it stores the measurements taken by the robot sensor while moving between two nodes. The objective of the SLAM problem is to find the optimal global measurements best satisfying the noisy relative measurements [12]. This problem of optimal estimation on a graph given relative measurements is a well-studied problem within the control community, for which several results and algorithms are known [3, 4]. SLAM is generally solved as a least squares problem. Robust kernels which are less sensitive to outliers are used to deal with noise and outlier measurements. However, robust kernels tend to be dependent on initialization and can fail as the number of outliers increase. Therefore, it's important to identify and prune the outlier (noisy) measurements represented by incorrect loop closure edges for an accurate pose estimate.**

In this paper we propose a multi-scale Heat-Kernel analysis based loop closure edge pruning algorithm for the SLAM graph. We show that compared to other pruning algorithms, our algorithm has a substantially higher precision and recall when compared and is able to handle a large amount of outlier measurements. We have corroborated results on several publicly available datasets and several types of noise. Our algorithm is not restricted to SLAM graphs only, but has a much wider applicability to other types of geometric graphs.

I. Introduction

The SLAM problem in robotics is a long studied problem with robust algorithms to solve it. It is typically formulated as a graph $G = (V, E)$ with the nodes representing robot poses and the edges capturing the measurements taken by the robot sensor while moving between the associated nodes. This is a special case of the another well-studied problem of assigning node values that best satisfy the edge constraints imposed by the graph. A problem that arises in other forms like sensor network localization, optimal sensor placement placement, etc. In case of SLAM, the node values $v \in V$ represent robot poses (usually $v \in \mathbb{SE}(2)$ or $\mathbb{SE}(3)$) relative to a fixed co-ordinate frame, canonically taken as the starting position of the robot. Each edge $e = (v_i, v_j) \in E$ connecting nodes v_i and v_j represents a relative measurement between the two nodes that are collected by a sensor mounted on the robot. There are mainly two types of sensors on the robot, *Odometry* sensors and *Exteroceptive* sensors. Odometry sensors construct edges connecting successive nodes and tend to be locally accurate. Exteroceptive sensors, like cameras, enable edge constructions between non-successive nodes by recognizing places that a robot has visited before and connecting them in the SLAM graph. These edges are called Loop Closure (LC) edges, which as the name suggests indicate that the robot has returned to a previous location and it's path has formed a loop.

All authors are associated with the Robotics Research Center, Kohli Center on Intelligent Systems, IIIT Hyderabad

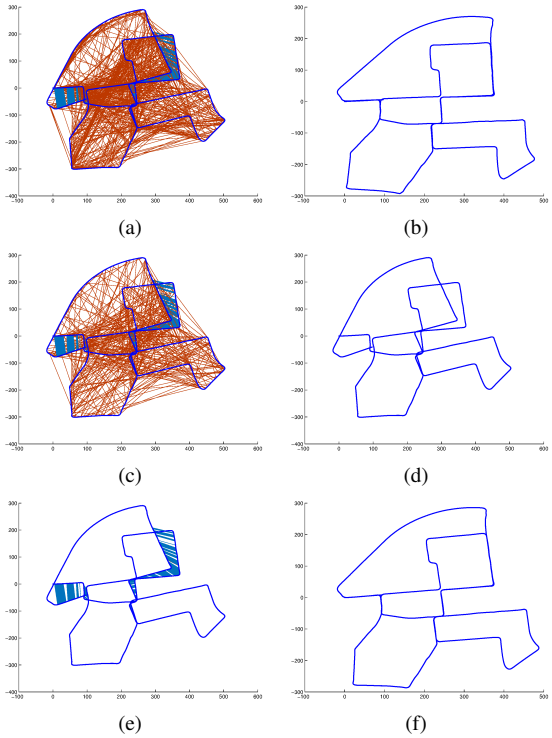


Fig. 1. Original and Optimized Pose-graphs. The thick dark blue line represents odometry edges. Fine Blue lines denote correct loop closures while the red lines denote an incorrect loop closure. **(a)** The pose-graph of Kitti00 along with added synthetically added outlier loop closures. **(b)** The desired ground truth map. **(c)** The optimized map from Switchable Constraints [23]. Note that a several outlier loop closure edges have been retained. **(d)** The output of RRR algorithm [18]. Although all the incorrect loop closures have been deleted here, several of the good loop closures are missing as well. **(e)** The robust pose-graph output from our proposed loop closure pruning method. Note that all the good loop closure edges have been successfully retained while being able to prune all the incorrect ones. **(f)** The final DCS [1] optimized map obtained from robust pose-graph.

A SLAM solution assigns pose values to all the graph nodes such that they optimally satisfy the edge constraints. A natural formulation for this would be as a non-linear least squares fitting of the node values to satisfy the edge measurements optimally. However, unlike odometry sensors, exteroceptive sensors tend to be more noisy and introduce significant number of outliers by establishing edges between incorrect nodes. This generally occurs due to a perceived similarity between distant nodes. The phenomenon is called perceptual aliasing, and occurs in map regions due to high self-similarity in architecture or greenery. To handle the out-

liers arising due to this phenomenon, researchers in SLAM have used robust penalty functions in the form of non-linear kernels instead of the standard l_2 penalty used in the least-squares problems, and were able to handle a fair amount of noise and reject outliers to a great extent.

However, like all non-linear optimizations, robust kernels are also very sensitive to initialization and can fail catastrophically with a sub-optimal initialization. An alternative approach to handle the outlier LC edges would be simply to prune them from the graph. This paper takes the later approach. We plan to build on Spectral Graph Theoretic (SGT) framework to identify and remove outlier edges from the SLAM graph. SGT based techniques have been successfully applied for topological analysis of graphs in multiple domain. However, we know of only few recent results where SGT techniques have been used to analyze the SLAM graph [16, 15].

In this paper we propose a multi-scale Heat-Kernel analysis based novel LC edge pruning algorithm for the SLAM graph. We show that it is capable of handling various types and significant amounts of noise in the graph structure and still manage to prune the outlier LC edges with a high precision and recall. We compare our algorithm against other pruning algorithms proposed in the literature [18, 23] and show superior results in terms of standard performance metrics.

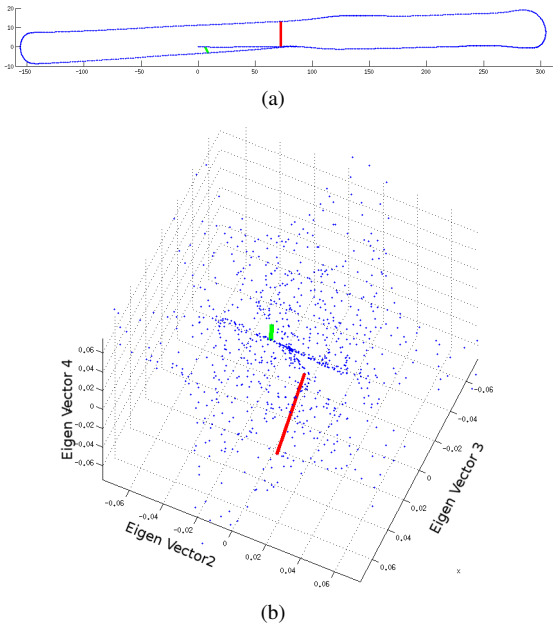
A pictorial representation of a result from our algorithm has been shown in figure 1 (e) and 1 (f). Considering a kitti00 pose graph which is heavily corrupted by outlier LC edges, our proposed algorithm can successfully prune the incorrect loop closures which are incoherent with the rest of the pose graph, where as other algorithms like Switchable Constraints(SC) delete scarcely any loop closures while Realizing, Reversing and Recovering (RRR) deletes several of the correct loop closures as well. On the other hand, in 1(f) we see that our algorithm visually appears closer to the ground truth.

II. Motivation & Contribution

The pose graph has an interesting structure making it amenable to such an analysis. One can easily adopt the weighted undirected graph representation for pose-graphs where the non-zero edge weighting scheme can model the uncertainity/confidence in each edge including both odometry and LC edges. In particular, we propose to adopt a convention where odometry edges can be set to relatively large values in comparison to the LC edges (set close to zero values) owing to the higher confidence in the odometric measurements as opposed to the LC constraints.

Speaking in terms of signal and noise, the pose graph usually has a very high signal-to-noise ratio along it's odometry edges, due to their relatively accurate measurements. On the other hand, LC edges typically have a lower signal-to-noise ratio, attributed to higher uncertainity from the exteroceptive sensor.

SGT techniques embeds graph nodes as points into the spectral space spanned by the eigen-vectors of the graph Laplacian matrix (corresponding to smaller eigen-values, see section IV). These eigen-vectors also have a dual PCA-like



(a)

(b)

Fig. 2. (a) The pose graph with the added correct loop closure (green) *(at the end of the odometry path)* and incorrect loop closure (red). (b) The corresponding loop closures drawn in the spectral space (see Section IV). The incorrect loop closure in (*red*) is still long enough while the correct loop closure in (*green*) has collapsed and is barely visible.

interpretation for graphs [20] i.e, these eigen-vectors captures the global structure of the underlying graph. Therefore, the correct LC edges, even with relatively low weights, will ideally collapse (project to points within close proximity in the embedding space) as they are consistent with respective to the graph topology dominated by the odometric edges. On the other hand the incorrect LC edges that will connect nodes that are projected as distant points in the embedding space.

Figure 2. shows a practical example where we plot a SLAM map (pose-graph) as an undirected graph with colored edges representing odometric (blue), correct LC (green) and incorrect LC (red). We can clearly see that in the (3d-)embedding space the correct LC edge collapse while the incorrect LC edge don't (Figure 2(b)). This clearly indicates that one can define a threshold on Euclidean distances computed across LC edges in the embedding space and prune incorrect LC edges. We propose to exploit this observation to filter out incorrect LC edges in SLAM graphs.

However, the real scenario is significantly more complex due to existence of multiple correct and incorrect LC edges present at different scales. Nevertheless, we can iteratively prune incorrect LC edges by considering different scale seperately using the multi-scale heat-kernel framework (see Section IV). We can start with a large scale heat-kernel embedding to prune incorrect LC edges, and in the subsequent iterations compute associated embeddings of the refined/filtered graphs at smaller scales to further prune incorrect LC edges.

Hence, we propose a novel LC pruning method where first the weighted undirected graph is induced for a given pose-graph. Subsequently, this graph is projected into the

spectral space to obtain the multi-scale Euclidean embeddings. Finally, the diffusion (heat-kernel) distances are used to identify incorrect LC edges which connect nodes that are projected far-away in the embedding space at different scales in iterative fashion. Furthermore, we also propose a novel LC detection method called "Lap-line" which helps initializing the LC edges by exploiting the local geometry of the pose-graph.

III. Related Work

SLAM as a problem has been formulated and solved in a number of ways. It has had been posed as a filtering problem [19], as an optimal control problem [8] etc. The stability, observability and controllability properties have been analysed for various sensors and motion models [24, 19, 10, 25, 13]. The most popular formulation however is in the form of a graph with relative measurements.

Different methods have been used to solve the SLAM problem including belief propagation [21], consensus algorithms [2], etc. However, the most frequently used and numerically stable method involves solving a non-linear least squares problem using quasi-newton methods. As with other all non-linear optimization methods, they tend to be sensitive to initialization. To deal with this short-coming graph pruning algorithms have been proposed, as a pre-processing to the graph optimization, to make the graph less sensitive to initialization by pruning the incorrect LCs.

Recently two pruning algorithms have been developed to handle incorrect loop closures. They are Realizing, Reversing and Recovering (RRR) algorithm [18] and Switchable Constraints (SC) [23]. RRR creates clusters of loop closure detections based on proximity of edges within the pose-graph. The algorithm starts with the odometry sub-graph and adds edge clusters that maintain consistency with the odometry data. The algorithm involves solving the pose graph and its subgraphs multiple times in order to detect inconsistent clusters and is thus computationally inefficient. It also suffers from very poor recall values at around 50%.

The switchable constraints (SC) algorithms, switches off constraints it deems to be inconsistent with the rest of the graph. A 0-1 weight variable is added into the non-linear objective for each constraint. The weight variable indicates whether a constraint is active or not. The integrality constraint is then relaxed and weights are optimized along with the rest of the objective. The optimized weight values indicate the relative accuracy/importance of an edge in the graph, with 1 being the highest. While this approach shows improvements over RRR, the additional weight variables optimized for give increase the time taken for the optimization dramatically.

To address this limitation Agarwal et al. proposed the Dynamic Covariance Scaling (DCS) [1] algorithm which computes a closed form approximation to each of the SC weight variables and thus avoids increasing the size of the optimization problem. This is an improvement over SC in terms of both speed and accuracy.

In comparison to both RRR and SC our algorithm has a much higher precision and recall when it comes to pruning false edges, and can handle a greater degree of noise. We demonstrate this on a number of SLAM datasets of varying graph topology.

IV. Background

The Spectral Graph Theory provides an implicit graph representation known as the Laplacian (spectral) embedding. The Laplacian embedding [5] is a popular spectral representation technique which maps each graph node to a K-dimensional space spanned by the first K non-null eigenvectors of the graph Laplacian matrix. This however is a fixed scale representation of the graph and a multi-scale representation can be easily derived by the related Heat-Kernel Framework. In this section, we introduce the mathematical constructs related to Laplacian and Heat-Kernel embedding of any undirected (non-negative) weighted graph, which subsequently is used in the proposed LC pruning method for pose-graphs.

A. Laplacian Embedding

Let $G = \{V, E, W\}$ be an undirected weighted graph where $V = \{v_1, \cdots, v_n\}$ be the set of graph nodes represented by (not-necessarily) some Euclidean coordinates, $E = \{e_1, \cdots, e_m\}$ be the set of undirected edges and W be the $n \times n$ square symmetric weighted adjacency matrix of the graph with each entry $W_{ij} \geq 0, \ \forall\, i, j \in \{1, \cdots, n\}$.

The un-normalized graph Laplacian matrix L can be derived as:

$$L = D - W, \tag{1}$$

where D is the diagonal degree matrix of the graph with each non-zero diagonal entry $d_i = \sum_{j=1}^{n} W_{ij}$. The L matrix is very sparse and positive semi-definite (PSD) and is seen as discrete counterpart of the Laplace-Beltrami Operator on continuous manifolds. The sparsity comes from the fact that the graph is considered to be only locally connected with each node having direct edges with nodes relatively smaller neighborhood. The rational behind is the fact that the notion of Laplacian operator (second order derivative) is essentially Euclidean and valid only for the tangent space of the manifold or in discrete case nearby nodes in the local neighborhood.

The Laplacian embedding of graph nodes is obtained using the eigen-spectrum of the graph Laplacian matrix. Let

$$L = U \Lambda U^T \tag{2}$$

be the eigen-decomposition of the L matrix where $U = [\vec{u}_1, \cdots, \vec{u}_n]$ be the eigen vectors and $\Lambda = Diag(\lambda_1, \cdots, \lambda_n)$ be the corresponding eigenvalues of the L matrix with property that $\{0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n\}$ and $\vec{u}_1 = \vec{1}$ for a connected graph [9]. The Euclidean embedding of the graphs nodes is defined as

$$X = \Lambda^{-1/2} U = [\vec{x}_1, \cdots, \vec{x}_i, \cdots, \vec{x}_n]. \tag{3}$$

Here, each vector $\vec{x}_i$ is the $n-$dimensional embedding of graph node $v_i \in V$. Interestingly, we can only select k ($\ll n$) non-null eigen-vectors corresponding to the k smallest, non-zero eigen-values and embed the graph nodes as point in the lower dimensional Euclidean space spanned by these selected k orthogonal eigen-vectors and values such that $X \in \mathbb{R}^{k \times n}$.

This is also known as the commute-time embedding where the Euclidean distance between the projection of two graph nodes in the embeddings space has an interpretation of commute-time distance covered by a random walker on the original graph [20]. Thus, it approximates the average connectivity over the original graph.

B. Heat-Kernel Embedding

Heat diffusion is a fundamental concept in physics. The heat diffusion equation is a partial differential equation which describes the distribution of heat (or the variation in temperature) in a given location and over time. Heat diffusion can be generalized to non-Euclidean spaces such as manifolds and graphs. Heat diffusion on graphs is exactly the parallel of diffusion on closed Riemannian manifolds where the heat-kernel matrix is defined as [7]:

$$H(t) = e^{-Lt}. \tag{4}$$

Here, $t > 0$ is a time/scale parameter and L is the unnormalized graph Laplacian matrix. Each entry of $H(t)$ matrix is a Mercer kernel [22] that has a very simple physical interpretation, as follows. We consider real-valued functions $\vec{f}$ over V, $\vec{f}$: $V \to \mathbb{R}$ and we note that $\vec{f} = (f_1, \cdots, f_n)^T$ is simply a vector indexed by the nodes of G. The vector $\vec{f}(t) = H(t)\vec{f}$ is a solution to the heat-diffusion equation $(\frac{\partial}{\partial t} + L)\vec{f}(t) = 0$.

Hence, $\vec{f}$ corresponds to some initial heat distribution over the nodes of graph G and $\vec{f}(t)$ is the heat distribution at time/scale t starting from $\vec{f}(0) = \vec{f}$. Notice that starting with a point heat distribution at node v_j, $\vec{f}_j(0) = [0, \cdots, 1, \cdots, 0]^T$, the heat distribution at time t is given by the jth column of the heat matrix which is denoted by $H(:, j; t)$ as

$$\vec{f}_j(t) = H(t)\vec{f}_j(0) = H(:, j; t) \tag{5}$$

From Eq. 5 we can obtain a straightforward interpretation of the entries of the heat matrix, namely each entry $h(i, j; t)$ of H(t) corresponds to the amount of heat available at node v_i at time t, starting with a point heat distribution at node v_j. The symmetric function h: $V \times V \to \mathbb{R}$ is the heat kernel of a graph G. Each diagonal term $h(i, i; t)$ of the heat-kernel matrix has an interesting interpretation as well. It corresponds to the amount of heat remaining at node v_i at time t. To conclude, the heat-kernel matrix encapsulates important intrinsic information about how heat travels from one part of the graph to another part or in a way heat diffusion at different scales can be used to capture scale-dependent topological characterization of graphs.

Interestingly, multiple scale dependent characterizations of graphs can be easily obtained by computing the heat-kernel matrix for different value of t using the pre-computed eigen-vectors and eigen-values of associated graph Laplacian matrix L as:

$$H(t) = \sum_{i=1}^{n} e^{-\lambda t}\vec{u}_i\vec{u}_i^T. \tag{6}$$

Let $\vec{u}_i = [u_{i1}, \cdots, u_{il}]^T$ be the i-th eigenvector of L matrix then each entry $h(i, j; t)$ of $H(t)$ matrix is a heat kernel:

$$h(i, j; t) = \sum_{l=1}^{n} e^{-\lambda_l t}u_{il}u_{jl} =< \vec{y}_i, \vec{y}_j >, \tag{7}$$

where,

$$\vec{y}_i = [e^{-\lambda_1 t/2}u_{i1}, \cdots, e^{-\lambda_n t/2}u_{in}]^T. \tag{8}$$

Hence, $\vec{y}_i$ is an element of the feature space, or heat-kernel embedding of the graph in $\mathbb{R}^n$. In practice one can use a reduced dimension $k \ll n$. The heat-kernel can be used to define distances in the Euclidean embedding (feature space), namely, the diffusion distance or heat-distance defined as:

$$||\vec{y}_i - \vec{y}_j||^2 = h(i, i; t) + h(j, j; t) - 2h(i, j; t) \tag{9}$$

V. Proposed Approach

We propose a novel multi-scale LC pruning method which can significantly improve the performance of existing SLAM optimization techniques by pruning the noisy constraints (incorrect LC edges) from the pose-graph. Figure 3. depicts the overview of proposed method. Our novelty lies in the two key modules: pose-graph induction and multi-scale LC Pruning. Here we discuss each of the modules in detail.

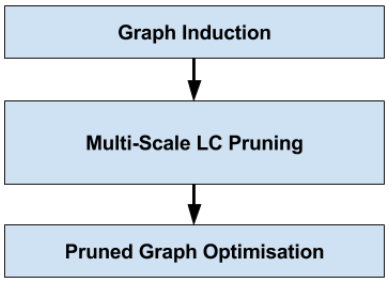


Fig. 3. Overview of Algorithm

A. Pose-graph Induction

The task of graph induction involves deriving a new affinity matrix (W) from the pose-graph, which has same number of edges and nodes. This will allow us to capture the relative strength of odometry and LC edges, which subsequently will be exploited in the spectral analysis (IV). An efficient and prudent weighing mechanism of edges is required for the graph laplacian and multi-scale heat-kernel to work well.

Our strategy is that the odometry edges of the graph are given a significantly higher weight, so that the overall structure of the graph is maintained. On the other hand the weight should signify confidence whether a LC edge might be correct or incorrect. The higher weights should ideally assigned to a correct LC edges as opposed to the lower weights assigned to incorrect LC edges while these weights are normalized between 0 and 1.

However, it is very challenging to initialize these weights for LC edges before performing the SLAM optimization. We have proposed two specific ways for weight initialization of LC edges in W.

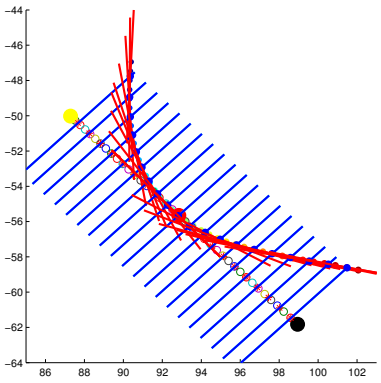


Fig. 4. Shows the trajectory descriptor of *Lapline* working along an odometry edge of a graph. The tangential line (composed of scatter points), is the line l, from which each of the perpendicular bisectors intersect the trajectory path at regular intervals. The point of intersections has been marked with a blue circle. The line descriptor is the vector of the length of blue line segments from the tangent of the trajectory at v_1 to its intersection with trajectory.

1) Lap-line: For pose-graphs which lack a perfect initialization in form of confidence matrices of the LC edges, creating a robust set of edge weights becomes more challenging. In order to handle such graphs efficiently, we introduce a innovative procedure called *Lap-line*.

In case of a correct LC edge in a pose-graph, it is considered that the robot is currently revisiting an already pre-traversed path. Ideally, as the robot is on the same path, it should be navigating with similar trajectory as it did take before. Therefore, we construct an underlying belief that both trajectories would be locally similar.

Our Lap-line algorithm starts with a local trajectory descriptor, which can also be expressed in a more generic term as a local curve descriptor. Consider a loop closing edge in a graph G is formed by the two nodes (poses) v_1 and v_2. The weight of this edge should be directly related of the similarity of their local trajectory descriptor $\mathcal{T}$. The steps for which are enumerated below:

1) We determine the length of the local trajectory to be matched. Considering v_1^f and v_1^b be the delimiter nodes which determines the length of the local trajectory. We define a local trajectory descriptor for v_1 in which, we keep dilating the difference between v_1^f and v_1^b on either direction of v_1. We do it till we reach a considerable length trajectories, $30 \times \mathcal{L}$ (where $\mathcal{L}$ is the average odometry edge length), or the path takes more than a $\pi/2$ turn within itself. The black and the yellow points in Fig. 4. are v_1^f and v_1^b.
2) We draw a straight line l through v_1, with a slope of a line defined as $((v_1^f)_y - (v_1^b)_y)/((v_1^f)_x - (v_1^b)_x)$ where x and y refer to the respective coordinate component.
3) With the line l as reference, we calculate perpendicular distances between itself and the trajectory at $\mathcal{L}$ intervals (on l). This is depicted by the blue lines in Fig. 4. As the trajectory is not a continuous line but formed of discrete poses, we relax the intersection coordinate to a piece-wise linear line segment between the poses. These line segments are depicted as the red lines in Fig. 4. and the blue filled dots (almost occluded by the red lines) are the point of intersection.
4) We record the length of each distances from l to the trajectory as $\vec{ld}_1$. The lengths are negative or positive depending on their relative position (left or right) from the orientation of robot travel. As the intervals, at which the elements of $\vec{ld}_1$ are polled, are equidistant and calculated on l instead of the pose-graph trajectory. This $\vec{ld}_1$ descriptor is invariant of a locally dense or a sparse map.

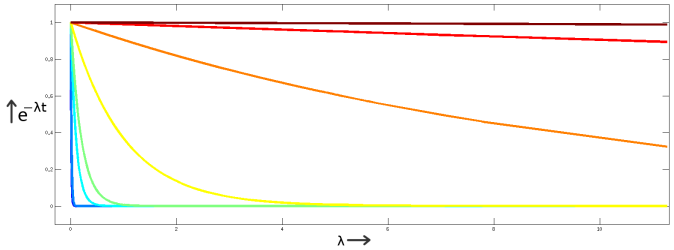


Fig. 5. Plot of $e^{-\lambda t}$(exponent of the eigen-values) versus λ(eigen-values) showing the exponential cover of the scale parameter. The scales used in this plot, $\vec{t}$ are *[100,70,10,5,1,0.1,0.01,0.001]*, highest corresponding to the dark blue (on the left edge of the graph) and the lowest corresponding to the brown (on the top of graph)

Now, as $\vec{ld}$ is calculated for both v_1 and v_2 of the loop closure edge, an equal length of $\vec{ld}_1$ and $\vec{ld}_2$ are compared. Smaller the disparity between the two descriptors, the better the loop closure it is.

2) χ^2 Initialization: χ^2 initialization is done for pose graphs which has a reliable information matrix for it's edges. It is done by taking the odometry map, and loop closure edge at a time, the χ^2 error for that edge is recorded in a vector $\vec{\chi d}$, and this is continued over all the loop closure edges. The elements in $\vec{\chi d}$ having the lowest values are given a higher weight in pose-graph induction, and vice-versa.

B. Multi-scale LC Pruning

Once, the weighted graph matrix is induced with appropriate initialization, we compute the eigen-values and eigen-vectors of the graph Laplacian matrix (see Eq. 2.) derived from weighted affinity matrix formed in the graph induction step (see Eq. 1). Subsequently, we compute the heat-kernel matrix (Eq. 6) for a fixed set of 'p' time/scale parameters represented in a vector as $\vec{t} = \{t_1, \cdots, t_p\}$ where $t_1 > t_2 > \cdots > t_p$. Elements of $\vec{t}$ varies from large to small values indicating global to local heat-diffusion on pose-graph.

Figure 5. is an example plot of $e^{-\lambda t}$ versus λ indexes on kitti00 dataset. It is clearly visible that a large value of scale parameter, the exponential cover drops very fast, pointing that only initial eigen-vectors with smallest eigenvalue (structural components) participate in the large scale diffusion, thereby making it a global phenomenon. On the other hand, the curve is relatively flat for small values of scale parameter suggesting the contribution of large number of eigen-vectors (high frequency components) thereby making it highly local.

The set of scale dependent heat-kernel embeddings (or heat-kernel matrices) are used in an iterative fashion, such that initially we focus on incorrect LC's that connect distant points in the embedding space and remove them by thresholding the heat-kernel distance (Eq. 9) between pair of nodes participating in every LC edge (Eq. 8) with a scale specific threshold parameter. In subsequent iterations, we recompute the the graph Laplacian with remaining edges in the pose-graph (both odometry and LC edges) and repeat the same process using other (smaller) values of scale parameter stored in vector $\vec{t}$ for removing incorrect (noisy) LC edges that have smaller scale. Algorithm 1. outlines the proposed iterative LC pruning method in details.

Algorithm 1: Multi-scale LC Pruning on Pose-graphs

Input: Pose-graph $G = \{V, E, W\}$ and scale vector $\vec{t}$.
Output: Robust Pose-graph $\widehat{G} = \{V, \widehat{E}, \widehat{W}\}$.

```
Initialize Ĝ = G
for i ← 1 to p do
    Initialize set of pruned LC index PLC=NULL
    Define L̂ using Ĝ as in Eq. 1
    Compute U and Λ from SVD of L̂ (see Eq. 2.)
    Compute Heat-Kernel matrix H(t_i) using Eq. 6.
    Compute threshold parameter θ_i using Eq. 11.
    for j ← 1 to q do
        if ||y_start(j) − y_end(j)||^2 > θ_i then
            Add j to set PLC
    Update Ĝ by removing set of LC edges stored in
    PLC from Ê and set corresponding affinity matrix
    entries to zero in Ŵ.
return Ĝ
```

C. Pruned Pose-graph Optimization

Once the LC edge pruning is performed by Algorithm 1, the respective edges are also removed from the original pose-graph. The final pose-graph optimization is done by solving a non-linear least squares problem that assigns to the node, values that best explain the measurements by minimizing the overall fitting error

$$\sum_{e_{i,j} \in E} \rho(||(T_{v_j} T_{v_i}^{-1})^{-1} T_{e_{i,j}}||) \tag{10}$$

In Eq. 10. ρ is the robust kernel being used, T_{v_j} is the transformation associated with vertex v_j and $T_{e_{i,j}}$ is the transformation associated with the edge connecting the vertices v_i and v_j. i and j are indices of the vertices. To carry out the optimization in our experiments we use the g2o library [17].

D. Choice of Parameters

We have two parameters for this algorithm, which is the threshold parameter θ_i and the time/scale parameter $\vec{t}$. The minimum element in $\vec{t}$ is always 0 but the largest one depends on the dataset (value at which $e^{\lambda t}$ saturates). The intermediate values in $\vec{t}$ are linearly sampled at the logarithmic scale.

The threshold parameter θ_i changes at every iteration and in different values of the $\vec{t}$. It is usually a combination of the mean μ and standard deviation σ of the euclidean distance of the loop closing edges edges in the spectral embedding, and is defined as follows:

$$\theta_i = \mu + (konst \times \sigma), konst \in \mathbb{R} \tag{11}$$

$konst$ varies from anywhere from 0 to 4, depending on the scale of heat diffusion.

VI. Experiments and Result

A. Experimental Setup

We have tested our algorithm on Kitti datasets 00,06 [11] and Bicocca [18]. Along with it, the pose-graphs of Bicocca (B25b) were also used. All the datasets have the ground truth GPS tracks. The Kitti and the B25b datasets also has the output of Bag-of-Words (BoW) loop detection algorithm.

The dataset had the ground truth loop closures sparsified to increase speed of the pose-graph optimization algorithms. In particular, we have observed that Kitti dataset has very sparse set of correct LC edges. As our proposed method exploits the graph topology, it's performance will be better if we have densified correct loop closures. Therefore, we have obtained densified correct loop closures by replicating the original set of sparse LC edges in local neighborhood, only for Kitti dataset.

For testing, we have considered a low α of the BoW system. This would result in our pose graph containing higher percentage of incorrect loop closures and hence a better test. Furthermore, synthetic loop closures were also added in the following 5 types of noises — local, local-grouped, random, random-grouped and all-random.

For each dataset, we have considered noise levels ranging from 10% to 70% added incorrect LCs of the pose count in the graph, in increments of 10% each. Every level of noise is the mean of 3 individual initializations of noise, for each of 5 types of noise stated above. Hence, a dataset of 105 files are created for every dataset.

B. Evaluation

We have evaluated the performance of our algorithm on two different grounds; (1) against robust kernels such as DCS and Cauchy, and (2) against robust pose graph optimizers which prune incorrect loop closures such as RRR and SC.

To compare the performance of our proposed algorithm with the m-estimators, we have used the error parameter Absolute Translation Error (ATE) as defined in the Rawseeds toolkit [6]. To compare the performance of the exactness of our pruning of the loop closing edges, we have used precision and recall values.

To calculate ATE of the optimized pose graph against ground truth graph, we align the graphs to be compared. We do so, by calculating the $\mathbb{SE}2$ or $\mathbb{SE}3$ transformation required to correlate the first pose of the graph to the first pose of ground truth pose graph. This transformation is applied for every pose throughout the optimized pose-graph.

Dataset	# of outliers	Outlier Type	Precision			Recall			F-Measure			ATE		
			SC	RRR	Ours	SC	RRR	Ours	SC	RRR	Ours	SC	RRR	Ours
Kitti00	402	Local	1.00	0.66	0.97	0.45	1.00	0.85	0.62	0.79	**0.90**	35.16	35.30	**4.12**
		Local-Grouped	1.00	0.68	0.95	0.44	1.00	0.65	0.61	**0.81**	0.77	35.16	35.30	**11.98**
		Random	1.00	0.52	0.99	0.42	1.00	0.99	0.60	0.69	**0.99**	35.16	35.30	**3.72**
		Random-Grouped	1.00	0.67	0.96	0.43	1.00	0.73	0.60	0.80	**0.81**	35.16	35.30	**3.39**
		All Random	1.00	0.65	0.96	0.42	1.00	0.73	0.60	0.79	**0.83**	35.16	35.30	**3.56**
Kitti06	88	Local	nan^*	0.22	0.71	0.00	1.00	0.99	0.00	0.36	**0.83**	**4.57**	5.91	4.61
		Local-Grouped	nan^*	0.22	0.57	0.00	1.00	1.00	0.00	0.36	**0.73**	4.57	5.91	**4.47**
		Random	nan^*	0.27	0.87	0.00	1.00	1.00	0.00	0.43	**0.93**	**4.50**	5.91	4.55
		Random-Grouped	nan^*	0.28	0.70	0.00	1.00	1.00	0.00	0.44	**0.83**	**4.56**	5.91	4.59
		All Random	nan^*	0.28	0.68	0.00	1.00	1.00	0.00	0.44	**0.81**	4.56	5.91	**4.35**

TABLE I
PERFORMANCE ON KITTI DATASETS.

* in the case of SC, switchable constraints (SC) did not delete any edges, resulting in a nan average over experiments due to a division by zero.

ATE is calculated by averaging over the translation disparity between ground truth poses and that of optimized pose-graph.

For precision and recall, the true positives (TP) are incorrect loop closures identified as such and eventually pruned, false positives (FP) are correct loop closures identified as a bad loop closure and hence pruned, false negatives (FN) are incorrect loop closures which had not been pruned, and true negatives (TN) are correct loop closures that remained in the output file.

We have used the publicly available version of RRR algorithm. For Switchable Constraints, we have used the default parameters. For DCS and Cauchy, we have checked the kernel width 1 to 20, and used the parameter which has provided least ATE error.

C. Results

To evaluate the performance of our proposed algorithm, we have compared our precision and recall performance while using χ^2 initialization, against that of RRR and SC. We believe this to be a fair comparison as RRR's inter cluster and intra cluster loop closure consistency algorithm is based on a similar note.

The Table I. shows the accuracy of our algorithm compared against that of SC and RRR. We find that over Kitti00 and Kitti06, our algorithm performs very well compared to that of the competitors. For Kitti06, the precision values for SC are nan, as it was unable to delete any of the loop closing edges. This resulted in the true positives being zero, as none of the incorrect loop closures were deleted, and the false positives also being zero, as none of the good loop closures were also deleted. When it comes to the ATE, our algorithm in most cases achieves a lower ATE than the other two approaches. It is substantially lower in the case of the Kitti00 dataset. Even in the cases when the ATE of our algorithm is not the lowest, it is very close to the best result. Overall, showing that our algorithm does exceptionally well in both performance metrics when compared to the state-of-the-art.

To evaluate the performance of our proposed LC pruning algorithm with Lap-line initialization against that of the m-estimators [14], see Fig. 6. We notice that there is a overall improvement of ATE, when our algorithm is used with DCS.

# of outliers	Outlier-Type	DCS	Ours + DCS	Cauchy
44	Local	9.90	**2.47**	5.73
	Local-Grouped	33.68	**3.25**	5.72
	Random	4.12	**3.49**	5.70
	Random-Grouped	5.69	**3.07**	6.98
	All Random	4.12	**3.53**	5.78
223	Local	36.82	**2.92**	5.70
	Local-Grouped	4.16	**3.67**	5.77
	Random	38.44	**3.91**	8.92
	Random-Grouped	10.87	**3.66**	10.97
	All Random	4.17	**2.23**	5.77
312	Local	7.64	**2.94**	5.67
	Local-Grouped	53.26	14.17	**9.78**
	Random	4.63	**4.42**	6.04
	Random-Grouped	5.14	**4.32**	24.76
	All Random	6.15	**3.73**	7.35

TABLE II
ATE RESULTS ON BICOCCA

This improvement is prevalent over multiple levels of noise and different types of noise as well. The quantitative results are shown in tabular form in Table II.

VII. CONCLUSION

We have presented a graph pruning algorithm specifically designed to handle erroneous and noisy loop closures by performing a multi-scale heat-kernel analysis on the SLAM graph. We compared our algorithm against other state-of-the-art graph pruning algorithms and showed that our algorithm performs significantly better for various types of high level noise added to the graph.

VIII. ACKNOWLEDGMENT

This work was supported by the Centre for Artificial Intelligence and Robotics, Defence Research and Development Organization, Government of India.

REFERENCES

[1] Pratik Agarwal et al. "Robust Map Optimization using Dynamic Covariance Scaling". In: *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE. 2013.

[2] Rosario Aragues, Jorge Cortes, and C Sagues. "Distributed consensus algorithms for merging feature-based maps with limited communication". In: *Robotics and Autonomous Systems* 59.3 (2011), pp. 163–180.

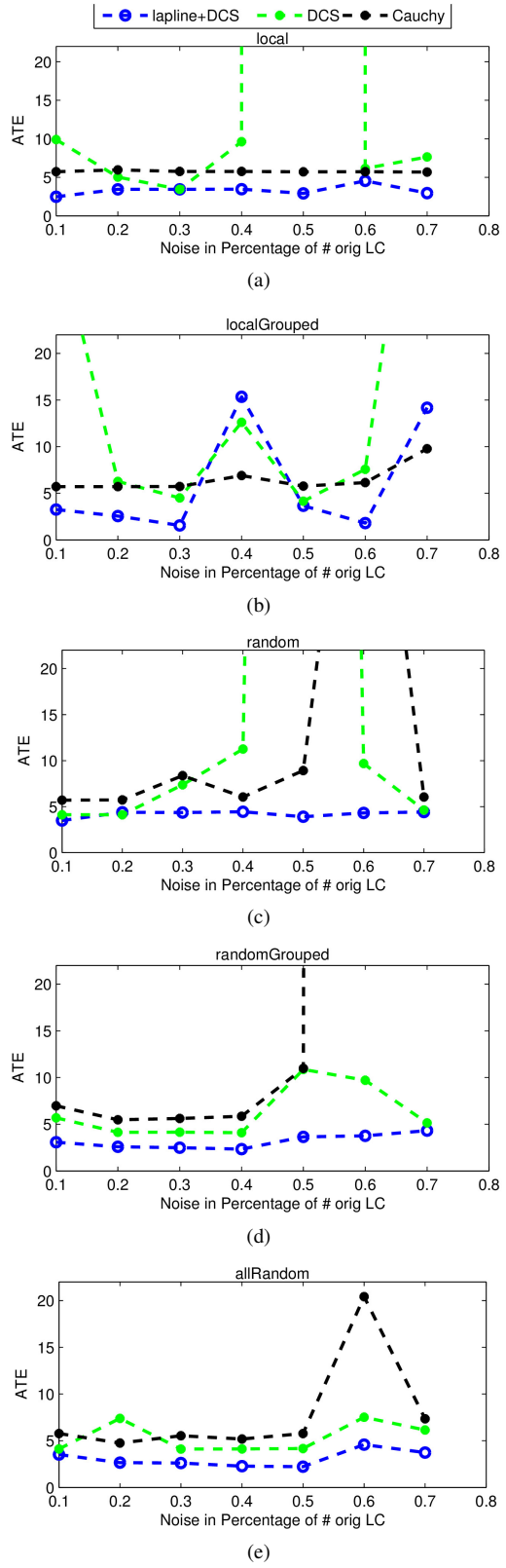


Fig. 6. Absolute Translation Error for the Bicocca dataset. Our proposed method (multi-scale LC pruning algorithm initialized with lapline and optimised by DCS) (*Blue*) vs DCS (*Green*) and Cauchy (*Black*). The X-axis denotes the level of noise, which increases from 0.1 to 0.7 times of the initial number of loop closures. The Y-axis denotes the Absolute Translation Error (ATE). Fig. **(a)** denotes the performance for local loop closures. Fig. **(b)** for locally-grouped Loop Closures. Figure **(c)** for random loop closures. Fig. **(d)** for randomly-grouped loop closures and Fig. **(a)** for all-random loop closures.

[3] Prabir Barooah and Joao P Hespanha. "Estimation from relative measurements: Electrical analogy and large graphs". In: *IEEE Transactions on Signal Processing* 56.6 (2008), pp. 2181–2193.

[4] Prabir Barooah and Joao P Hespanha. "Graph effective resistance and distributed control: Spectral properties and applications". In: *IEEE 45th IEEE conference on Decision and control*. IEEE. 2006, pp. 3479–3485.

[5] Mikhail Belkin and Partha Niyogi. "Laplacian eigenmaps for dimensionality reduction and data representation". In: *Neural computation* 15.6 (2003), pp. 1373–1396.

[6] Simone Ceriani et al. "Rawseeds ground truth collection systems for indoor self-localization and mapping". In: *Autonomous Robots* 27.4 (2009), pp. 353–371.

[7] Ronald R. Coifman and Stphane Lafon. "Diffusion maps". In: *Applied and Computational Harmonic Analysis* 21.1 (2006), pp. 5 –30. ISSN: 1063-5203.

[8] M. Egerstedt and P. Jensfelt. "A Control Theoretic Formulation of the Generalized SLAM Problem in Robotics". In: *American Control Conference*. 2008.

[9] Miroslav Fiedler. "Algebraic connectivity of graphs". In: *Czechoslovak mathematical journal* 23.2 (1973), pp. 298–305.

[10] Nikolaos M Freris and Anastasios Zouzias. "Fast distributed smoothing of relative measurements". In: *IEEE 51st Annual Conference on Decision and Control*. IEEE. 2012, pp. 1411–1416.

[11] Andreas Geiger et al. "Vision meets robotics: The KITTI dataset". In: *The International Journal of Robotics Research (IJRR)* (2013).

[12] Giorgio Grisetti et al. "A Tutorial on Graph-Based SLAM". In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43.

[13] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. "Observability-based rules for designing consistent EKF SLAM estimators". In: *The International Journal of Robotics Research* 29.5 (2010), pp. 502–528.

[14] Peter J Huber. *Robust statistics*. Springer, 2011.

[15] Kasra Khosoussi, Shoudong Huang, and Gamini Dissanayake. "Good, Bad and Ugly Graphs for SLAM". In: *RSS Workshop on The Problem of Mobile Sensors*. 2015.

[16] Kasra Khosoussi, Shoudong Huang, and Gamini Dissanayake. "Novel insights into the impact of graph structure on SLAM". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2014, pp. 2707–2714.

[17] Rainer Kümmerle et al. "g2o: A general framework for graph optimization". In: *IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3607–3613.

[18] Yasir Latif, Cesar Cadena, and Jose Neira. "Robust Loop Closing over Time for Pose-Graph SLAM". In: *International Journal of Robotics Research (IJRR)* (2013).

[19] Faraz M Mirzaei and Stergios I Roumeliotis. "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation". In: *Robotics, IEEE Transactions on* 24.5 (2008), pp. 1143–1156.

[20] H. Qiu and E. R. Hancock. "Clustering and Embedding Using Commute Times". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.11 (2007), pp. 1873–1890. ISSN: 0162-8828.

[21] Ananth Ranganathan, Michael Kaess, and Frank Dellaert. "Loopy sam". In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*. 2007, pp. 6–12.

[22] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. New York, NY, USA: Cambridge University Press, 2004. ISBN: 0521813972.

[23] Niko Sünderhauf and Peter Protzel. "Switchable constraints for robust pose graph slam". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2012, pp. 1879–1884.

[24] Zhan Wang and Gamini Dissanayake. "Observability analysis of SLAM using fisher information matrix". In: *10th International Conference on Control, Automation, Robotics and Vision*. IEEE. 2008, pp. 1242–1247.

[25] XS Zhou et al. "Interrobot relative pose controllability and observability". In: *University of Minnesota, Dept. of Comp. Sci. & Eng., MARS Lab, Tech. Rep* 1 (2007).