To appear in Proc. of IEEE Conf. on Intelligent Robots and Systems (IROS), 2012. DOI: not yet available

Switchable Constraints for Robust Pose Graph SLAM

Niko Sünderhauf and Peter Protzel

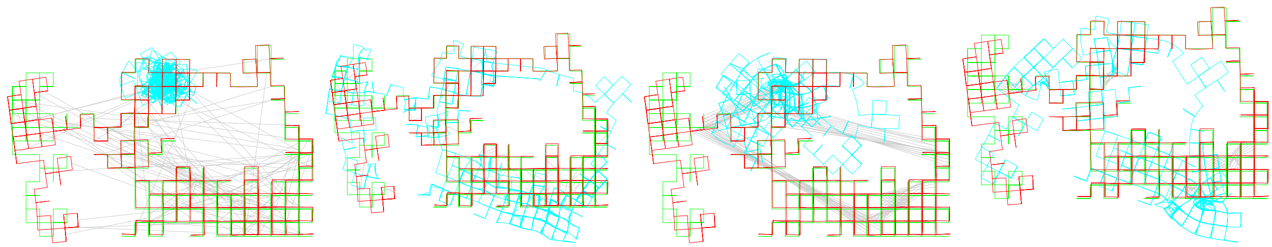


Fig. 1: Exemplary results of the proposed robust SLAM back-end on the synthetic Manhattan world dataset [10] that contains 3500 poses and 2099 loop closures. In these examples, we corrupted the dataset by introducing 100 additional wrong loop closures that might have been produced due to data association errors (e.g. failed place recognition) in the SLAM front-end. Current back-ends like g^2o [6] are not able to converge to a correct solution (shown in blue) despite being supported by so called robust cost functions like the Huber function [1]. Our robust solution (red) that uses switchable constraints correctly discards the wrong loop closure candidates (visible as grey links) *during* the optimization and converges to a correct solution. For comparison, the ground truth is plotted in green. Our robust back-end was able to cope with 1000 outliers on a number of 2D and 3D datasets. Notice that the outlier loop closure constraints have been added following different policies (from left to right: random, local, random group, local group) which are explained later on.

***Abstract*— Current SLAM back-ends are based on least squares optimization and thus are not robust against outliers like data association errors and false positive loop closure detections. Our paper presents and evaluates a robust back-end formulation for SLAM using switchable constraints. Instead of proposing yet another appearance-based data association technique, our system is able to recognize and reject outliers *during* the optimization. This is achieved by making the topology of the underlying factor graph representation subject to the optimization instead of keeping it fixed. The evaluation shows that the approach can deal with up to 1000 false positive loop closure constraints on various datasets. This largely increases the robustness of the overall SLAM system and closes a gap between the sensor-driven front-end and the back-end optimizers.**

I. Introduction

After recent advances in the field of optimization-based SLAM, efficient algorithms that exploit the sparsity of SLAM have been proposed, e.g. among others [2]–[4], [6]. Implementations of these approaches in the form of open-source libraries are available to the robotics community, ready to be applied. These well-documented frameworks support batch and incremental processing and thus can be used to solve both the full and the online SLAM problem, including pose graph SLAM and landmark-based SLAM.

Current state of the art SLAM back-ends like g^2o [6], or iSAM2 [2] are least squares optimizers and as such, they are naturally not robust against outliers. The problem is widely acknowledged but usually ignored: While the front-end is responsible for sensor data processing, data association and graph construction, the back-end optimizer considers the data association problem solved. That means the back-end relies heavily on the front-end and expects it to produce a topologically correct factor graph representation. Although any failure in the data association can have catastrophic implications on the resultant map and robot state estimates, state of the art back-ends do little or nothing at all to mitigate these potentially severe effects.

In our understanding, state of the art SLAM back-ends and complete SLAM systems lack the necessary robustness to cope with data association errors that will inevitably appear especially – but not exclusively – in long-term operations outside controlled lab environments. Since robustness is the key to transfer SLAM from academia to a broader variety of every-day applications, our work focuses on making SLAM robust. This paper therefore evaluates a robust back-end for optimization-based SLAM systems. The core novelty of this robust back-end compared to state of the art approaches are the *switchable constraints*. While we described a preliminary version of the system in [13], [14] and applied it to a large-scale real-world urban dataset, no in-depth evaluation has been conducted so far. As we are going to see, the proposed system is able to cope with an extremely high amount of outliers in the SLAM problem formulation. Exemplary results of the system on the Manhattan dataset are shown in Fig. 1.

The next section quickly reviews the switchable constraints and their application for robust pose graph SLAM. The evaluation and presentation of results and the discussion of failure cases follow in sections III and IV.

II. Switchable Constraints

Since false positive loop closures are expressed as additional constraint edges in the factor graph representation of the SLAM problem [5], our main idea to increase the robustness of SLAM back-ends is that the topology of the graph should be subject to the optimization instead of keeping it fixed. This is achieved by using *switchable constraints*.

The authors are with the Department of Electrical Engineering and Information Technology, Chemnitz University of Technology, 09126 Chemnitz, Germany. niko.suenderhauf@etit.tu-chemnitz.de

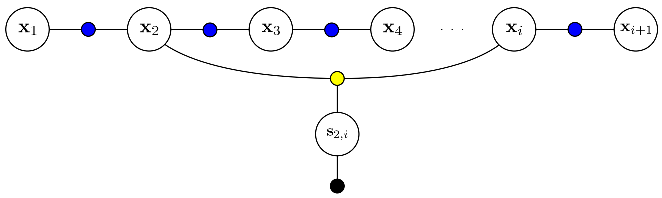


Fig. 2: Factor graph representation of the robustified pose graph SLAM formulation evaluated in this paper. Individual robot pose nodes are connected by odometry factors (blue). The switch variable $s_{2,i}$ governs the loop closure factor (yellow). Depending on the value assigned to the switch variable s_{ij}, the loop closure factor is switched on or off, i.e. it is activated or deactivated as part of the optimization process. The switch variable is governed by a prior factor (black) that penalizes the deactivation of loop closures.

To achieve the desired switchable behaviour, we introduce another type of hidden variable into the problem formulation: A *switch variable* s_{ij} is associated with each constraint factor that could potentially represent an outlier. The optimization now works on an augmented problem, searching for the joint optimal configuration of the original variables and the newly introduced switch variables, hence searching the optimal graph topology.

Our proposed problem formulation for robust pose graph SLAM is:

$$\begin{aligned} X^*, S^* = \underset{X,S}{\operatorname{argmin}} & \underbrace{\sum_i \|f(\mathbf{x}_i, \mathbf{u}_i) - \mathbf{x}_{i+1}\|^2_{\mathbf{\Sigma}_i}}_{\text{Odometry Constraints}} \\ & + \underbrace{\sum_{ij} \|\Psi(s_{ij}) \cdot (f(\mathbf{x}_i, \mathbf{u}_{ij}) - \mathbf{x}_j)\|^2_{\mathbf{\Lambda}_{ij}}}_{\text{Switchable Loop Closure Constraints}} \\ & + \underbrace{\sum_{ij} \|\gamma_{ij} - s_{ij}\|^2_{\mathbf{\Xi}_{ij}}}_{\text{Switch Prior Constraints}} \end{aligned} \tag{1}$$

Fig. 2 illustrates the factor graph corresponding to this problem formulation. In the following, we will explain this formulation step by step. Notice that compared to the conventional, non-robust formulation, the odometry constraints remain unchanged.

A. The Switch Variables and Switch Function

The loop closure constraints have been augmented by a multiplication with the function $\Psi(s_{ij})$, which we call the *switch function*. This switch function is defined as $\Psi : \mathbb{R} \to [0, 1]$, i.e. it is a mapping from the continuous real numbers to the interval $[0, 1]$, defined on $\mathbb{R}$. The idea behind the switch variables is that the influence of a loop closure constraint between the poses $\mathbf{x}_i$ and $\mathbf{x}_j$ can be removed by driving the associated switch variable s_{ij} to a value so that $\omega_{ij} = \Psi(s_{ij}) \approx 0$.

Different switch functions can be defined, e.g. a step function, or a sigmoid. However, the sigmoid function shows an undesired plateau behaviour where the gradient is very small over a large range of values and where the iterative optimizer can become stuck. Although we previously proposed to use the sigmoid [13], [14], more elaborate experiments showed that a simple linear function of the form $\omega_{ij} = \Psi^{\text{lin}}(s_{ij}) = s_{ij}$ results in a better convergence behaviour if we constrain $0 \leq s_{ij} \leq 1$.

The influence of the switch variables can be described and understood in two equivalent ways: In the topological interpretation, a switch can enable or disable the constraint edge it is associated with, thus literally remove it from the graph topology. In the probabilistic interpretation, the switch variable influences the information matrix of the factor it is associated with and can drive it from its original value to zero, thus increasing the covariance associated with this factor until infinity. The resulting information matrix is given as $\hat{\mathbf{\Lambda}}_{ij}^{-1} = \Psi(s_{ij})^2 \cdot \mathbf{\Lambda}_{ij}^{-1}$. It has been shown that both interpretations are equivalent [11], [14].

B. The Switch Prior Constraint

The switch prior constraints are necessary to anchor the switch variables at their initial values. Since it is reasonable to initially accept all loop closure constraints, a proper and convenient initial value for all switch variables would be $s_{ij} = 1$ when using the linear switch function Ψ^{lin}. For the following, we call these initial values γ_{ij}. Like any other variable or observation in our probabilistic framework, the switch variables s_{ij} are modelled as normally distributed Gaussian variables. The initial value is used as mean of the distribution, so that $s_{ij} \sim \mathcal{N}(\gamma_{ij}, \mathbf{\Xi}_{ij})$. Notice that we have to determine a proper value for the switch prior covariance $\mathbf{\Xi}_{ij}$ which will be explained later during the evaluation. This value is a free parameter of the proposed robust back-end formulation.

III. Preparing the Evaluation

A. Used Datasets

TABLE I: The datasets used during the evaluation.

Dataset	synthetic / real	2D/3D	Poses	Loop Closures
Manhattan (original)	synthetic	2D	3500	2099
Manhattan (g^{2}o version)	synthetic	2D	3500	2099
City10000	synthetic	2D	10000	10688
Sphere2500	synthetic	3D	2500	2450
Intel	real	2D	943	894
Parking Garage	real	3D	1661	4615

In order to show the versatility and general feasibility of the proposed approach, six very different datasets were used for the evaluation. Table I lists and summarizes their important properties. The synthetic datasets were created from simulation, while the two real-world datasets have been recorded in a 2D (Intel) or 3D (Parking Garage) environment respectively. These datasets are publicly available to the community and have been used as examples and benchmarks in a number of SLAM publications before.

The Manhattan dataset is available in two versions: The original dataset was first published by [10] and the second version was included in the open source implementation of g^{2}o [6]. The difference between the two versions is the quality of the initial estimate: It is much closer to the ground truth for the g^{2}o version than for Olson's original dataset. For evaluation purposes, having the same dataset with different initializations allows us to see the influence of the initial estimates on the overall behaviour of the back-end system. We will therefore use both versions and indicate whether the original or the g^{2}o version has been used.

The datasets City and Sphere shipped with the open-source implementation of iSAM [3]. The two real-world datasets Intel and Parking Garage are part of g^{2}o [6]. For the two latter real-world datasets, no ground truth information is available. Instead, the estimation results for the outlier-free dataset are used as a pseudo ground truth when necessary.

B. General Methodology

The robust back-end formulation was implemented by extending the framework g^{2}o [6]. The used datasets are pose graphs consisting of odometry measurements and loop closure constraints. They are free of outliers, i.e. all loop closure constraints are correct. To evaluate and benchmark the robust back-end, the datasets are spoiled by additional, wrong loop closure constraints. That means, loop closure constraints which do not connect corresponding poses and thus are outliers are added to the dataset. Given the spoiled datasets, the performance of the robust back-end can be evaluated using two different methods:

- Use a suitable error metric to compare the resulting trajectory against the ground truth solution and the solution reached by state of the art non-robust back-ends. The relative pose error metric RPE [7] was chosen for the evaluation.
- Use precision-recall statistics to identify how many of the added wrong outlier constraints could be identified and disabled, while maintaining the correct loop closure constraints.

C. Policies for Adding Outlier Loop Closure Constraints

For the evaluation, the datasets are spoiled by adding false positive loop closure constraints between two poses $\mathbf{x}_i$ and $\mathbf{x}_j$. As discussed before, in real applications, these outliers might have been introduced by the front-end after a failed place recognition etc. The indices i and j are determined using four different policies, which are explained below and illustrated in Fig. 1.

a) Random Constraints: This policy adds constraints between two randomly chosen pose vertices $\mathbf{x}_i$ and $\mathbf{x}_j$, i.e. the indices i and j are drawn from a uniform distribution over all available indices. Most of the constraints that are created using this policy will span over large areas of the dataset since they connect two distant poses $\mathbf{x}_i$ and $\mathbf{x}_j$.

b) Local Constraints: Following this policy, constraints are added only locally. That means that the first pose vertex of the constraint is chosen randomly from all available vertices. The second vertex however is chosen so that it is in the spacial vicinity of the first vertex. This follows the intuition that in reality nearby places are more likely to appear similar than distant places and thus false place recognitions are more likely to be established between these nearby poses.

c) Randomly Grouped Constraints: In real front-ends, false positive loop closure constraints can be expected to appear in consistent groups. Imagine a robot driving through a corridor or street where the visual appearance is very similar to that of another corridor or street already mapped. The front-end may erroneously recognize loop closures for several successive frames while the robot traverses the ambiguous part of the environment. This is simulated by the two grouped policies. The randomly grouped policy first picks i and j randomly from all available indices, but then adds 20 successive and consistent constraints between the poses with indices $i \dots i+20$ and $j \dots j+20$.

d) Locally Grouped Constraints: This last policy is a combination of the local and grouped policies and creates groups of short constraints that connect nearby places. Following this policy, the first index i is chosen randomly. The second index j is chosen from the vicinity of i, like with the local constraint policy above. Then 20 successive constraints between the vertices with indices $i \dots i+20$ and $j \dots j+20$ are added.

IV. EVALUATION

Now that all necessary preliminary information are given, the evaluation of the proposed robust-back end commences.

A. The Influence of Ξ_{ij} on the Estimation Results

Remember that the formulation of the proposed robust back-end in (1), involved the switch prior constraints $\|\gamma_{ij} - s_{ij}\|^2_{\Xi_{ij}}$. The exact value of the switch prior variances Ξ_{ij} could not be deduced in a mathematically sound way. It rather has to be set empirically. Therefore, the first question we want to explore in this evaluation is the influence of Ξ_{ij} on the estimation results and what a suitable value for Ξ_{ij} would be.

Ξ_{ij} controls the penalty the system gains for the deactivation of a loop closure constraint. By adapting this value individually for each constraint, the front-end could express a degree of certainty about that particular loop closure constraint. A small value of Ξ_{ij} leads to a high penalty if it is deactivated. Thus the front-end would assign small Ξ_{ij} to loop closures it is very certain about and large Ξ_{ij} to those loop closures that appear more doubtful. If the front-end is not capable of determining an individual degree of certainty, all constraints could be assigned the same Ξ_{ij}. For the following, to show the general influence of Ξ_{ij}, we assume the same value $\Xi_{ij} = \xi$ was assigned to all constraints.

1) Methodology: To determine the influence of ξ, the mean relative pose errors $\mathrm{RPE}_{\mathrm{pos}}$ were determined for three different values of random outliers (1, 10, and 100) and varying ξ on the Manhattan dataset. Fig. 3 shows the results. Every data point represents the mean $\mathrm{RPE}_{\mathrm{pos}}$ of 10 trials for that particular pairing of ξ and number of outliers. In total, 720 optimization runs were conducted to create the plot.

2) Results and Interpretation: The curves in Fig. 3 reveal that the value of $\xi = \Xi_{ij}$ indeed influences the quality of the optimization result which is measured by the RPE metric. The second insight is that the quality of the estimation drops

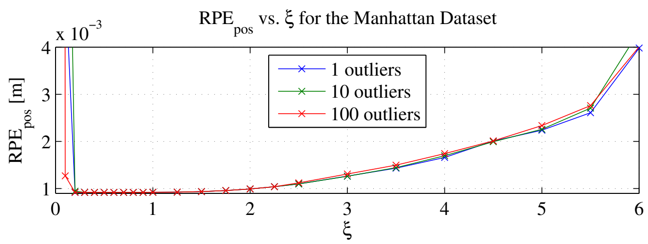


Fig. 3: The influence of $\Xi_{ij} = \xi$ on the estimation quality. The figure shows the mean translational error $\mathrm{RPE}_{\mathrm{pos}}$. The best choice for ξ is in the interval $0.3 \leq \xi \leq 1.5$ where the error measures are relatively constant and have their minimum.

drastically if ξ is too small or too large. This can be seen from the raising RPE measures for $\xi < 0.3$ and $\xi > 2.0$. The most interesting result however is, that the RPE stays relatively constant for a broad range of values for ξ between 0.3 and 1.5, where the error is minimal. This behaviour is independent from the number of outliers.

If the front-end is not able to assign sound individual values for Ξ_{ij}, it is save to set all $\Xi_{ij} = 1$, since this value is close to the individual optimal choice of Ξ_{ij} for a large range of outliers. Therefore, for all evaluations that follow, all Ξ_{ij} have been set to 1. Although that value was determined using the Manhattan dataset, it proved to be a suitable value for all other evaluated datasets and even for applications beyond SLAM [12].

B. The Robustness in the Presence of Outliers

After a sound value for the free parameter of the system, Ξ_{ij} has been determined, we can now examine how well the proposed back-end performs in the presence of outliers.

1) Methodology: A large number of test cases were considered for the different datasets. The number of added outliers was varied between 0 and 1000, using all of the four policies described above. For each number of additional wrong outliers, 10 trials per policy were calculated, resulting in a total of 500 trials per dataset. For each trial the RPE was determined. Notice from Table I that the number of correct loop closure constraints in the datasets varied between 10688 (City10000) and only 894 (Intel). Therefore, 1000 additional outlier loop closures is a huge number, leading to outlier ratios between 9.4% for the City10000 dataset and almost 112% for the Intel dataset. In real applications, we can expect much smaller outlier ratios in the range of a few percent or even below, depending on how sophisticated the front-end is built. The evaluation here uses much higher outlier ratios to demonstrate the remarkable robustness of the system.

2) Results and Interpretation: Table II summarizes the results for the different datasets. The minimum, maximum, and median $\mathrm{RPE}_{\mathrm{pos}}$ measures are listed, as well as a success rate which measures the percentage of correct solutions.

We see that the overall success rates are very high. In total, from all 2500 trials, only two failed, leading to success rates equal or close to 100%. The two failure cases and the special case of the Parking Garage dataset which does not appear in the table are discussed in detail in section IV-D.

TABLE II: Overall $\mathrm{RPE}_{\mathrm{pos}}$ metric for the different datasets, with $0\ldots1000$ outliers using all policies and 500 trials per dataset.

Dataset	max outl. ratio	min $\mathrm{RPE}_{\mathrm{pos}}$	max $\mathrm{RPE}_{\mathrm{pos}}$	median $\mathrm{RPE}_{\mathrm{pos}}$	success rate
Manhattan (g^2o)	47.6%	0.0009	0.0009	0.0009	100%
Manhattan (orig.)	47.6%	0.0009	5.9659	0.0009	99.8%
City10000	9.4%	0.0005	0.0005	0.0005	100%
Sphere2500	40.8%	0.0953	18.1674	0.0964	99.8%
Intel	111.9%	0.2122	0.2147	0.2132	100%

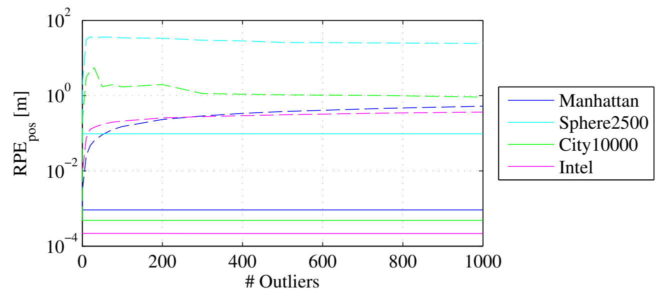


Fig. 4: Comparison of RPE measures between the proposed robust (solid line) and the state of the art non-robust back-ends (dashed). Notice how the robust solution is up to two orders of magnitude more accurate for large numbers of outliers and stays constant, independently of the amount of outliers. The non-robust solution was supported by the Huber cost function, as proposed in [6].

We want to remark that the two failure cases for the original Manhattan dataset and for the sphere world dataset could be successfully resolved by using the Huber cost function in combination with the proposed back-end. If we allow this further extension (which comes at the cost of slower convergence, due to the partially linear cost function), we can conclude that except for the Parking Garage dataset, a success rate of 100% was reached.

Fig. 4 compares the RPE of the proposed robust back-end to that of the non-robust state of the art formulation for all datasets and the different numbers of outliers. The robust back-end performs orders of magnitudes better, since the state of the art approach is not able to cope with outliers, despite being supported by the Huber cost function [1].

While the RPE metric compares the deviation of the estimated trajectory from the ground truth, precision-recall statistics allow us to determine how well the proposed robust technique is able to identify and disable the outlier constraints while leaving the true positive (i.e. correct) constraints intact. A system operating at a precision and recall rate both equal to 1 would be optimal, since all false positives are disabled, while all true positives are left untouched.

From Fig. 5 we can see that the results for the proposed back-end almost reach that point of optimal performance. For the datasets Intel, City10000, and Manhattan (g^2o), the recall is exactly 1 for a large span of precision. This means that there are values for ω_{ij} where all false positive constraints would be considered disabled, while a large amount of true positives are enabled. For the datasets sphere2500 and Olson's version of the Manhattan world, the recall is slightly smaller,

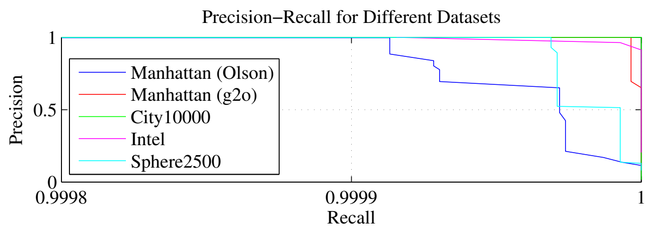


Fig. 5: Precision-recall statistics for the various datasets. Notice the scale of the X-axis, representing recall. The results indicate a close to optimal performance of the proposed system.

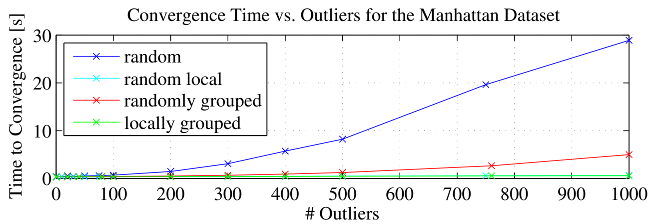


Fig. 6: Convergence time (on an Intel Core 2 Duo) for different outlier policies for the Manhattan dataset (g^2o version). Notice that the two local policies require much less time for convergence than the non-local policies.

since the statistics include the two failure cases that occurred with these two datasets and were mentioned before. However, we have to regard that the back-end never really performs a binary decision on whether a constraint is supposed to be active or deactivated. It is only the precision-recall benchmark that emulates such a behaviour. Overall, the system reached a recall of over 99.99% at 100% precision for all tested datasets.

C. Convergence and Runtime Behaviour

The influence of the number of outliers on the runtime behaviour for the Manhattan dataset can be seen from Fig. 6. Interestingly, the results are very different depending on the outlier policy that was used to add the false positives to the datasets. For the two non-local policies, the required time until convergence increases faster with the number of outliers than for the local policies. Obviously the non-local outlier constraints that often connect two very distant places in the dataset are more difficult to resolve. The same effects were observed for all other datasets.

How the estimation error is minimized during the optimization is visible from Fig. 7. These plots reveal that the optimizer behaves very differently, depending on the structure of the dataset and the outlier policy used. For some datasets like the g^2o version of the Manhattan dataset or the Intel dataset, the χ^2 error drops quickly and monotonically. For others however, the Gauss-Newton optimizer does not decrease the χ^2 error monotonically, but rather even increases it before finally finding its minimum. This behaviour indicates the difficult non-convex structure of the error function that is to be minimized.

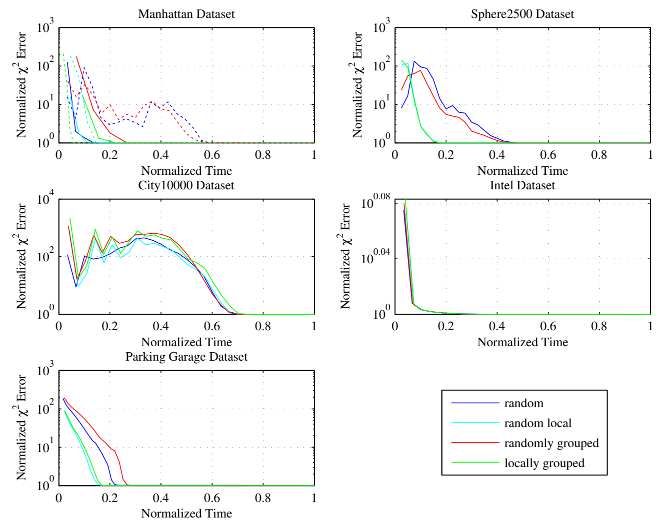
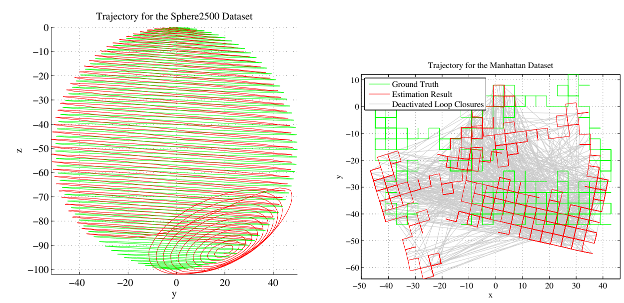


Fig. 7: Convergence behaviour for all datasets and all four policies with 1000 outliers. Both the time and the χ^2 errors have been normalized so that the time to convergence and the final residual error correspond to 1. The four outlier policies are color coded. For the Manhattan dataset the solid lines correspond to the g^2o version, while the dashed lines are for Olson's original dataset. Notice that the convergence behaviour is very different depending on the dataset and also the policy in some cases. Due to the Gauss-Newton algorithm used here, the optimizer first steps into regions of higher error before converging towards lower error measures.



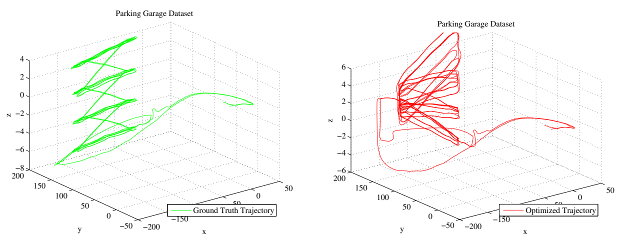
(a) (b)

Fig. 8: Two failure cases for the Sphere World dataset with 300 random outliers in (a) and the Manhattan dataset (Olson's original) containing 750 random outliers. Notice how the maps are still locally consistent. Both cases could be successfully resolved by combining the robust back-end with the Huber cost function.

D. Discussion of the Failure Cases

1) Manhattan and Sphere Datasets: As we saw in Table II, from 2500 trials conducted using the datasets Manhattan, Sphere, City and Intel with up to 1000 added outlier constraints, only 2 trials failed to converge to a correct solution. One of these failures occurred with the Sphere2500 dataset and the other one occurred with Olson's version of the Manhattan dataset. Both failure cases are illustrated in Fig. 8. We can see from the figures that although the resulting maps are significantly distorted when compared to the ground truth on a global level, they are still locally intact.

The main reason for that beneficial behaviour is that apparently only *single* false positives are not deactivated correctly,



(a) (b)

Fig. 9: The parking garage dataset is in particular difficult: (a) shows the ground truth trajectory from the side. An exemplary failure case is illustrated in (b). See the text for further discussion and explanation.

leading to punctual errors that cause global distortion but retain local consistency. In the examples above, exactly *one* false positive was incorrectly *not* disabled. On the other hand, that means that still 299 out of 300 or 749 out of 750 false positive loop closure constraints were correctly disabled.

As mentioned before, both of the failure cases for the sphere and Manhattan datasets could be correctly resolved by combining the robust back-end with the Huber error function [1], using the default kernel width of 1.0.

2) The Parking Garage Dataset: The parking garage dataset is a particular difficult dataset and the robust back-end was not able to perform better than the non-robust approach. Fig. 9(a) shows the pseudo ground truth trajectory (generated from the outlier-free data, since no real ground truth is available) from the side. Also notice that the z-axis is scaled differently to better show the spatial structure of the data. The dataset was recorded in a parking garage with four parking decks, which are clearly visible in Fig. 9(a). The single decks are connected by only two strands of odometry constraints that originate from the driveways.

The problems arising from this sparse connection structure can be seen in 9(b) which is an exemplary result. In this typical example, the proposed robust back-end failed to deactivate a group of false positive loop closures between the parking levels, leading to a corrupted result. The reason for the failures is the insufficient amount of information on the relative pose of the individual decks, due to the small number of constraints between these decks. Since the SLAM system has no further knowledge about the structure of the environment, e.g. that certain regions of the map can never intersect or are required to be level, the error introduced by the false positive loop closure requests cannot be resolved by the proposed back-end.

V. Conclusions

This paper evaluated a method to identify and reject outliers in the back-end of a SLAM system by using switchable constraints. The feasibility of the proposed approach has been demonstrated and evaluated on a number of standard datasets. The datasets used for the evaluation are available on our website, along with code that shows the implementation of the switchable constraints for g^2o. The robust back-end successfully solved SLAM problems with a large number – up to 1000 – outlier loop closure constraints, both in 2D and 3D, using synthetic and real-world datasets. The proposed approach was shown to outperform state of the art approaches by orders of magnitude, since these are not able to cope with outlier constraints.

A typical failure case was discussed and we conclude that degenerate environments consisting of distinctive parts that are only sparsely interconnected are prone to errors if false positive loop closure constraints are established between these almost independent parts. Further high-level knowledge about the environment and its spatial structure may be necessary to successfully resolve these situations.

Notice that the system's feasibility to work on large-scale real-world datasets has been demonstrated before [14]. In parallel work [12] we also found that the proposed approach is not limited to SLAM problems, but can be beneficially applied to other domains where least squares problems have to be solved but outliers have to be suspected, like multipath mitigation in GNSS-based localization.

Future work will include a comparison with alternative methods proposed by other authors [8], [9] that were published after this paper has been written.

References

[1] Peter J. Huber. Robust regression: Asymptotics, conjectures and monte carlo. *The Annals of Statistics*, 1(5):799–821, 1973.

[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*, 2011.

[3] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(6), 2008.

[4] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *International Journal of Robotics Research (IJRR)*, 29(10), 2010.

[5] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[6] 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)*, 2011.

[7] Rainer Kümmerle, Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner. On Measuring the Accuracy of SLAM Algorithms. *Auton. Robots*, 27:387–407, 2009.

[8] Yasir Latif, Cesar Cadena, and José Neira. Robust loop closing over time. In *Proceedings of Robotics: Science and Systems (RSS)*, Sydney, Australia, July 2012.

[9] Edwin Olson and Pratik Agarwal. Inference on networks of mixtures for robust robot mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, Sydney, Australia, July 2012.

[10] Edwin Olson, John Leonard, and Seth Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Inl. Conf. on Robotics and Automation, ICRA*, 2006.

[11] Niko Sünderhauf. *Robust Optimization for Simultaneous Localization and Mapping*. PhD thesis, Chemnitz University of Technology, 2012.

[12] Niko Sünderhauf, Marcus Obst, Gerd Wanielik, and Peter Protzel. Multipath Mitigation in GNSS-Based Localization using Robust Optimization. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, 2012.

[13] Niko Sünderhauf and Peter Protzel. BRIEF-Gist – Closing the Loop by Simple Means. In *Proc. of IEEE Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[14] Niko Sünderhauf and Peter Protzel. Towards a Robust Back-End for Pose Graph SLAM. In *Proc. of IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2012.