Cluster-based Penalty Scaling for Robust Pose Graph Optimization

Fang Wu[1] and Giovanni Beltrame[2]

***Abstract*— Robust pose graph optimization is essential for reliable pose estimation in Simultaneous Localization and Mapping (SLAM) system. Due to the nature of loop closures, even one spurious measurement could trick the SLAM estimator and severely distort the mapping results. Existing methods to avoid this problem mostly focus on ensuring local measurement consistency by evaluating measurements independently, often requiring parameters that are difficult to tune. This paper proposes a cluster-based penalty scaling (CPS) method to ensure both the local and global consistency by first evaluating the edge quality locally, and then integrating this information into the optimization formulation.**

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) [1] has been widely studied in the robotic community during the past decade. By providing a global and evolving representation of the environment, SLAM is a key component for true autonomy of robotic platforms.

The architecture of state-of-the-art SLAM pipelines can be divided into two components: the *front-end* and the *back-end*. The *front-end* extracts geometrical information from raw measurements of various sensor modules, such as cameras or lidar systems, and the *back-end* performs inference on the front-end results by solving a Maximum A Posteriori (MAP) estimation problem. Pose graph optimization (PGO) is a compact formulation of this MAP estimation problem. Information from the *front-end* typically consists of a pose graph with two types of edges: odometry edges link sequential poses, while loop closures edges link arbitrary poses identified to be in the same location by a place recognition algorithm. Each edge represents the transformation between two poses with a covariance term, which accounts for uncertainty. When solving the MAP problem, both transformation and covariance contribute to a cost term in the objective function. Due to the sparsity of the matrices involved in PGO, local optimization techniques [2], [3], [4] such as Gauss-Newton can be used to efficiently solve the problem. However, the long-term data association that generates loop closure edges has to deal with higher noise and it is more susceptible to mistakes. We refer to these incorrect loop closure edges as outliers, and correct loop closure edges as inliers in this paper.

Outliers could appear for different reasons: as an example, a low threshold could cause false positives during place recognition. This false positive would in turn cause two very different images (or scans) to be used for motion estimation, generating outliers with high uncertainty. This type of outliers can be easily handled since the covariance matrix serves as a set of weights to scale the residual error. A much more difficult type of outliers occur due to perceptual aliasing, which happens when two different places appear similar. Since loop closure detection has generally no information about the global graph structure when computing the covariance, it relies only on the similarity of two images or scans. This type of outliers would have a low covariance matrix, encouraging the optimizer to close the loop. The problem is exacerbated by the fact that perceptual aliasing tends to create large numbers of mutually-consistent outliers, which further increase the incentives for optimizer.

Numerous studies [5], [6], [7], [8], [9], [10], [11], [12] have proposed robust pose graph optimization techniques to handle spurious measurements from the *front-end*. Some works [6], [7], [10], [13] examine and remove loop closure edges according to their consistency with other measurements while some [5], [8], [12] modify the objective function to encourage intelligent behaviours from the optimizer. To specifically target the clustered nature of outliers from perceptual aliasing, Latif et al. [6] proposed to generate spatial clusters by collecting topologically related loop closure edges. Lajoie et al. [9] modelled perceptual aliasing explicitly in the objective function and solved for binary variables that accounts for the cluster nature. Despite the clear evidence that cluster-based methods handle perceptual aliasing favourably, clustering techniques remain understudied. Another difficulty of applying robust pose graph optimization techniques lies in the parameter tuning. Many existing methods share a core idea: there exists a maximum admissible error for individual cost terms during optimization. By hand-picking a threshold, cost terms that exceeded this threshold would be tuned down (referred as *M-estimation* in some literature). However, setting this threshold is far from trivial and tuning parameters in different methods determines the threshold in various indirect ways.

Contribution: In this paper, we develop a novel clustering technique that leverages the objective function change in incremental SLAM and proposed a cluster-based formulation to scale the cost terms accordingly. Our clustering technique aims at extracting both the topological structure and the local consistency information. By strategically admitting intra-consistent clusters into the optimization formulation, we can maximize the global consistency even when odometry measurements are very noisy.

*This work was supported by the CSA Grant 15FASTA17 and ARA Robotics.

[1]Fang Wu and Giovanni Beltrame are with the Department of Computer and Software Engineering, Polytechnique Montreal, Canada. fang.wu@polymtl.ca, giovanni.beltrame@polymtl.ca

II. RELATED WORK

A. Robust Pose Graph Optimization

Many researchers have attempted to achieve robust pose graph optimization (PGO), either by mitigating the effect of outliers through revising the objective function of the non-linear least squares (NLS) problem [5], [8], or by removing outliers during preprocessing of the pose graph [6]. Sunderhauf et al. [8] introduced additional variables to allow the loop closures that produce large errors to be switched off during optimization:

$$\hat{\mathbf{x}} = arg\min_{x}(\sum_{j=1}^{J} \mathbf{e}_j^T(\mathbf{x})\mathbf{W}_j^{-1}\mathbf{e}_j(\mathbf{x}) + \sum_{i=1}^{I} s_i^2\mathbf{e}_i^T(\mathbf{x})\mathbf{W}_i^{-1}\mathbf{e}_i(\mathbf{x}) \; + \; \sum_{i=1}^{I}(\lambda_i - s_i)^2\sigma_s^{-1}) \tag{1}$$

where $\mathbf{e}_j$ and $\mathbf{e}_i$ are the residual errors of odometry and loop closure edges, respectively. $\mathbf{W}_j$ is the covariance matrix of the jth odometry measurement and $\mathbf{W}_i$ of the ith loop closure measurement. The additional variable s_i scales the cost term for the ith loop closure measurement. The term $(\lambda_i - s_i)^2\sigma_s^{-1}$ works as a penalty when s_i varies from its initial value λ_i. By introducing s, the optimizer is provided with another gradient direction to minimize the objective function instead of closing the loop. And which direction to move in the solution space depends on the balance between $\mathbf{e}_i^T(\mathbf{x})\mathbf{W}_i^{-1}\mathbf{e}_i(\mathbf{x})$ and σ_s^{-1}. Agarwal et al. [5] leveraged this insight and proposed a revised form of equation (1) to explicitly scale the covariance according to the residual error. Additional variables s were eliminated, but σ_s is still used. Intuitively, σ_s should reflect the threshold on the admissible residual error. However, there is no mathematically rigorous way to set its value.

A second line of work [6], [7] exploits the probabilistic interpretation of the objective function, a χ^2 distribution under Gaussian noise assumption. Latif et al. [6] conducted χ^2 tests on spatially clustered loop closures aiming to find a set of consistent measurements. However, when the spatial clustering algorithm produces a heterogeneous cluster of loop closures, Latif's method tends to be too conservative since it would reject the whole cluster upon failing the χ^2 test. In addition, since the algorithm solves the pose graph optimization problem after including a cluster of measurements, the resultant χ^2 error could pass the χ^2 test due to high quality measurements compensating for poor quality measurements. Additional results are presented in Section IV-B.

Another line of work originated from a different perspective: many researchers attempted to avoid convergence to local minimum by performing convex relaxation on the original formulation. Carlone et al. [11] formulated the measurement selection problem as a ℓ_0 minimization problem and performed ℓ_1 relaxation to remove the non-convexity and avoid the combinatorial complexity. Semidefinite relaxation [12] has also been used to handle the formulation that models heavy-tailed measurement noise. Regarding outlier mitigation, Lajoie et al. [9] proposed a novel formulation to model perceptual aliasing. The authors introduced binary variables to switch on or off the loop closure cost terms and a correlation term was added between subsets of the binary variables, which penalizes mismatched variables from one cluster. While showing promising results with semidefinite relaxation and a convex solver, the authors acknowledged that a heterogeneous cluster of loop closures (containing both inliers and outliers) would cause the algorithm to perform poorly, and they did not explore strategies for clustering. The authors also pointed out the importance of setting the threshold on maximum admissible residual error.

B. Incremental Outlier Rejection

The works described above assume the complete graph is known when trying to decide the nature of a loop closure edge. This type of edge selection cannot be used online since future data is not available at the time of the estimation. Incremental outlier mitigation techniques such as [6] and [7] perform evaluations after adding new measurements. Both studies use a χ^2 test on a subset of the pose graph after solving the optimization problem. Bai et al. [13] proposed a metric to predict the objective function change in an incremental pose graph optimization scheme without actually solving the problem. Instead of checking for inconsistencies after adding a measurement, Bai et al. predicted the quality of the specific measurement before including it and empirically validated their metric as a good approximation in an online incremental setting. Despite showing promising results, the authors acknowledged the objective function change alone has its limitations: when the optimization problem is strongly nonlinear, the solution could change dramatically while the objective function barely changes [14]. With long trajectories and densely connected pose graphs, a naive use of this metric is not sufficiently robust (see Section IV).

III. METHODOLOGY

In this section, we present a cluster-based approach that leverages existing works [6], [8], [9], [13]. We first generate intra-consistent clusters based on their local consistency and then model this local consistence in a cluster-based robust formulation, balancing local and global consistency.

A. Clustering

We start with the spatial clustering techniques proposed by Latif et al. [6] to extract topologically related loop closures. By setting a clustering threshold γ on the pose index of loop closures, the spatial clustering technique groups loop closures according to their togological closeness. The larger γ, the farther the edges on the graph will form a cluster. This operation provides a set of spatial clusters, each containing the loop closure edges that link the same portions (index wise) of the trajectory. We use the following notation to represent the edge E between two poses, the ordered set L of edges, the set LC of loop closure edges, the set OD

of odometry edges, and the set of spatial clusters, C:

$$\begin{aligned} E &= (from, to), \quad from, to \in [1, N] \\ L &= \{E_k |\ E_k = (from_k, to_k), \quad k \in [1, K]\} \\ LC &= \{E_i |\ E_i = (from_i, to_i), \quad i \in [1, I]\} \\ OD &= \{E_j |\ E_j = (from_j, to_j), \quad j \in [1, J]\} \\ C &= \{C_m |\ C_m \subseteq LC, \quad m \in [1, M]\} \end{aligned}$$

where N is the number of poses, I is the number of loop closures, J is the number of odometry edges, K the total number of edges (i.e. $K = I + J$), and M is the number of spatial clusters. We assume L is sorted by the to index.

We then examine the incremental consistency of each cluster by computing the predicted change of the objective function Δf proposed in [13]. The estimation problem before adding a loop closure edge is considered as a primal nonlinear least-squares (NLS) problem, while the estimation after adding a loop closure edge is an augmented NLS problem. Given the solution $\mathbf{x}^\star$ of the primal NLS problem, without solving the augmented NLS problem, the change in objective function when adding the loop closure i can be computed as:

$$\Delta f = \mathbf{e}_i^T(\mathbf{x}^\star) \cdot [\mathbf{J}_i(\mathbf{x}^\star)\mathbf{M}_i(\mathbf{x}^\star)\mathbf{J}_i^T(\mathbf{x}^\star) + \mathbf{W}_i]^{-1} \cdot \mathbf{e}_i(\mathbf{x}^\star) \quad (2)$$

where $\mathbf{e}_i^T(\mathbf{x}^\star)$ refers to the residual error vector for loop closure i at the state $\mathbf{x}^\star$; $\mathbf{J}_i(\mathbf{x}^\star)$ refers to the Jacobians of loop closure measurement with respect to individual poses; $\mathbf{M}_i(\mathbf{x}^\star)$ refers to the marginal covariance matrix of the two poses constructing loop closure i; $\mathbf{W}_i$ refers to the covariance matrix of loop closure edge i. To determine the validity of a measurement according to the objective function change, we use the χ^2 difference test following [13], a variation of the χ^2 test applied to the difference of χ^2 error and the change in degrees of freedom. In the 2D case, after adding a loop closure edge, the increase in χ^2 error is Δf, and the degree of freedom (DOF) of the graph increase by 2. Then the validity of Δf can be examined with the $\chi^2_\alpha(2)$ test, where α, the confidence probability, is ususlly set to 0.95.

We note a difference between [13] and our method: Bai et al. [13] aim at either accepting or rejecting an individual edge after evaluating it on its own merits, the quality of which is determined from its consistency with the odometry edges and all the previously accepted loop closures; on the contrary, our method aims at exploiting local consistency by examining a spatially coherent cluster.

As shown in Algorithm 1, we incrementally examine the loop closures from one spatial cluster. If the loop closure passes the χ^2 difference test, indicating it is consistent with the odometry edges and the loop closure edges that have come before the current time step, it is considered as an inlier and added into the optimizer; otherwise, it is rejected and labelled as an outlier. After processing one cluster, we examine how many inliers and outliers have been produced. If there are more inliers than outliers, the inliers are returned as a *pro-odometry* cluster, with a binary quality indicator set to 1. If there are more outliers than inliers, we perform a consistency check (Algorithm 2) on the *against-odometry* cluster, consisting of all the outliers from Algorithm 1.

Algorithm 1: consistency check within the spatial cluster

Input: A spatial cluster C_m
Output: A intra-consistent cluster with its quality indicator

for E_k ***in*** L **do**
  if $E_k \in OD$ **then**
    Add E_k **into** $Optimizer$
  else
    if $E_k \in C_m$ **then**
      Optimize and Compute Δf
      if $\Delta f \leq \chi^2_{0.95}(DOF)$ **then**
        $inlierSet \leftarrow \{inlierSet, E_k\}$
        Add E_k **into** $Optimizer$
      else
        $outlierSet \leftarrow \{outlierSet, E_k\}$
if $inlierSet.size() > outlierSet.size()$ **then**
  $quality = 1$
  return $inlierSet$, $quality$
else
  return Algorithm 2($outlierSet$)

The purpose of this second check is to prevent rejecting all the loop closures that are inconsistent with odometry edges, in case of noisy odometry measurements. In a scenario where a specific cluster attempts to close long loops without information on the smaller loops that support the same inference, the second test would save it from being rejected immediately. Algorithm 2 has similar logic as Algorithm 1, except that the first loop closure is always accepted into the optimizer. This would change the system state based on the first loop closure so that outliers from Algorithm 1 could turn out to be inliers now. After processing the *against-odometry* cluster, we again examine the quantity of inliers and outliers. If there are more inliers than outliers, the inliers are returned with a quality indicator of value 0, which means although intra-consistent, these edges are not consistent with the odometry edges. Otherwise, the new outliers are processed again. By iteratively executing Algorithm 2, we isolate an intra-consistent *against-odometry* cluster. Intuitively, our clustering techniques combs through the spatial clusters and untangles the interdependence between edges by extracting only the majority set after every iteration of consistency checks. Compared with the intra-cluster consistency check techniques in [6], our method preserves more inliers as shown by experimental results in Section IV-B.

B. Cluster-based Penalty Scaling

Inspired by [8] and [9], we propose a robust optimization formulation as (3). The objective function for minimization includes four types of cost terms: odometry edges, scaled loop closure edges, prior constraints for the scale s, and clustering constraints that bind the scales of the edges belonging to similar clusters. The first cost term in the optimization, which corresponds to odometry edges, remains unchanged from the classical formulation. The following scaled loop

Algorithm 2: consistency check within the *against-odometry* cluster

Input: A set of loop closure edges C_m
Output: A intra-consistent cluster with its quality indicator

for E_k ***in*** L **do**
  if $E_k \in OD$ **then**
    Add E_k **into** $Optimizer$
  else
    if $E_k \in C_m$ **then**
      if *no loop closure in* $Optimizer$ **then**
        $inlierSet \leftarrow \{inlierSet, E_k\}$
        Add E_k **into** $Optimizer$
      else
        Optimize and Compute Δf
        if $\Delta f \leq \chi^2_{0.95}(DOF)$ **then**
          $inlierSet \leftarrow \{inlierSet, E_k\}$
          Add E_k **into** $Optimizer$
        else
          $outlierSet \leftarrow \{outlierSet, E_k\}$
if $inlierSet.size() \geq outlierSet.size()$ **then**
  $quality = 0$
  return $inlierSet$, $quality$
else
  return Algorithm 2$(outlierSet)$

closure term and prior term have the same interpretation as in [8]. The prior term accounts for the penalty of tuning downa loop closure cost term and σ_i is constant for all loop closure edges in [8], i.e. the admissible error thresholds are the same for all loop closure edges. To overcome this limitation, we scale σ_i for different loop closures depending on the quality provided by the clustering process.

$$\hat{\mathbf{x}} = arg\min_{x}\Bigg(\underbrace{\sum_{j=1}^{J}\mathbf{e}_j^T(\mathbf{x})\mathbf{W}_j^{-1}\mathbf{e}_j(\mathbf{x})}_{\text{Odometry}} + \underbrace{\sum_{i=1}^{I}s_i^2\mathbf{e}_i^T(\mathbf{x})\mathbf{W}_i^{-1}\mathbf{e}_i(\mathbf{x})}_{\text{Scaled Loop Closure}} + \underbrace{\sum_{i=1}^{I}(\lambda_i - s_i)^2\sigma_i^{-1}}_{\text{Prior}} + \underbrace{\sum_{m=1}^{M}\sum_{\substack{E_i,E_{i'}\in C_m \\ E_i\neq E_{i'}}}(s_i - s_{i'})^2\sigma_m^{-1}}_{\text{Clustering}}\Bigg) \tag{3}$$

$$\sigma_i = \begin{cases} 1^2 & if\ \ E_i \in C_m\ \ \&\ \ quality_{C_m} = 0 \\ 0.1^2 & if\ \ E_i \in C_m\ \ \&\ \ quality_{C_m} = 1 \end{cases} \tag{4}$$

As explained in previous section, two types of intra-consistent clusters are produced by Algorithms 1 and 2, the *pro-odometry* and *against-odometry* clusters. For the optimizer to favor the former, we decrease the standard deviation $\sqrt{\sigma}$ of the noise model for the prior constraint by ten times so that the penalty of tuning down these edges is increased, as in (4). The recommended value for variance σ_i in [8] is 1.

The last term in Equation (3) accounts for the clustering, which encourages the scale variables that correspond to edges within one cluster to move in the same direction in the solution space. After replacing the detailed Mahalanobis distance form with the symbol ψ_i^2 for simplicity, $\psi_i^2 = \mathbf{e}_i^T(\mathbf{x})\mathbf{W}_i^{-1}\mathbf{e}_i(\mathbf{x})$, we can express the cost terms corresponding to all loop closure edges in cluster C_m as:

$$\begin{aligned} Cost(C_m) = &\sum_{E_i\in C_m} s_i^2\psi_i^2 + \sum_{E_i\in C_m}(\lambda_i - s_i)^2\sigma_i^{-1} \\ &+ \sum_{\substack{E_i,E_{i'}\in C_m \\ E_i\neq E_{i'}}}(s_i - s_{i'})^2\sigma_m^{-1}. \end{aligned} \tag{5}$$

To determine the value of σ_m, we examine the cost terms corresponding to edge E_i from cluster C_m:

$$Cost(E_i) = s_i^2\psi_i^2 + (\lambda_i - s_i)^2\sigma_i^{-1} + \sum_{\substack{E_{i'}\in C_m \\ E_{i'}\neq E_i}}(s_i - s_{i'})^2\sigma_{ii'}^{-1} \tag{6}$$

Both the prior term and clustering term contribute to the penalty of tuning down edge E_i. We can observe that when $s_i = s_{i'}$, the clustering term is zero, and therefore it has no effect on the penalty. When $s_i \neq s_{i'}$, to let the penalty representative of the prior constraint and clustering constraint have similar weights in the formulation, we assert:

$$\sigma_{ii'} = (D_m - 1)\sigma_i \tag{7}$$

where D_m represents the number of edges in cluster C_m. When accumulating all the cost terms for cluster C_m, each clustering term will show up twice since each of them is related to two edges, leading to

$$\begin{aligned} Cost(C_m) = &\sum_{E_i\in C_m} s_i^2\psi_i^2 + \sum_{E_i\in C_m}(\lambda_i - s_i)^2\sigma_i^{-1} \\ &+2\sum_{\substack{E_i,E_{i'}\in C_m \\ E_i\neq E_{i'}}}(s_i - s_{i'})^2\sigma_{ii'}^{-1}. \end{aligned} \tag{8}$$

Comparing equation (5) and (8), we obtain:

$$2\sigma_{ii'}^{-1} = \sigma_m^{-1}. \tag{9}$$

By substituting equation (7) into (9), the variance of clustering terms can be determined as

$$\sigma_m = \frac{(D_m - 1)\sigma_i}{2}. \tag{10}$$

This way, we have formulated a cluster-based objective function that scales the penalty intelligently. It can be interpreted on a factor graph as shown in Fig. 1. Factors that link to the switchable variables vary depending on their local consistency and topological relation with other edges.

The clustering algorithms are implemented with the SLAM++ [15] optimizer due to its state-of-art marginal covariance recovery implementation, and the robust formulation is implemented within the RTABMAP [16] framework. Code is available at `https://github.com/MISTLab/Cluster-based-Penalty-Scaling`.

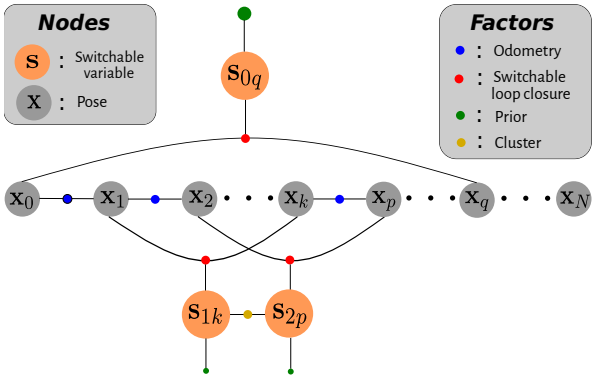


Fig. 1: Factor graph representation of the proposed formulation (factors of similar color but different sizes hint at their various weights)

IV. EXPERIMENTS

A. Datasets

2D pose datasets have been routinely used to validate outlier mitigation techniques and it is common practice to evaluate different methods by comparing their performance on the same dataset. However, to the best of our knowledge, there has not been a single "winner" on all datasets and there is no clear indication of which characteristics of the datasets affect the performance of outlier mitigation techniques. Inspired by the analysis in [17], [18], we purposefully selected datasets that vary in graph sparsity, trajectory length and odometry noise to test our approach, as listed in Table I. With the exception of the Manhattan dataset, all other datasets are real datasets. Only the Bicocca dataset has real outliers that are generated by the *front-end* while others are "clean" datasets, into which we inject synthetic outliers for our experiments. We used a script provided by Sunderhauf et al. [8] and made improvements to generate the covariance of synthetic outliers by averaging the covariance of all inliers. Besides the number of poses and loop closures, we also computed the average node degree (ND) and relative pose error (RPE). Khosoussi et al. [18] have validated that the average node degree, defined as $2K/N$, reflects the accuracy of the estimation when no outliers are present. The mean values of RPE between ground truth and the initial graph reflect the odometry noise [19].

TABLE I: Information on the datasets used

Dataset	Source/ Outliers	Poses	Loop Closures	Node Degree	RPE(m)
Bicocca	Real /Real	8357	446	2.11	0.0005
CSAIL	Real /Syn	1045	128	2.24	0.004
Manhattan	Syn /Syn	3500	2099	3.20	0.014
Intel	Real /Syn	1228	256	2.42	0.019

We compare our work with four other outlier mitigation methods: *Switchable Constraints* (SC) [8], *Dynamic Covariance Scaling* (DCS) [5], *Objective Function Change Checking* (OFCC) [13], and *Realizing, Reversing, and Recovering* (RRR) [6]. For SC, DCS and RRR, we used the original implementation provided by the authors. Since authors of OFCC did not provide their code, we implemented the method with the SLAM++ [15] optimizer.

B. Measurement Selection: Admit or Deny

We aim to select as many inliers as possible during the clustering process. Intuitively, this would later allow the optimizer to make better decisions on which edges to tune down and therefore improve accuracy. As detailed in the previous section, we leverage the spatial clustering technique from RRR. To validate the superior performance of our method in admitting more inliers, we conducted experiments by injecting different number of synthetic outliers into the CSAIL, Intel, and Manhattan datasets so that the resulting datasets have an increasing percentage of outliers: 10%, 20%, 30%, 40%, and 50%. To analyze the effect of the clustering threshold γ on performance, we tested three different configurations: $\gamma = 10$, $\gamma = 50$, $\gamma = 100$. Each configuration is repeated 10 times with randomly generated outliers and the averaged results are shown in Fig. 2.

We can make three observations: first, all methods reject more inliers as the number of outliers increase. This is expected since all methods aim to admit only consistent clusters of edges, despite using different metrics.

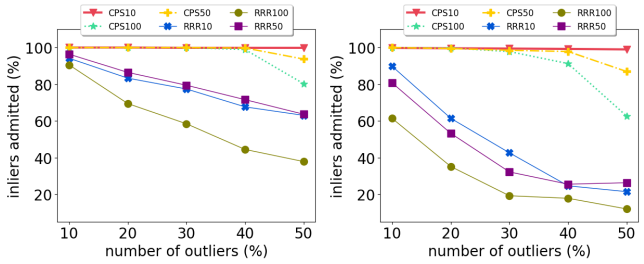
Second, regardless of the clustering threshold, RRR rejected almost half of the inliers when the number of outliers is high. Especially for the Intel and Manhattan datasets, which have noisier odometry, RRR rejected as much as 80% of the inliers when 50% outliers are present. By comparison, our method consistently admitted more inliers.

Third, considering the effect of γ, RRR ($\gamma = 10$) performed significantly better than the other two configurations on the Manhattan dataset. While RRR ($\gamma = 10$) performed slightly better than RRR ($\gamma = 50$) on the Intel dataset, the opposite can be said on CSAIL. This makes the choice of γ particularly difficult for RRR. On the contrary, CPS ($\gamma = 10$) performed consistently better than other configurations, admitting 98.9% of inliers in the worst case scenario. We use $\gamma = 10$ for all other experiments.

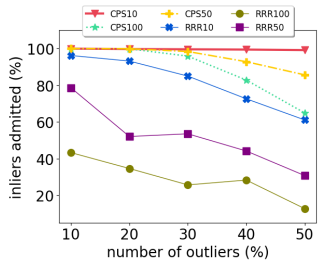
C. Accuracy

To evaluate the overall performance of outlier mitigation, we compare with three methods: SC [8] with different parameters ($\sigma_s = 1.0,\ 0.1$), DCS [5] with different parameters ($\Phi = 1,\ 5$) and OFCC [13]. We injected CSAIL, Intel and Manhattan with different numbers of synthetic outliers so that 10%, 20%, 30%, 40%, and 50% of outliers are present. We repeated each configuration 10 times and averaged the results for analysis.

We used three metrics in this section: absolute translation error (ATE), inlier acceptance rate, and outlier acceptance rate. Please note the difference between the edges admitted in the previous section and edges accepted here: after preprocessing (i.e. RRR or the clustering step of CPS), an admitted edge has its corresponding cost term in the objective function for optimization. An accepted edge has to pass the preprocessing step (in the case of SC and DCS, there is



(a) CSAIL (b) Intel



(c) Manhattan

Fig. 2: Comparison plot between CPS and RRR

no preprocessing) and correspond to a cost term with scale s larger than 0.5 at the end of optimization. A rejected edge could be eliminated during the preprocessing step, or tuned down during the optimization step. We emphasize the difference using different terminology because an edge that was first admitted during preprocessing and then tuned down during the optimization step still affects the optimization results, while an initially rejected edge has no effect on the optimization process. Our method aims to admit as many inliers as possible during the preprocessing step, even though this also means admitting more outliers. By providing a quality indicator for each cluster, the optimizer has more information to decide which edge should be tuned down.

For CSAIL, all outliers have been rejected for all trials except SC ($\sigma_s = 0.1$), which performed much worse than other methods by accepting outliers even when the number of outliers is only at 10%. ATE and inliers acceptance rate are plotted in Fig. 3. Overall, all methods performed reasonably since the highest ATE value is 0.10m. We can see that CPS manages to achieve the lowest ATE due to its capability to accept more than 99% of the inliers even when 50% of the loop closures are outliers. DCS ($\Phi = 1$) and SC ($\sigma_s = 1$) achieved slightly higher ATE because they rejected more inliers. DCS ($\Phi = 5$) has lower ATE than DCS ($\Phi = 1$) by accepting more inliers.

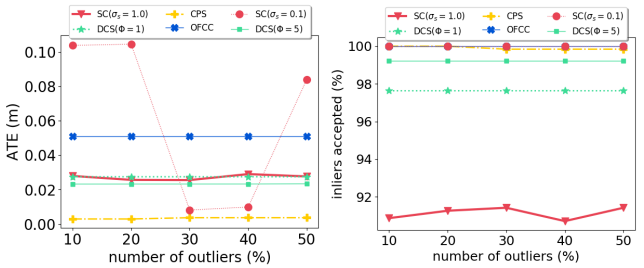
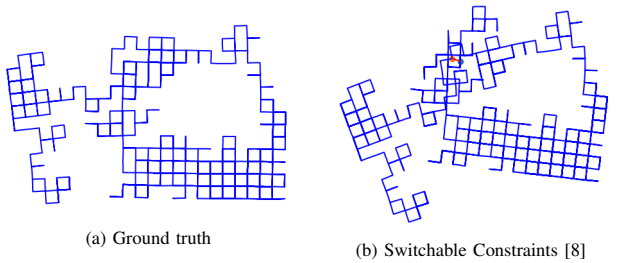
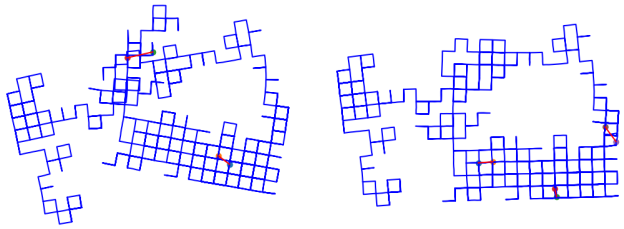


Fig. 3: Comparison of different outlier mitigation methods on CSAIL dataset

Experimental results on Manhattan are shown in Fig. 5. We can see that OFCC and SC ($\sigma_s = 0.1$) failed by accepting outliers even when only 10% of the loop closures are outliers. Both DCS ($\Phi = 1$) and DCS ($\Phi = 5$) performed reasonably when the number of outliers are low, but soon deteriorated as the number of outliers increased. SC ($\sigma_s = 1$) achived the lowest outlier acceptance rate and highest inlier acceptance rate, however, its ATE at 50% outliers is slightly larger than CPS. This is because CPS, even when accepting outliers, tends to only accept outliers that cause minimal local distortions, due to its explicit penalty scaling based on cluster structure and quality. This is also confirmed by the comparison between DCS ($\Phi = 1$) and CPS: despite having a higher outlier acceptance rate, CPS achieved a much lower ATE. Typical scenarios when DCS, SC, CPS accepted outliers are shown in Fig .4.



(a) Ground truth (b) Switchable Constraints [8]



(c) Dynamic Covariance Scaling [5] (d) Cluster-based Penalty Scaling

Fig. 4: Comparison of different outlier mitigation methods when they accepted outliers on the Manhattan dataset: blue lines indicate the optimized graph, and red lines indicate the accepted outliers

The Intel dataset has the highest odometry noise out of all the datasets we used for our experiments. The ATE, inliers acceptance rate and outliers acceptance rate are plotted in Fig. 5. We can see that SC ($\sigma_s = 1$) failed in this dataset by being too conservative, rejecting all the outliers however also rejecting too many inliers. The threshold on the admissible error is apparently too low. SC ($\sigma_s = 0.1$) performed significantly better by making good compromise, accepting more outliers but also more inliers. DCS ($\Phi = 1$) performed well when the number of outliers is relatively low, and its performance deteriorated as the number of outliers increased. DCS ($\Phi = 5$), OFCC and CPS performed well by accepting most inliers. The ATE of CPS is slightly lower than OFCC when 50% of outliers are present due to its capability of rejecting more outliers.

We also experimented with group outliers and all the results are summarized in Table II. More details on group outlier generation and analysis on the results can be found in the supplementary material, `https://mistlab.ca/papers/ClusterBasedPenaltyScaling/`.

TABLE II: Evaluation of ATE (m): we marked the proposed method in green and highest ATE for each dataset in red

	Random (mean/max)				Group (mean/max)		
	Bicocca	**Csail**	**Manhattan**	**Intel**	**Csail**	**Manhattan**	**Intel**
SC($\sigma_s = 1.0$)	2.871/-	0.027/0.030	0.245/ 7.375	8.961/ 10.224	0.028/ 0.030	0.413/ 15.317	6.967/ 10.309
SC($\sigma_s = 0.1$)	1.120/-	0.062/ 1.008	13.071/ 26.794	0.634/ 2.952	0.040/ 0.584	13.567/ 27.151	0.621/ 3.687
DCS($\Phi = 1$)	2.818/-	0.027/0.027	3.212/15.480	0.659/ 9.504	0.027/ 0.027	3.243/ 15.336	0.253/ 7.871
DCS($\Phi = 5$)	0.986/-	0.023/ 0.024	7.453/ 17.370	0.028/ 0.936	0.023/ 0.023	6.839/ 20.005	0.005/ 0.006
CPS	1.102/-	0.003/0.007	0.219/1.196	0.070/ 0.133	0.003/ 0.007	2.076/ 14.591	0.056/ 0.089
OFCC	10.531/-	0.051 /0.051	22.856 /38.010	0.099/ 0.371	0.243/ 1.827	19.524/ 35.218	0.117/ 0.964

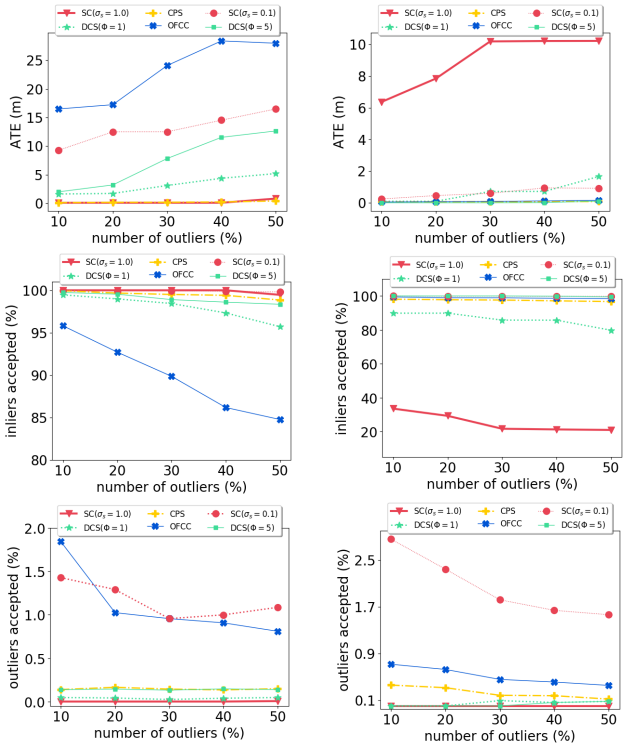
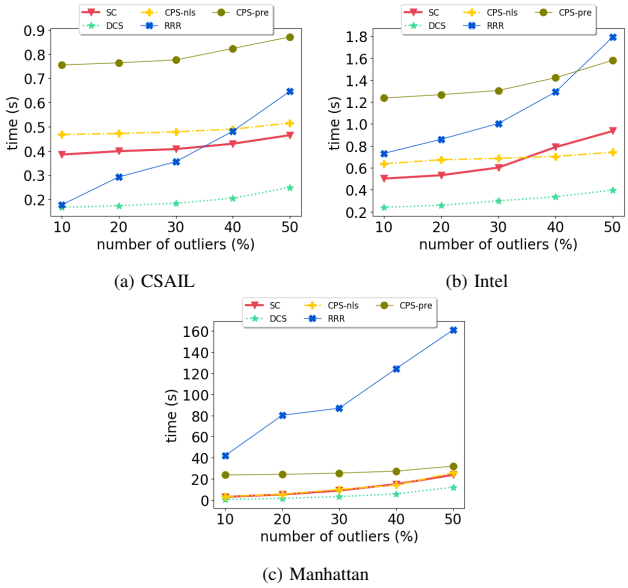


Fig. 5: Comparison of different outlier mitigation methods on the Manhattan and Intel datasets (Manhattan on the left column, Intel on the right)



(a) CSAIL (b) Intel

(c) Manhattan

Fig. 6: Comparison plots on processing time (CPS-pre refers to the preprocessing time for clustering, and CPS-nls refers to the NLS solving time)

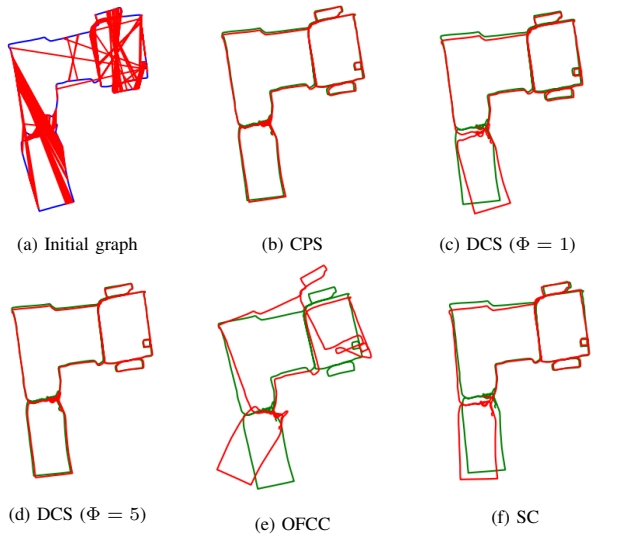
D. Timing Analysis

We performed all experiments on a laptop with an Intel 4-core processor and plotted the average processing time for different configurations in Fig. 6. We can observe that preprocessing steps, such as RRR and clustering in CPS, are generally more expensive than the NLS solving. As a result, SC and DCS are faster than RRR and CPS. Processing time of RRR increases significantly as the node degree increases, seen in the Manhattan dataset. By comparison, CPS maintained reasonable performance. Its overall timing can be further reduced by multithreading the preprocessing step since each cluster is examined independently.

E. Real Datasets with Real Outliers

For further validation, we also performed experiments on real datasets with real outliers. Latif et al.[6] provided the Bicocca dataset with varying confidence parameter for the *front-end* place recognition system, and the number of loop closures ranged from 446 to 23. Here we use the dataset with lowest confidence parameter, Bicocca-000, which contains 446 loop closures and is plotted in Fig. 7a. We can observe that outliers are present in this dataset, however we do not have ground truth on which edges are outliers, so we used only ATE for the evaluation on this dataset. The ground truth trajectory is obtained by solving a clean dataset generated with high confidence parameter for the *front-end* [6], using a non-robust solver. In Fig. 7, we compare the resulting graphs and present the mean of ATE in Table II. It is apparent that OFCC failed to reject all the outliers and the optimized graph is severely distorted. We can see that CPS and DCS with the tuning parameter Φ set to 5 performed similarly while DCS with $\Phi = 1$ (and SC) had larger errors. We used $\Phi = 5$ for DCS because its authors proposed using this value for the Bicocca Dataset. Despite producing good results, needing a specific value to produce good results for a dataset with low odometry noise is unwanted. Another interesting phenomenon is that DCS with $\Phi = 1$ works well for CSAIL and fails for Bicocca, despite both datasets having low odometry noise and similar node degrees. This further confirms the difficulty in parameter turning for DCS.

To validate our method on 3D datsets, we processed the KITTI-00 and KITTI-02 sequences using RTABMAP [16] and tuned the *front-end* parameters to produce more loop

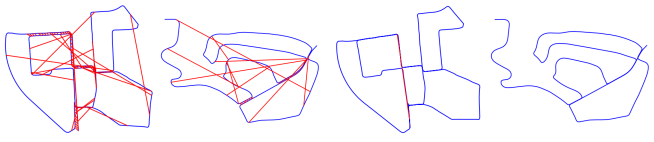


(a) Initial graph (b) CPS (c) DCS ($\Phi = 1$)

(d) DCS ($\Phi = 5$) (e) OFCC (f) SC

Fig. 7: Comparison plot between ground truth and optimized graph (in 7a, blue lines indicate the odometry while red lines indicate the loop closures; in the rest of plots, green lines indicate ground truth and red lines indicate the optimized results)

closures. As a result, we produced 101 loop closures for KITTI-00, which consists of 453 poses. For KITTI-02, we produced 36 loop closures with respect to its 464 poses. The initial graphs are plotted in Fig. 8a and Fig. 8b. After using the proposed method to mitigate the effect of outliers, we achieved the optimized graph in Figs. 8c and 8d, which show that all outliers have been rejected.



(a) 00-initial (b) 02-initial (c) 00-opti (d) 02-opti

Fig. 8: Initial and optimized graphs of the KITTI datasets (3D datasets are projected onto the horizontal plane for clarity; blue lines indicate the trajectory while red lines indicate the loop closures)

In total, we evaluate five methods on four datasets with five levels of outlier ratios. As shown in Table II, CPS performed well on all datasets. By comparison, OFCC failed on Bicocca and Manhattan dataset. SC failed on Intel dataset. DCS ($\Phi = 1$) produced much higher ATE than CPS on Bicocca and Manhattan datasets.

V. CONCLUSIONS AND FUTURE WORK

Without fine tuning the parameters, our methods perform better than other state-of-the-art outlier mitigation methods on data sets with varying odometry noise, trajectory length, and graph sparsity. For future work, we will perform in-depth analysis of the convergence behaviour of proposed method. We also plan to adapt the proposed method for online outlier mitigation. In addition, current research on the *front-end* and *back-end* is mostly in isolation: we will explore tighter integrations to study the effect of poorly estimated covariances on outlier mitigation.

ACKNOWLEDGMENT

The authors would like to thank Pierre-Yves Lajoie for his valuable insights.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[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," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3281–3288, 2011.

[3] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A General Framework for Graph Optimization," *IEEE International Conference on Robotics and Automation*, 2011.

[4] V. Ila, L. Polok, M. Solony, and P. Svoboda, "SLAM++ -A highly efficient and temporally scalable incremental SLAM framework," *International Journal of Robotics Research*, vol. 36, no. 2, pp. 210–230, 2017.

[5] P. Agarwal, G. D. Tipaldi, and L. Spinello, "Robot Map Optimization using Dynamic Covariance Scaling," *2013 IEEE International Conference onRobotics and Automation (ICRA)*, 2013.

[6] Y. Latif, C. Cadena, and J. Neira, "Robust loop closing over time for pose graph SLAM," *International Journal of Robotics Research*, vol. 32, no. 14, pp. 1611–1626, 2013.

[7] M. C. Graham, J. P. How, and D. E. Gustafson, "Robust incremental SLAM with consistency-checking," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2015-Decem, pp. 117–124, 2015.

[8] N. Sunderhauf and P. Protzel, "Switchable constraints for robust pose graph SLAM," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1879–1884, 2012.

[9] P.-Y. Lajoie, S. Hu, G. Beltrame, and L. Carlone, "Modeling Perceptual Aliasing in SLAM via Discrete-Continuous Graphical Models," pp. 1–13, 2018. [Online]. Available: http://arxiv.org/abs/1810.11692

[10] J. G. Mangelson, D. Dominic, R. M. Eustice, and R. Vasudevan, "Pairwise Consistent Measurement Set Maximization for Robust Multi-Robot Map Merging," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2916–2923, 2018.

[11] L. Carlone, A. Censi, and F. Dellaert, "Selecting good measurements via $\ell 1$ relaxation: A convex approach for robust estimation over graphs," *IEEE International Conference on Intelligent Robots and Systems*, pp. 2667–2674, 2014.

[12] L. Carlone and G. C. Calafiore, "Convex Relaxations for Pose Graph Optimization with Outliers," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1160–1167, 2018.

[13] F. Bai, T. Vidal-Calleja, S. Huang, and R. Xiong, "Predicting Objective Function Change in Pose-Graph Optimization," in *IEEE/RSJ 2018 International Conference on Intelligent Robots and Systems*, 2018.

[14] E. B. Olson and M. Kaess, "Evaluating the performance of map optimization algorithms," *RSS Workshop on Good Experimental Methodology in Robotics*, 2009.

[15] Lukas Polok, "Accelerated Sparse Matrix Operations in Nonlinear Least Squares Solvers," *Ph.D. Thesis at Faculty of Information Technology, Brno University of Technology*, 2016.

[16] M. Labbé and F. Michaud, "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[17] N. Sunderhauf and P. Protzel, "Switchable constraints vs. max-mixture models vs. RRR - A comparison of three approaches to robust pose graph SLAM," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5198–5203, 2013.

[18] K. Khosoussi, S. Huang, and G. Dissanayake, "Novel insights into the impact of graph structure on SLAM," *IEEE International Conference on Intelligent Robots and Systems*, no. Iros, pp. 2707–2714, 2014.

[19] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," *IEEE International Conference on Intelligent Robots and Systems*, pp. 573–580, 2012.