Detecting the correct graph structure in pose graph SLAM

Yasir Latif, César Cadena, and José Neira

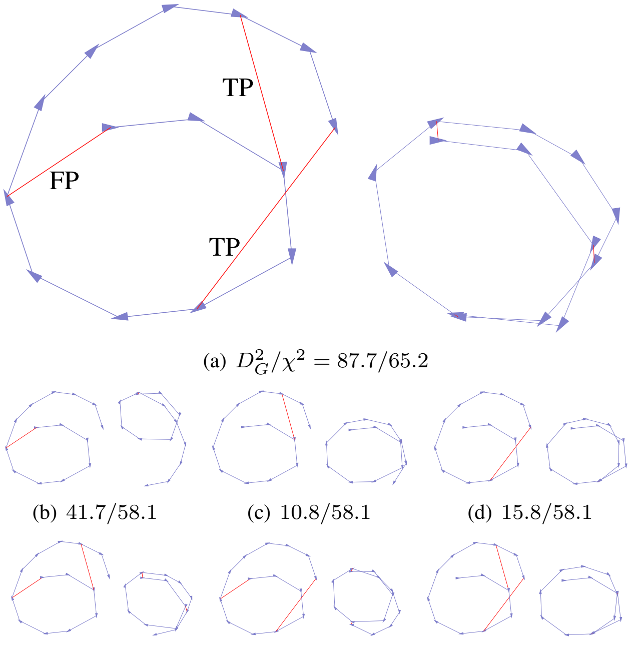
***Abstract*— While graph-based representations allow an efficient solution to the SLAM problem posing it as a non-linear least squares optimization, they require additional methods to detect and eliminate outliers. It is necessary to obtain the correct structure of the graph representing the SLAM problem which is topologically correct and will lead to a metric correct solution once optimized. In the graph-SLAM context, the edges represent constraints relating two poses whereas the vertices represent the robot poses. While all the edges between consecutive poses actually exist (odometry), the same may not be true for edges coming from a place recognition system. Place recognition algorithms always have some degree of failure in the presence of perceptual aliasing in real environments, creating edges between otherwise unrelated poses. We argue that rather than mitigating the effect of incorrect loop closures, these edges must be detected and removed as they represent non-existent topological connections in the graph. In this paper we describe a method that is able to detect and remove such false edges, leading to a solution of the SLAM problem based on the resulting topologically correct graph. Our method is robust both to outliers in place recognition as well as errors in odometry systems. We focus our experiments on real world and synthetic datasets and provide comparisons against other robust SLAM methods.**

I. Introduction

Recently, the graph based formulation to solve the SLAM problem has become a common choice. Even though this representation was proposed a long time ago [1], it has made a comeback thanks to the advances which allow efficient solution for the non-linear optimization problem by taking advantages of the sparse nature of the SLAM problem, for instance iSAM [2], HOG-Man [3], and g2o [4].

A toy example of a pose-graph is shown in Fig. 1. The nodes in the graph represent unknown robot poses and the edges represent constraints between these poses. The edges are obtained from an odometry system (sequential constraints) and a place recognition system (loop closure constraints), blue and red lines in Fig 1, respectively. Once the graph is built, an optimization process finds the configuration of poses that best explains all the constraints.

The majority of pose graph optimizers consider constraints with some degree of noise, typically Gaussian, but also that all the constraints really exist. For these kind of problems, different improvements have been developed in presence of non-Gaussian noise, e.g. using robust cost functions (Huber function) [5], using robust optimization methods [6], or more recently handling directly non-Gaussian distributions [7]. These approaches are excellent options to handle non modeled errors on the constraints, like slippage errors in odometry or inaccurate transformations computed from different views in a loop closure.



(a) $D_G^2/\chi^2 = 87.7/65.2$

(b) 41.7/58.1 (c) 10.8/58.1 (d) 15.8/58.1

(e) 82.7/61.7 (f) 59.2/61.7 (g) 16.4/61.7

Fig. 1. Toy example. Graph with the poses as nodes and constraints from some odometry source (blue) and a place recognition system (red) with one false positive (FP) and 2 true positives (TP). While we can obtain an acceptable value for the cost function (D_G^2) from the optimizer for the graphs in (b-d, f and g), only the graphs on (c, d and g) are topological correct (do not contain false edges). The simplest case of only selecting the odometry edges, 0 loop closure edges, obtains a cost function value of 0. The goal is find to a correct configuration with a consistent cost function ($D_G^2 < \chi^2$) and as many loop closure edges as possible.

Unfortunately, in the presence of false positives in the place recognition system (constraints that really do not exist) these approaches can only reduce, and not eliminate, the effect of them in the final estimation, and they fail in the presence of multiple or persistent false edges. Note that there exists a fundamental difference between edges generated by the odometry and those by the place recognition. Odometry constraints by nature are topologically correct (even if they might be metrically inaccurate) while the same does not hold for loop closure constraints. The problem is then to determine which loop closure constraints are topologically consistent with the graph being constructed for the SLAM problem.

Yasir Latif, and José Neira are with the Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Zaragoza 50018, Spain. {ylatif, jneira}@unizar.es.

César Cadena is with the Computer Science Department, Volgenau School of Engineering at George Mason University, Fairfax, VA 20030, US. ccadenal@gmu.edu.

This research has been funded by the Dirección General de Investigación of Spain under projects DPI2009-13710 and DPI2009-07130 and by DGA-FSE(group T04), and by the US Army Research Office Grant W911NF-1110476.

Different works have been developed to avoid or at least minimize the effect of outlier constraints. One popular way is to robustify the front-end place recognition system. However, no system can guarantee 100% accuracy. In some environments perceptual aliasing is almost impossible to avoid, even for human beings. At the moment it seems that all these improvements in front-ends are insufficient to guarantee complete robustness.

Olson [8] proposed a hypothesis verification method for loop closure constraints using graph partitioning based on spectral clustering. This method fails in the presence of odometry drift for large loops. Another approach is to delay decision making and maintain multiple topologies of the map with an associated belief for each one [9]. Their approach uses a Rao-Blackwellized particle filter. However, it was not explicitly shown how the system is affected by and recovers from incorrect loop closures. Their method is also unique in the sense that it uses the estimation process itself to reason about possible loop closures.

In recent literature, two methods have been proposed to deal with the problem of robustifying the back-end to deal with possible false positive loop closures: [10] and [11]. The robust SLAM back end using "switchable constraints" [10] penalizes those loop closure edges during graph optimization that deviate from the constraints they suggest between two nodes. Their method suggests a continuous function governing the state of "switch factors" which may not be appropriate in many cases, for example in persistent false positives on traversal paths.

The "max-mixture" [11] attaches a null hypothesis to each possible loop closure and lets the optimizer select the more probable one. The null hypothesis represents a uniform distribution over the whole space and is modeled by a Gaussian distribution with the same mean as the loop closure hypothesis and a very large variance. We will return to both these methods in the experimental section showing comparisons on real data.

In order to deal with the problem of incorrect place recognition, we should be able to: **(a)** distinguish between correct and incorrect loop closures being introduced by the front-end place recognition algorithm, **(b)** discard incorrect loop closures from the estimation process and **(c)** recover the correct map estimate.

We use the novel method: *Realizing, Reversing, and Recovering* (*RRR*) for loop closure verification i.e: given one or more sets of sequential constraints provided by odometry and a set of potential loop closing constraints provided by a place recognition system, the algorithm is able to differentiate between the correct and incorrect loop closures. The underlying idea is that of consensus: correct loop closures tend to agree among themselves and with the sequential constraints, while incorrect ones tend not to. This, along with convergence properties of optimization techniques, provides a robust method for rejecting false loop closures.

RRR makes the best possible decisions over the acceptance of loop closing given all the available information provided by the place recognition and the odometry systems, either incrementally or in batch. The output is an optimized pose graph with the correct topological structure, with no auxiliary constraints and without changing the formulation of the problem.

In the next section we show the SLAM problem in the graph formulation. In section III we detail the detection of a correct graph structure using the RRR algorithm. Experiments are detailed in section IV along with comparisons and evaluations with competing state of the art methods. Finally, in section V we present further discussion and conclusions about this work.

II. The Pose Graph Formulation

In the graph based formulation for SLAM, the so-called "graph-SLAM", robot poses as modeled as state variables in the graph's nodes and constraints as factors on the graph's edges. The factors represent a distance to minimize between the poses and the observations given by the sensors. Considering the Gaussian assumption to model the sensor's noise, we use the covariance or the information matrix. Let $\mathbf{x} = (x_1 \ldots x_n)^T$ be a vector of parameters that describe the configuration of the nodes. Let ω_{ij} and Ω_{ij} be the mean and the information matrix of the observation of node j from node i. Given the state $\mathbf{x}$, let the function $f_{ij}(\mathbf{x})$ be a function that calculates the perfect observation according to the current state. The residual r_{ij} can then be calculated as:

$$r_{ij}(\mathbf{x}) = \omega_{ij} - f_{ij}(\mathbf{x}) \tag{1}$$

Constraints can either be introduced by odometry which are sequential constraints $(j = i + 1)$, or from place recognition system which are non-sequential. The amount of error introduced by each constraint, weighed by its information, can be calculated as:

$$d_{ij}(\mathbf{x})^2 = r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) \tag{2}$$

and therefore the overall error, assuming all the constraints to be independent, is given by:

$$D^2(\mathbf{x}) = \sum d_{ij}(\mathbf{x})^2 = \sum r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) \tag{3}$$

where $d_{ij}(\mathbf{x})^2$ is the pairwise factor of the present variables in nodes i and j. The solution to graph-SLAM problem is to find a state $\mathbf{x}^*$ that minimizes the overall error.

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \sum r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) \tag{4}$$

III. Method

With the graph formulation we have to find the optimal state estimation in Eq. 4. At this point we face two different problems at the same time: figuring out which factor actually exist and which do not, and inferring the optimal state. In the following we summarize the RRR algorithm, for further details and algorithms the reader is referred to our recent work [12].

We start by dividing the factors into two sets; the first one, S contains the factors that exist by definition, from the sequential odometry sensor, and the second set R contains

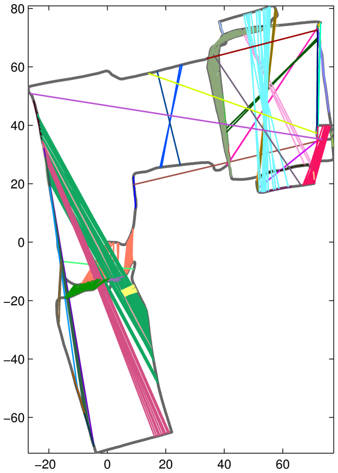


Fig. 2. Clustering loop closures. Each cluster is represented by a different colour.

the factors provided by the place recognition system, for which we have to determine if they are correct or not. Since all constraints are mutually independent, the error in (3) be written as:

$$D^2(\mathbf{x}) = \sum_{(i,j)\in S} d_{ij}(\mathbf{x})^2 + \sum_{(i,j)\in R} d_{ij}(\mathbf{x})^2 \qquad (5)$$

We further divide the set R into n disjoint subsets R_c, where each subset only contains topologically related constraints (sequences of links that relate similar portions of the robot trajectory) such that $R = \cup_{c=1}^{n} R_c$ and $\forall(i \neq j) R_i \cap R_j = \emptyset$. We term each of theses subsets as "clusters". An example of clustering can be seen in Fig. 2 where loop closures for one of the sessions of Bicocca dataset are shown. Each cluster is represented by a different colour. Then the error for set R can be written as:

$$\sum_{(i,j)\in R} d_{ij}(\mathbf{x})^2 = \sum_{c=1}^{n} \sum_{(i,j)\in R_c} d_{ij}(\mathbf{x})^2 = \sum_{c=1}^{n} d_{R_c}(\mathbf{x})^2 \qquad (6)$$

where $d_{R_c}(\mathbf{x})^2$ is the error contributed by the cth cluster. This simply means that the overall error introduced due to place recognition constraints is the sum of the individual errors of each cluster.

In the absence of any loop closing constraints, the best estimate of the nodes is the one that is constructed from the odometry. If the graph is optimized with just this information, the overall error would be zero because all the constraints agree with each other. Therefore, the error in (3) is caused practically only by the loop closing links. Once we iterate to find the optimal state, the error in the odometry is no longer zero. This increase in odometry error gives us a measure of the metric change that must take place so that the graph conforms to the place recognition constraints. Having clustered loop closures, the next step is to find if there may be outlier with in cluster. This is done by evaluating 'intra-cluster' consistency. Mathematically, for any cluster R_i to be individually consistent, the following two conditions must hold:

$$D_G^2(\mathbf{x}) = \sum_{(i,j)\in R_i} r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) + \sum_{(i,j)\in S} d_{ij}(\mathbf{x})^2 < \chi^2_{\alpha,\delta_G} \qquad (7)$$

where δ_G are the degrees of freedom of the whole graph. And,

$$D_l^2(\mathbf{x}) = r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) < \chi^2_{\alpha,\delta_l},\ (i,j) \in R_i \qquad (8)$$

ensures that if there are any outliers within the cluster they are omitted. δ_l are the degrees of freedom of each constraint. Now that we have clusters that are free from outliers inside them, the next step is to find out which clusters are consistent with each other, what we call 'inter-cluster consistency'. For a selected subsets of clusters C, we term the clusters in C to be jointly consistent if:

$$D_C^2(\mathbf{x}) = \sum_{c=1}^{|C|} \sum_{(i,j)\in R_c} r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) < \chi^2_{\alpha,\delta_C} \qquad (9)$$

and

$$D_G^2(\mathbf{x}) = D_C^2(\mathbf{x}) + \sum_{(i,j)\in S} r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) < \chi^2_{\alpha,\delta_G} \qquad (10)$$

This first criteria ensures that the present clusters are consistent with each other while the second one ensures the consistency of the clusters with the odometry links.

Having an initial set of consistent clusters, the algorithm iterates until it can no longer find any more consistent clusters, gathering all the clusters that are consistent given the all the available information.

IV. Experiments

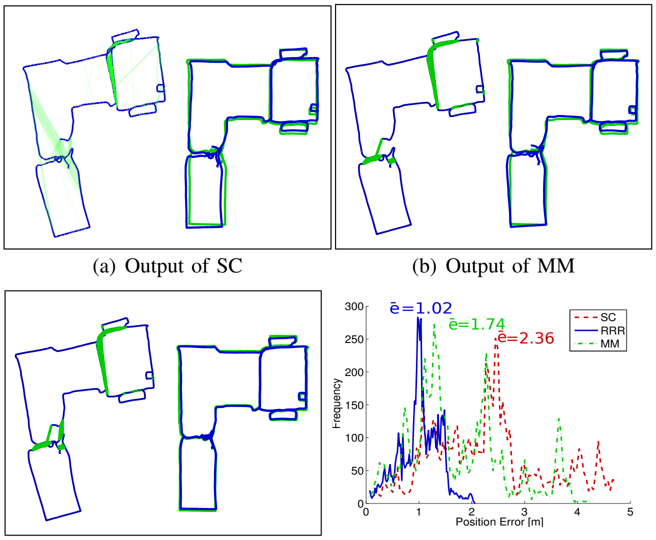
In this section we compare the performance of our method against the two other state-of-the-art approaches to robust back-end methods, namely Switchable constraints (SC) [10] and Max-Mixtures (MM) [11]. We explore the effect of number of outliers on all the three algorithms. We also present experiments that show the effect of varying amount of outliers and well as the odometry noise on our method.

Experiments are carried out on a dataset from the RAWSEEDS project [13]. We also present results for synthetic dataset, city10000. For all the experiments with RRR, the back end used was g2o configured with Gauss-Newton and every optimization was carried out for 4 iterations. The code used for SC and MM is available from the openslam website[1].

A. Comparison of Robust SLAM methods

1) Single Run: In this experiment we compare the performance of SC, MM and RRR in batch mode on a single run from the Bicocca-25b dataset. The loop closures come from a simple place recognition algorithm (the BoW stage described in [14]) with the minimum confidence parameter (α^-) set to 0.15. Under these settings, the algorithm is known

[1] http://openslam.org/vertigo.html, http://openslam.org/maxmixture.html



(a) Output of SC (b) Output of MM

(c) Output of RRR (d) ATE distributions

Fig. 3. Bicocca-25b experiment, see Fig. 2, with laser odometry and the constraints from the place recognition system. **(a):** The result of Sunderhauf et. al [10], **(b):** Olson et. al [11], **(c):** Output of RRR ***(a:left):*** Link colors proportional to the switch values **(b-c:left):** Accepted loop closures (green) **(a-c:right):** Optimized graph (blue) and ground truth (green). **(d):** The distribution of the Absolute Trajectory Errors for the final pose graphs against the ground truth (b,c,d:right).

	Mean (m)	Std. Dev (m).	Time (sec)
SC	2.358	1.059	0.88
MM	1.7353	0.923	**0.54**
RRR	**1.018**	**0.367**	8.45

TABLE I

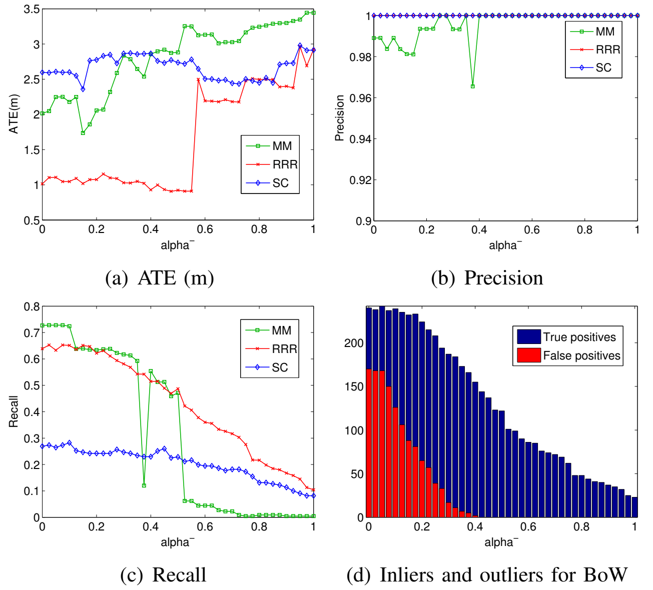
SUMMARY OF RESULTS FOR FIG. 3.

to have many false positives. The laser odometry along with the loop closures suggested by BoW are shown in Fig. 2.

As the performance metric for this experiment, we compare the Absolute Trajectory Error (ATE) for the three algorithms. ATE measures the global mismatch between the constructed map and the ground truth using root mean squared error. As a preprocessing step, a transformation that best aligns the constructed map with the ground truth is also calculated if the two maps are in different frames of reference. Tools for automatically calculating ATE are provided in the RAWSEEDS project [13].

SC was used with the default parameters given in the authors' code. In order to correctly determine the weight (w) and scaling (s) parameter for MM, a parameters sweep was carried out in the range $0.1 \leq$ s $\leq 10^{-19}$ and $0.1 \leq$ w $\leq 10^{-9}$. The parameters that provided the least Absolute Trajectory Error were selected for comparison (s=10^{-6}, w=10^{-3}). Both SC and MM were run until convergence. The results are presented in Fig. 3 and Table I.

It can be seen from Fig. 3(d) that RRR outperforms the other algorithms in this case in terms of ATE. Table I provides details of the ATE and its spread as well as the execution time for each algorithm. SC is the one which suffers more from drift in odometry for large loops. RRR on the other hand is the most expensive, time-wise, but generates the best results in terms of the overall ATE. This behaviour will be seen again for RRR in cases where there are a few correct loop closures. While the difference between the slowest and the fastest algorithm (MM and RRR) appears to be one order of magnitude, considering that the trajectory was collected over half an hour, this allows for near-real time use of the method.



(a) ATE (m) (b) Precision

(c) Recall (d) Inliers and outliers for BoW

Fig. 4. Comparison of RRR, SC and MM: Performance under varying number of outliers

In order to investigate the robustness of each algorithm to false positive loop closures, we compare how each algorithm behaves under varying amount of false positive loop closures in the next experiment.

2) Robustness to false positive loop closures: In this experiment, we compare the robustness of the RRR, SC and MM to varying amount of outliers in loop closures. The minimum confidence parameter for the front-end place recognition system was varied from 0 to 1 with increment of 0.025 to generate 41 different sets of loop closing constraints. The number of loop closures suggested vary from 446 to 23. The number of outliers and inliers as are present in the output from BoW are shown in Fig. 4(d).

SC was run with the default parameters as provided in the source code. MM was run with the parameters that gave the least ATE for the previous experiment (s=10^{-6}, w=10^{-3}).

The performance of the algorithms is compared based on three metrics: Precision, Recall and Absolute Trajectory Error. The results are shown in Fig. 4. SC does not make a binary decision about the correctness of a loop closing constraint but gives a value between 0 and 1. Therefore, for calculating the precision-recall curves any switch with a value greater than 0.5 was considered to be "on" and "off" otherwise. For MM, all the links in which the suggested loop closing constraint was more probable than the corresponding null hypothesis were selected and included in the precision-recall calculation.

The most important metric to look at in the context of robust back-ends is precision. Full precision means that the all the accepted constraints are topologically correct.

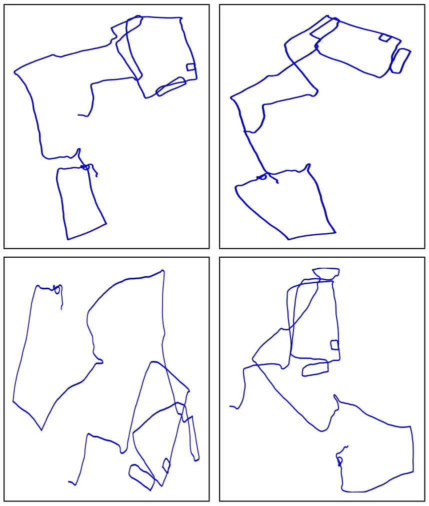


Fig. 6. Examples of odometry corrupted by noise : $2\sigma = 4.0$, 6.0, 8.0, 10.0 degrees

Secondly, we would like to have full precision with as much recall as possible. That means that while we only select the correct loop closures, we selected as many of them as possible.

As can be seen in the Fig. 4(b) both SC and RRR are able to provide full precision for all the 41 experiments. RRR has a greater recall which amounts to lower ATE. MM is confused by the presence of many outliers but as the number of outliers decreases, MM is also able to achieve full precision.

MM initially has greater ATE because of accepting some false positives but when it reaches full precision, it suffers from low recall. The lowest point in the ATE plot for MM is the point for which we calculated the parameters using a parameter sweep. This means that if we could calculate the correct parameters for every experiment, MM may perform better. On the other hand, this means that MM is very sensitive to the tuning parameters and that they do not depend on the trajectory, but rather on the distribution of the loop closing hypotheses.

B. *Evaluation of RRR under increasing odometry error*

Another interesting aspect to look at is the performance of the algorithm with varying odometry errors. The main metric of concern is again precision. We use the trajectory and loop closings given in Fig. 3(a) and add noise to orientation of every pose. This has a cumulative effect on the overall trajectory. We simulate 10 noise levels with noise varying from $2\sigma = 1$ degree to 10 degrees. For each noise level 100 random experiments were generated and RRR was run on them. Fig. 6 illustrates some of the corrupted trajectories.

The results are shown in Fig. 5. In this setup, errors can come from two sources. One is the error in odometry and the other is the error that might come from false positive or false negative loop closures.

Fig. 5(a) gives box plots for ATE of the experiments. The medians are marked by the line inside every box. We are still able to perform with full precision, although recall is seriously affected. It can be seen that the ATE is affected by the amount of noise and greater noise leads to greater error. With full precision, this means that most of the error is due to the own odometry noise and not due to false positives. In only one of the 1000 experiments, we accept a single false positive loop closure. Recall degrades with varying amount of noise.

C. *Synthetic dataset: city10000*

While real datasets provide us with a way of evaluating the performance of our algorithm when there are a few loop closings, synthetic datasets can be used to evaluate the performance in the presence of massive loop closing hypotheses, due for example to highly self-similar environments.

Here we evaluate the performance of our algorithm on a synthetic dataset: city10000 provided with iSAM [2]. The dataset simulates a run in a city with $10,000$ nodes and provides ground truth loop closings.

In order to evaluate the performance of our algorithm under varying number of false positives, randomly generated incorrect loop closures were added to the dataset starting from 100 to 1000 with a step of 100. For every level of outliers, 10 random experiments were generated and the RRR algorithm was run on them. The metrics calculated are ATE, precision and recall. For this dataset, the number of iterations for the optimizer were set to 5. Sample results are shown in Fig. 8 and results for calculated metric are given in Fig. 7.

Fig. 7(b) gives the precision. Since links were added at random, some of them are actually correct but do not exist in the ground truth. This is reflected in the corresponding ATE. If the loop accepted were actually false positives, this would severely corrupt the map, which is not the case as can be seen from the corresponding value of ATE in Fig. 7(a).

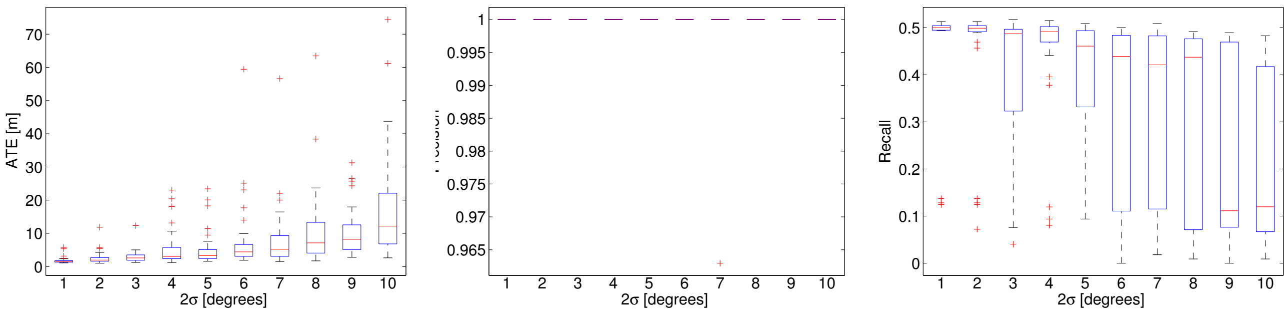
Recall is effected by the number of outliers. Table II gives the numerical results for the shown plots. One point to note is that while the median recall at worst is about 41% , the corresponding value of ATE, 0.11 m is not very large. This means that although we are discarding a lot of correct loop closures, we are not discarding anything important; the maps estimate is still very accurate.

The details for execution time are given in Table II. The algorithm takes a considerable time to generate the result. The table also gives the number of clusters in each case and time per cluster.

V. Discussion

The RRR algorithm is a consistency-based approach for a robust backend of a SLAM system. RRR works on any pose-graph built from sequential constraints (normally laser/visual or other forms of odometry) and loop closure edges.

We have presented the comparisons of RRR against two recently proposed approaches to robust back ends, namely Olson's Max-mixtures and Sünderhauf's switchable constraints. On real datasets where loop closing constraints are sparse, RRR is able to perform at full precision with a considerable recall, resulting in a smaller absolute trajectory error.

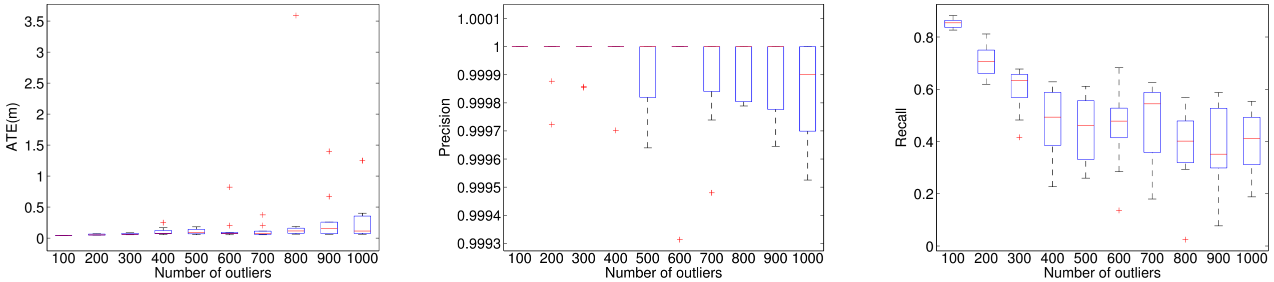


(a) ATE

(b) Precision

(c) Recall

Fig. 5. Effects of different level of odometry noise on RRR



(a) ATE in meters

(b) Precision

(c) Recall

Fig. 7. Results for 100 experiments on the city10000 dataset with varying amount of incorrect loop closures

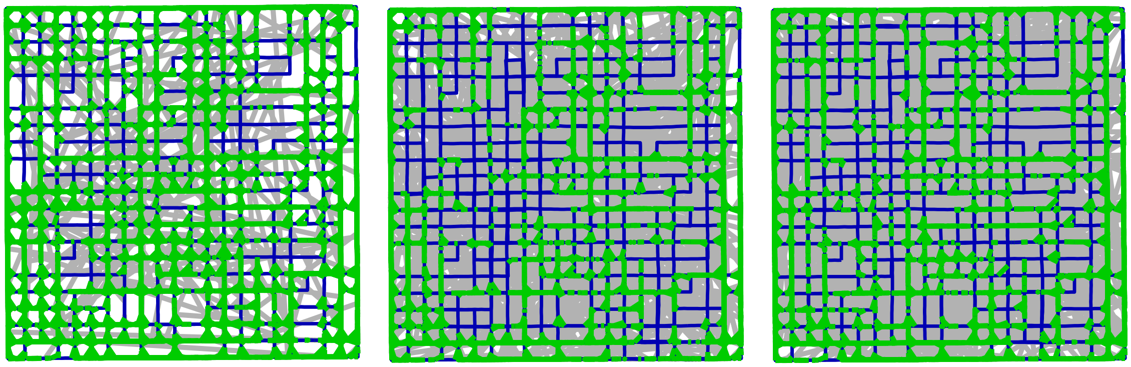


Fig. 8. Sample results for city10000. *From left to right:* Optimized graph after RRR in presence of 100, 500 and 1000 outliers. Accepted links in green, final optimized trajectory in blue and rejected links in gray.

outliers	ATE (m)		Precision		Recall		Total time (s)		clusters		time (s)/cluster	
	mean	median	mean	median	mean	median	mean	median	mean	median	mean	median
100	0.04	0.04	1.00	1.00	0.85	0.85	764.2	785.7	1661.3	1661.5	0.46	0.47
400	0.10	0.07	1.00	1.00	0.47	0.49	701.3	699.1	1954.0	1954.5	0.35	0.35
700	0.11	0.06	0.99	1.00	0.46	0.54	922.1	918.6	2246.6	2246.5	0.41	0.40
1000	0.26	0.11	0.99	0.99	0.38	0.41	1362.2	1261.2	2538.5	2540.5	0.53	0.49

TABLE II

ATE, PRECISION, RECALL FOR CITY10,000

The aim of SLAM is not just to build a good looking map but to use it further to carry out higher level tasks such as planning and navigation. In that regard, loop closing edges provide traversability information. Switchable constraints may provide a good map estimate, but the switches are governed by a continuous function, where as loop closures should either be completely accepted or completely rejected. The distribution of the switch values for the experiment shown in Fig. 3 is given in Fig. 9. Although most of the loop closures are either rejected (in the first bin) or accepted (last bin), there are still quite a few loop closures which remain in the map and provide topologically inconsistent

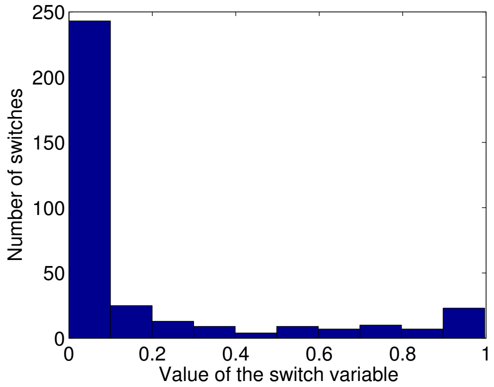


Fig. 9. The distribution of the values governing the behaviour of Switch Variables. Ideally, all the values should be either 0 or 1

links. In certain cases, this might cause problems when the map is actually being used to carry out higher level tasks. Max-mixtures is a mathematically sound approach, but that requires a great deal of parameter tuning, as has been shown in comparisons. From a topological point of view, MM introduces null hypotheses into the graph which replace incorrect loop closures. Before using the map for planning or any higher level task, such links may need to be pruned from the map. Both SC and MM are fast and very useful when there are dense loop closing hypotheses such as in synthetic datasets. However, in real world scenarios, loop closing are comparatively sparse and there is weak evidence for MM and SC to work on. In those cases, RRR out performs both. In synthetic datasets RRR also achieves good results. While execution time is greater, it is still real time for the duration of the experiments.

We have demonstrated that RRR is able to correctly identify false positives and removes them to robustly recover the correct map estimate. We have investigated the effect of number of outliers and odometry error on the method and, even in the presence of a considerable amount of noise, RRR is able to work at full precision, which is a highly desirable quality for a robust back-end. RRR is more robust than the alternative state-of-the-art methods and always provides a topological correct graph structure given all the gathered evidence.

References

[1] F. Lu and E. Milios, “Globally consistent range scan alignment for environment mapping,” *Autonomous Robots*, vol. 4, pp. 333–349, 1997.

[2] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, “iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering,” in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Shanghai, China, May 2011.

[3] G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, and C. Hertzberg, “Hierarchical optimization on manifolds for online 2d and 3d mapping,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 273 –278.

[4] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g2o: A general framework for graph optimization,” in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[5] P. Huber, “Robust regression: asymptotics, conjectures and monte carlo,” *The Annals of Statistics*, vol. 1, no. 5, pp. 799–821, 1973.

[6] D. Rosen, M. Kaess, and J. Leonard, “An incremental trust-region method for robust online sparse least-squares estimation,” in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, St. Paul, MN, May 2012, pp. 1262–1269.

[7] ——, “Robust incremental online inference over sparse factor graphs: Beyond the Gaussian case,” in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Karlsruhe, Germany, May 2013, to appear.

[8] E. Olson, “Recognizing places using spectrally clustered local matches,” *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1157–1172, December 2009.

[9] A. Ranganathan and F. Dellaert, “Online probabilistic topological mapping,” *The International Journal of Robotics Research*, vol. 30, no. 6, pp. 755–771, May 2011. [Online]. Available: http://ijr.sagepub.com/content/early/2011/01/23/0278364910393287.abstract

[10] N. Sünderhauf and P. Protzel, “Switchable Constraints for Robust Pose Graph SLAM,” in *Proc. IEEE/RJS Int. Conference on Intelligent Robots and Systems*, Vilamoura, Portugal, 2012.

[11] E. Olson and P. Agarwal, “Inference on networks of mixtures for robust robot mapping,” in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[12] Y. Latif, C. Cadena, and J. Neira, “Robust Loop Closing Over Time,” in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[13] RAWSEEDS, “Robotics advancement through Webpublishing of sensorial and elaborated extensive data sets (project FP6-IST-045144),” 2009, http://www.rawseeds.org/rs/datasets.

[14] C. Cadena, D. Gálvez-López, J. Tardós, and J. Neira, “Robust place recognition with stereo sequences,” *IEEE Transaction on RObotics*, vol. 28, no. 4, pp. 871 –885, 2012.