# Robust Incremental Smoothing and Mapping (riSAM)

Daniel McGann[1], John G. Rogers III[2], and Michael Kaess[1]

***Abstract*— This paper presents a method for robust optimization for online incremental Simultaneous Localization and Mapping (SLAM). Due to the NP-Hardness of data association in the presence of perceptual aliasing, tractable (approximate) approaches to data association will produce erroneous measurements. We require SLAM back-ends that can converge to accurate solutions in the presence of outlier measurements while meeting online efficiency constraints. Existing robust SLAM methods either remain sensitive to outliers, become increasingly sensitive to initialization, or fail to provide online efficiency. We present the robust incremental Smoothing and Mapping (riSAM) algorithm, a robust back-end optimizer for incremental SLAM based on Graduated Non-Convexity. We demonstrate on benchmarking datasets that our algorithm achieves online efficiency, outperforms existing online approaches, and matches or improves the performance of existing offline methods.**

## I. Introduction

Simultaneous Localization and Mapping (SLAM) is a key component that supports many complex robot behaviors. The modern approach to SLAM is to separate the task into a front-end process responsible for parsing raw sensor data into useful measurements and a back-end process responsible for estimating unknown variables from these measurements. A large variety of front-end processes have been proposed for various sensing modalities [1, Sec. B]. However, all front-ends must solve the data association problem. This is of particular importance given that the complexity of data association, in the presence of perceptual aliasing, is NP-Hard [2]. Therefore, in order to maintain tractability, we can only solve this association approximately.

Unlike front-end processes, the SLAM community has converged to a single standard approach to the back-end. This approach formulates SLAM as Maximum a Posteriori (MAP) estimation and solves for the estimate using local Nonlinear Least Squares (NLS) optimization [1], [3]. Importantly, back-end methods have been developed that can operate both *incrementally* and *online*, which is necessary for application to real-world robotics [4, Sec. 2]

Since NLS optimization is sensitive to outliers, imperfect front-end processes can cause poor performance in SLAM systems that are unable to handle these outliers. Given the importance of SLAM in robotics, techniques to handle outliers have been well studied (See Sec. II). These methods have shown promising results, however, state-of-the-art robust SLAM back-ends are predominantly batch methods. While batch methods can be used in place of incremental ones, their complexity permits online operation only for small problems. Additionally, existing incremental algorithms that can achieve online efficiency on moderate and large problems remain brittle under common conditions making them difficult to apply generally (See Sec. IV).

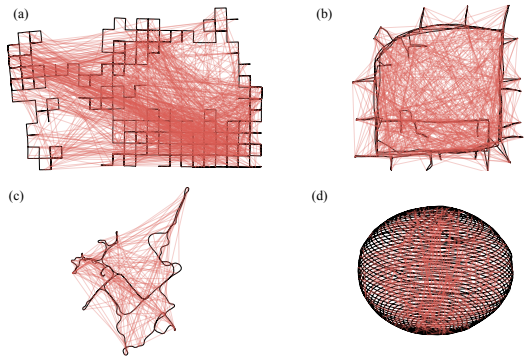



**Fig. 1:** Examples of riSAM solutions on the **(a)** Manhattan 3500 **(b)** Intel **(c)** CSAIL and **(d)** Sphere 2500 datasets corrupted with outlier loop closures (shown in red).

In this paper, we present a novel incremental algorithm for robust SLAM that achieves performance equal to or better than that of state-of-the-art batch methods while maintaining online efficiency. As part of this algorithm we develop (1) an efficient form of Graduated Non-Convexity [5], (2) the Scale Invariant Graduated kernel which provides improved performance and efficiency, and (3) a novel trust region optimization algorithm for incremental graduated optimization. Finally, we evaluate our algorithm under a variety of conditions on benchmark datasets demonstrating its ability to outperform existing methods.

## II. Related Work

Approaches to robust SLAM can be broadly classified as either maximum consensus or robust estimation methods.

Maximum consensus methods attempt to sanitize incoming measurements by identifying the largest set that are jointly consistent. They then apply existing non-robust optimization techniques assuming only inlier measurements remain. While it is possible to exactly compute the the maximally consistent set, doing so requires exponential time. Exact methods like Joint Compatibility Branch and Bound (JCBB) show that algorithms can often do better than exponential time, but are still too inefficient for real-time constraints [6]. For tractability, other methods seek to approximate the maximum consensus set. Pairwise Consistency Maximization (PCM) does so by approximating joint consistency with pairwise consistency and using approximate Max-Clique algorithms to compute the maximum pairwise consistent set [7]. Realizing, Reversing, and Recovering (RRR) and its incremental derivative (iRRR) approximate the maximum consensus set using repeated optimization of temporally partitioned sets of measurements [8], [9]. However, despite these approximations, neither method claims real-time efficiency and as we will see in Sec. IV these approximations may not always result in good performance.

This work was partially supported by DEVCOM Army Research Laboratory Distributed, Collaborative, Intelligent Systems and Technology Collaborative Research Alliance (DCIST-CRA) W911NF-17-2-0181 and by NSF grant IIS-2008279.

[1] D. McGann and M. Kaess are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. {danmcgann, kaess}@cmu.edu

[2] J. Rogers is with the DEVCOM Army Research Laboratory, Adelphi, MD, USA. john.g.rogers59.civ@army.mil

Robust estimation methods attempt to construct the SLAM problem such that the effects of outlier measurements on the final solution are mitigated. The oldest and most common form of robust estimation is the use of M-Estimators [10]. This method modifies the NLS cost function with a robust kernel $\rho$ that applies a sub-quadratic cost to outliers. This cost function can be seen in Eq. (1):

$$\sum \rho\left(r\left(z_i, f\left(x_i\right)\right)\right) \tag{1}$$

where the function $r$ computes the residual between the $i^{th}$ measurement ($z_i$) and its predicted value ($f(x_i)$) given a variable estimate ($x_i$). This robust cost function is solvable with Iteratively Reweighted Least Squares (IRLS) when the robust kernel meets known criteria [11]. A large number of different robust kernels have been proposed [10], [12], [13]. Unfortunately, M-Estimator optimization is very sensitive to the choice of $\rho$. Kernels with infinite growth (e.g. Huber) allow outliers to have significant influence on the solution. Conversely, kernels that are asymptotically constant (e.g. Geman-McClure) introduce local optima to the cost function and cause sensitivity to initialization [14, A6.8]. Methods have been proposed to better optimize the cost function when using asymptotically constant kernels. Graduated Non-Convexity (GNC) applies ideas from continuation to better optimize this highly non-convex cost function [5]. Empirical evidence shows that using GNC significantly reduces sensitivity to initialization that such kernels typically exhibit.

Another robust estimation approach is to augment the SLAM problem with additional variables that, during optimization, classify measurements and reduce the influence of outliers. Switchable Constraints adds continuous variables for each measurement that softly classify measurements as inliers or outliers [15]. Interestingly, this approach has been shown to be equivalent to the use of M-Estimators when the switch variables are conditionally independent [16], [5]. Thus, we expect it to exhibit the same behavior as M-Estimators. In a similar manner, one can augment the graph with discrete variables to classify measurements and solve the mixed problem using alternating optimization [17]. Similar to the continuous case, when these variables are conditionally independent, this is equivalent to the use of a robust kernel (a Max-Mixture Kernel [13] in the case of the approach presented by Doherty et al. [17]).

Variable augmentation can also allow for more complex modeling of correlations between measurements. However, solving the resulting graphical models can be difficult due to their hybrid nature or decreased sparsity. Prior works have used convex relaxations to find solutions and even provide correctness guarantees [2]. Unfortunately, these methods require significant computational time and cannot meet online constraints. Another alternative model is employed by the Adaptive Kernel method which adds a single continuous variable to the optimization [18]. This variable optimizes the shape of the robust kernel used for all measurements to avoid the sensitivity derived from hand selecting a kernel. However, after the kernel shape converges the method recovers that of a standard M-Estimator resulting, once again, in sensitivity to either outliers or initialization.

Robust estimation methods are generally compatible with any NLS solver including existing fast and efficient incremental SLAM solvers like iSAM2 [4], [19]. Therefore, unlike maximum consensus methods, some robust estimation methods can achieve online efficiency for moderate to large problems. The exceptions to this rule are approaches that solve via convex relaxations or GNC. These methods run offline and in batch [2], [5].

Given currently published results and further supported in Sec IV, it appears that out of all robust SLAM methods GNC can provide the best results across general SLAM scenarios. We pursue this approach to robust SLAM and introduce an incremental version of GNC that is able to match the performance of its batch variant, outperform alternative methods, and achieve online efficiency.

## III. Methodology

In this section we present the *robust, incremental Smoothing and Mapping* (riSAM) algorithm. We first present the individual building blocks of riSAM before summarizing the algorithm in full.

### A. Graduated Non-Convexity (GNC)

GNC is a technique that seeks to mitigate the difficulties observed when optimizing the robust cost function (Eq. (1)) [5]. It does so by first convexifying the problem and then solving a series of progressively less convex instances. GNC uses the solutions from more convex problems to initialize the less convex versions. The degree of convexity is changed by a control parameter $\mu$ which affects the shape of a graduated robust kernel $\rho(r;\mu)$. We refer to this general process as "graduation" though it is also known in optimization literature as "continuation" and is frequently employed to optimize difficult problems [20].

We can understand how graduation helps solve the robust SLAM problem by relating it to the behaviors of different robust kernels discussed in Sec. II. For a *linear problem*, we know that infinite-growth kernels will result in cost functions with a single well defined optimum that is skewed by outliers. Inversely, asymptotically constant kernels result in a cost function with many local optima, but where the global optimum well approximates the outlier-free case. Using GNC we better initialize the non-convex versions of the problem reducing the likelihood that we converge to a local-optima. While this is evident for a linear problem (Fig 2), we find the same behavior even in nonlinear problems like SLAM.

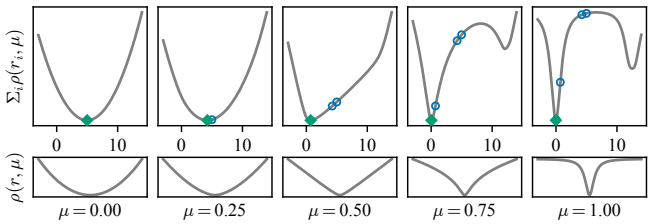



**Fig. 2:** Example of GNC solving a linear problem with outliers $\sum_i \rho(||z_i||_2^2, \mu)\ \forall\ z_i \in \{0.1, -0.05, -0.1, 12, 13\}$. The top row depicts the cost function while the bottom row depicts the shape of the robust kernel for each $\mu$. Diamonds (◆) mark the global optima for each cost function and circles (○) indicate the initialization history. Initialization from solving the convexified problem (left) ensure that we converge to the global optima for the non-convex problem (right).

### B. Efficient GNC

GNC provides an effective method to optimize the robust cost function at a computational cost. GNC, as presented by Yang et al. [5], requires we solve the global optima of the cost function for each value of $\mu$. For SLAM, this requires an iterative optimization routine, making the process inefficient. However, for all values of $\mu$ except the final, we need iterate only until the variable estimates lie within the basin of convergence of the cost function defined by the next $\mu$ value. While, it is impractical to evaluate when this condition is met, full convergence is likely unnecessary. For efficiency, we propose using only a single update per control parameter value (Alg. 1). Experimental results (Sec. IV) indicate that for typical SLAM problems this approximation is sufficient.

**Algorithm 1** Efficient Graduated Non-Convexity

1: In: Nonlinear Problem $\mathcal{P}$, Initial Variable Estimate $\mathcal{X}_0$, Control Parameter Bounds $\mu_{init}$ and $\mu_{final}$
2: Out: Final Variable Estimate $\mathcal{X}_{final}$
3: $\mu_0 \leftarrow \mu_{init}$
4: **while** Not IsConverged($\mu_i, \mu_{final}$) **do**
5: $\quad \mathcal{L} \leftarrow$ Linearize($\mathcal{P}, \mathcal{X}_i, \mu_i$)
6: $\quad \mathcal{R}, b \leftarrow$ Solve($\mathcal{L}$)
7: $\quad \Delta \leftarrow$ UpdateStepAlgorithm($\mathcal{R}, b$)
8: $\quad \mathcal{X}_{i+1} \leftarrow \mathcal{X}_i \oplus \Delta$
9: $\quad \mu_{i+1} \leftarrow$ Update($\mu_i$)
10: **end while**

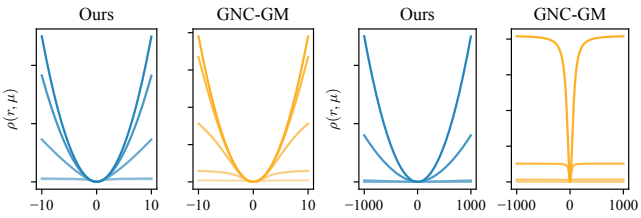
### C. The Scale Invariant Graduated (SIG) Kernel

The authors of GNC propose two graduated kernels based on the Geman-McClure (GM) [5, Eq. 4] and the Truncated Least Squares (TLS) [5, Eq. 6] kernels. Both kernels transition between a quadratic kernel and their respective basis kernel for $\mu \in [0, \infty)$. Thus, for these kernels, it takes theoretically infinite steps to transition between the convex and non-convex cases. In practice, a range can be constructed to admit a finite number of steps for $\mu$ to converge from $\mu_{init}$ to $\mu_{final}$ [5, Remark. 5]. However, the number of steps is dependent on the largest measurement residual. This would result in an inconsistent number of iterations within Alg. 1. Also, such a construction admits theoretically incorrect behavior as the convexity structure of GNC-GM and GNC-TLS will not be consistent across the residual domain for any value of $\mu$. Therefore, using these kernels allows variable estimates to jump into non-convex regions of the cost function even for iterations of GNC in which we expect the problem to be fully convex.

For online efficiency, we require a kernel that admits a known constant number of GNC iterations. Further, we desire a kernel for which convexity is known to be invariant to the scale of the residual domain. Towards this goal, we propose the Scale Invariant Graduated (SIG) kernel (Eq. (2)).

$$\rho_{\text{SIG}}(r; \mu) = \frac{1}{2} \frac{c^2 r^2}{c^2 + (r^2)^\mu} \tag{2}$$

Based on the Geman-McClure kernel, the SIG kernel is quadratic when $\mu = 0$ and recovers the Geman-McClure kernel when $\mu = 1$. The shape parameter $c$ defines the width of quadratic support of the basis kernel. The finite range of $\mu$ enables us to perform a known constant number of iterations in Alg.1. Additionally, the convexity structure of this kernel is invariant to the scale of the domain and is known for ranges of $\mu$ as shown in Fig. 3.




**(a)** Small Scale **(b)** Large Scale

**Fig. 3:** The SIG kernel compared to the GNC-GM kernel [5] at two different scales. While the GNC-GM kernel appears to transition from convex to non-convex at the smaller scale (3a) the kernel is not actually convex across the entire domain (3b). Our kernel is convex over a scale invariant domain for $\mu \geq 0.5$ and non-convex for $\mu < 0.5$.

**Remark 1** (SIG Kernel Impl. Details)**:** *We update $\mu$ according to $\mu_{i+1} = \min\left(1.0, \mu_i + 1.2(\mu_i - \mu_{init} + 0.1)\right)$. This update rule was found empirically using the intuition that we desire multiple convex steps and multiple non-convex steps before convergence. The shape of the kernel for these values can be seen in Fig. 3. We select $c = 3$ as we expect inliers fall within $3\sigma$ of the mean.*

### D. Graduated iSAM2

While Alg. 1 and Eq. (2) reduce the computational cost of GNC, it is still a batch process. We, however, can incrementalize Alg. 1 as the algorithm is agnostic to the process used in Steps 5 and 6 to linearize and solve the problem. We can therefore use existing incremental SLAM solvers in conjunction with our Efficient GNC algorithm.

Specifically, we adapt the iSAM2 algorithm for this purpose [4]. We modify iSAM2's update procedure [4, Alg. 6] to linearize according to the SIG kernel and GNC control parameter $\mu$. For correctness, we also modify the algorithm to track all variables affected by factors that are relinearized with $\mu \neq \mu_{final}$ (See Remark 3). The modified version can be found in Alg. 2 with modified steps marked in bold.

**Algorithm 2** Graduated iSAM2 Update

1: In: Bayes Tree $\mathcal{T}$ , Nonlinear Factors $\mathcal{F}$, Affected Variables $\mathcal{J}$, Control Parameter $\mu$
2: Out: Modified Bayes Tree $\mathcal{T}'$, Convex Variables $\mathcal{C}$
3: Remove top of Bayes tree:
$\quad$(a) For each affected variable in $\mathcal{J}$ remove the corresponding clique and all parents up to the root.
$\quad$(b) Store orphaned sub-trees $\mathcal{T}_{orph}$ of removed cliques.
4: **Relinearize all factors required to recreate top, using $\mu$ if applicable**.
5: **Store variables of factors relinearized with $\mu \neq \mu_{final}$ in $\mathcal{C}$**.
6: Add cached linear factors from orphans $\mathcal{T}_{orph}$.
7: Re-order variables, see Section 3.4.
8: Eliminate factor graph [4, Alg. 2], create new Bayes tree [4, Alg. 3].
9: Insert the orphans $\mathcal{T}_{orph}$ back into the new Bayes Tree.

### E. Dog-Leg Line Search

Within Alg. 1 we compute updates to the NLS problem. Existing SLAM solvers often use the Gauss-Newton algorithm to compute updates. However, we find that using Gauss-Newton on datasets corrupted with outliers causes large updates and results in numerical overflow. The same behavior is observed in Alg. 1 during interations where $\rho(r; \mu)$ is convex.

To handle such behavior we turn to trust region optimization algorithms. The only trust region algorithm that

has been found to be incrementalizable is Powell's Dog-Leg, whose incremental variant is the RISE algorithm [21]. RISE, however, is incompatible with our graduated approach. Firstly, changes in $\mu$ mean that the trust region is not correlated between iterations as expected by RISE. Secondly, RISE's step-acceptance criterion relies on comparing the cost decrease between the nonlinear and linearized problems. We find this criteria is biased to accept steps where $\rho(r;\mu)$ is convex and reject steps where $\rho(r;\mu)$ is non-convex.

Given that no viable method exists, we developed a novel trust region algorithm tailored to GNC that, like RISE, is based on Powell's Dog-Leg. It is structured as a standard line-search algorithm satisfying the Wolfe conditions [22] with three modifications. Firstly, we set a maximum step size $\alpha_{max}$ to avoid numerically unstable steps. Secondly, we search along the Dog-Leg arc rather than along a single direction to gain the advantages of combining the Gauss-Newton and gradient directions. Thirdly, we always accept the first step regardless of whether it satisfies convergence conditions to encourage the exploration of the cost function.

**Algorithm 3** Dog-Leg Line Search

In: Bayes Tree $\mathcal{T}$, Gauss-Newton Step $\Delta_{GN}$, Gradient Step $\Delta_G$
Out: Update Step $\Delta$
$\alpha_0 = \min\left(\alpha_{min}, |\Delta_{GN}|\right)$ ▷ Initial trust region size.
$\alpha_f = \min\left(\alpha_{max}, |\Delta_{GN}|\right)$ ▷ Maximum trust region size.
$\Delta \leftarrow$ ComputeDogLegPoint($\mathcal{T}, \Delta_{GN}, \Delta_G, \alpha_0$) ▷ [21, Alg. 3]
**while** $\alpha_{i+1} =$ Update($\alpha_i$) $\leq \alpha_f$ and Wolfe cond. are not satisfied **do**
  $\Delta_{test} \leftarrow$ ComputeDogLegPoint($\mathcal{T}, \Delta_{GN}, \Delta_G, \alpha_i$)
  $\Delta \leftarrow \Delta_{test}$ if Wolfe cond. are satisfied.
**end while**

**Remark 2** (Dog-Leg Line Search Impl. Details)**:** *To allow for convergence, the initial step size and max step size are upper-bounded by the magnitude of the Gauss-Newton step. $\alpha_{min}$ is selected $\approx 1$. $\alpha_{max}$ is selected based on the scale of the robot's operational environment (i.e. $O(10^2)$ for building sized environment). Update($\alpha$) is selected as $\alpha_{i+1} = 1.5\alpha_i$. Finally, the user must supply a sufficient decrease coefficient for checking the Wolfe conditions.*

### F. riSAM

Putting together Efficient GNC (Alg. 1), the SIG kernel (Eq. (2)), our graduated iSAM2 update (Alg. 2), and the Dog-Leg Line Search Algorithm (Alg. 3) we create the robust incremental Smoothing and Mapping (riSAM) algorithm. We summarize the algorithm in Alg. 4

**Algorithm 4** riSAM

In: Bayes Tree $\mathcal{T}$, Variable Estimate $\mathcal{X}_0$, Nonlinear Factors $\mathcal{F}$, Affected Variables $\mathcal{J}$
Out: Modified Bayes Tree $\mathcal{T}'$, Updated Variable Estimate $\mathcal{X}'$
$\mu_0 \leftarrow \mu_{init}$ ▷ $\mu_{init} = 0$ Sec. III-C
**while** not IsConverged($\mu_i, \mu_{final}$) **do** ▷ $\mu_{final} = 1$ Sec. III-C
  $\mathcal{T}, \mathcal{C} \leftarrow$ UpdateBayesTree($\mathcal{T}, \mathcal{F}, \mathcal{J}, \mu_i$) ▷ Alg. 2, Eq. (2)
  $\Delta \leftarrow$ DogLegLineSearch($\mathcal{T}, \Delta_{GN}, \Delta_G$) ▷ Alg. 3
  $\mathcal{X}_{i+1} \leftarrow \mathcal{X}_i \oplus \Delta$
  $\mathcal{J} \leftarrow \mathcal{C} \cup$ FluidRelinearization($\Delta$) ▷ [4, Alg. 5]
  $\mu_{i+1} \leftarrow$ UpdateMu($\mu_i$) ▷ Remark 1
**end while**

**Remark 3** (riSAM Correctness)**:** *New information is only added in the first Bayes tree update of the riSAM algorithm. To ensure that all relevant factors are included in subsequent updates to the Bayes tree and are therefore relinearized according to the updated value of $\mu$ we must track variables of these factors in Alg. 2 and mark them in subsequent iterations. Additionally, to ensure correctness we account for fluid relinearization during internal iterations of the riSAM algorithm (For fluid relinearization details see [4, Sec. 4.1]).*

While the base riSAM algorithm is a full incremental robust SLAM solver, we further improve its efficiency with some minor modifications.

*1) Known Inliers:* In-so-far we have treated all measurements identically. However, in practical applications there are likely measurements that we don't need consider as potential outliers (e.g. odometry). Within riSAM, these measurements can be given a quadratic kernel and when such a new measurement arrives we apply only a base iSAM2 update skipping the internal loop of Alg. 4 for efficiency.

*2) Adjusting Initial Control Parameters:* Large numbers of outlier measurements have the effect of reducing the sparsity of the underlying SLAM problem (e.g. Fig. 1). Given we rely on sparsity for our efficiency, this can greatly increase computational cost. However, we expect that over time we accumulate evidence that some measurements are definite outliers. We use this knowledge to mitigate the effect of outliers by maintaining a unique $\mu^i_{init}$ for each measurement $z_i$. When the algorithm converges with respect to its variable estimate we classify *strong inliers* and *strong outliers* according to user defined $\chi^2$ thresholds (we select thresholds of 0.25 and 0.9 respectively). For all strong outliers we adjust $\mu^i_{init}$ according to Remark 1, and for all strong inliers we adjust their $\mu^i_{init}$ by $\max\left(0, \mu^i_{init} - 0.1\right)$. This results in fewer variables marked convex by Alg. 2 and later iterations of Alg. 4 are, in turn, more efficient.

## IV. Experiments

In this section we provide empirical evidence to support the effectiveness and efficiency of the riSAM algorithm when handling erroneous loop-closures that we expect due to perceptual aliasing. We evaluate against state-of-the-art batch SLAM algorithms GNC [5] and PCM [7], [23] as well as against incremental robust estimation methods Switchable Constraints (SC) [15], Max-Mixture kernels [13], Gemen-McClure kernels [10], and Huber kernels [10] which use iSAM2 as their underlying solver [4]. We also perform two ablation studies to evaluate the efficacy of our Dog-Leg Line Search algorithm and the efficiency gain from $\mu_{init}$ updates.

**Remark 4** (Equivalencies)**:** *In Sec. II we note that there are equivalencies between M-Estimators and corresponding variable augmentation methods. We investigated this equivalency for continuous variables and discrete variables. We found that in the case of discrete variables the variable augmentation method [17] was empirically equivalent to the kernel approach [13]. For clarity we report only the kernel method (Max-Mixtures). Interestingly, we found that in the continuous case this equivalency does not result in identical empirical behavior. For further discussion see Sec. V.*

**Remark 5** (Prior Work Impl. Details)**:** *GNC provided by GTSAM[1] and uses default parameters. PCM provided by Kimera[2]. Switchable Constraints provided by OpenSLAM[3] Max-Mixtures were implemented by the authors and use a isotropic outlier noise model with $\sigma = 10^7$ and an outlier weight of $10^{-7}$ based on the values used in the original work [13]. To match the riSAM shape parameter (See*

[1] https://github.com/borglab/gtsam
[2] https://github.com/MIT-SPARK/Kimera-RPGO
[3] https://github.com/OpenSLAM-org/openslam_vertigo

*Remark 1), we use $c = 3$ for all kernel methods, a threshold of 3 for PCM, and a switch variable covariance of 3 for SC.*

### A. Metrics

We seek to evaluate the effectiveness of the algorithms at correctly rejecting outlier measurements and providing accurate solutions at every time-step. To quantify performance we propose the following incremental versions of Absolute Trajectory Error (ATE) [24], Precision, and Recall:

$$\text{i}\{\text{METRIC}\} = \sum_{k \in K} \left[ \frac{k}{\sum_{k \in K} k} \{\text{METRIC}\} \left( \mathcal{X}^k, \mathcal{X}^k_{pgt} \right) \right] \quad (3)$$

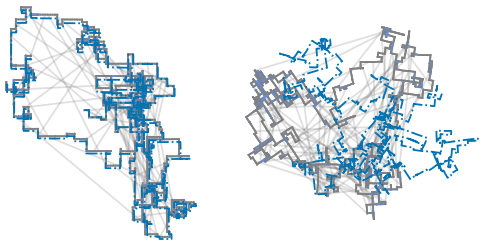
where $K$ are a set of keyframes which are taken every $m$ iterations, $\mathcal{X}^k$ is the solution computed at keyframe $k$, $\mathcal{X}^k_{pgt}$ is a pseudo-ground-truth defined as the iSAM2 solution using only inliers at keyframe $k$, and METRIC is a stand in for one of ATE, Precision, or Recall. Precision and Recall are computed from measurement classifications where, for example, "True Positive" corresponds to an inlier that is correctly classified. For methods that do not explicitly classify measurements, we define an inlier as having a $\chi^2$ error less than 0.95. This value was also used in dataset generation to ensure synthetic measurements are actual outliers.

### B. Experiments

*1) Experiment 1 - Outlier Quantity:* We evaluate the robustness of the riSAM algorithm to different quantities of outliers. To provide continuity with prior works we evaluate on the MIT CSAIL Dataset [25] and the Intel Research Lab Dataset [26]. We generated outliers as a percentile of total loop-closures. To generate outliers we sample pairs of poses and add an identity loop-closure measurement between them. For each outlier percentile we generate ten random trials and report the incremental metrics via a box-and-whisker plot. The results from this experiment can be seen in Fig. 5.

*2) Experiment 2 - Initialization:* We evaluate the robustness of riSAM to poor initialization. In the incremental SLAM problem we use the existing solution to initialize new variables. Under small amounts of noise this means new variables are initialized close to their ground truth value. However, under significant noise, variables are likely to be initialized far from their ground truth value. Therefore, by varying the amount of odometric noise we can evaluate how the different methods handle good (low noise) and poor (large noise) initialization. We generate random grid world trajectories like the examples shown in Fig. 4 using different levels of odometric orientation noise. We generate 50 random trajectories for each noise level. Outliers are added during trajectory generation with probability 0.1 at each step in which there is no inlier measurement. Results from this experiment can be found in Fig. 6.

*3) Experiment 3 - Runtime:* In this experiment we evaluate the runtime performance of the riSAM algorithm. We do so using the Intel dataset as it has documented runtime information. We generate a single trial with 30% outliers. We run each method five times on the same dataset and average results to reduce noise from CPU interrupts. We evaluate using per-iteration and total runtime. This experiment was run on a machine with an Intel Core i7-8665U processor. The results can be seen in Fig. 7.



**(a)** Small Noise: $\sigma_\theta = 0.05°$ **(b)** Large Noise: $\sigma_\theta = 2.0°$

**Fig. 4:** Example random grid world trajectories. The trajectories' pseudo-ground-truths are shown in solid gray, while the odometric initializations are shown in dashed blue. Small odometric noise results in good initialization (4a) while larger odometric noise results in poor initialization (4b).

*4) Experiment 4 - Dimensionality:* All previous experiments have used 2D SLAM problems. In this section we validate that the riSAM algorithm remains effective for 3D problems that real-world robots face. We report the performance of all methods from a single trial on the Sphere 2500 dataset [27] with 10% outliers. As we can see qualitatively in Fig. 1 and quantitatively in Tab. I riSAM is able to perform well regardless of the dimensionality of the underlying problem.

| | riSAM | GNC | PCM* | GM | Max-Mix | Huber | SC |
|---|---|---|---|---|---|---|---|
| iPrecision | 1.0 | 1.0 | 0.96 | 1.0 | 0.99 | 0.99 | 1.0 |
| iRecall | 0.99 | 0.97 | 0.99 | 0.98 | 0.02 | 0.23 | 0.32 |
| iATE (m) | 2.12 | 2.89 | 33.5 | 2.46 | 23.5 | 26.7 | 9.22 |

**TABLE I:** Evaluation of methods on the 3D Sphere 2500 dataset. riSAM generalizes across the dimensionality of the problem. *PCM results computed on the first 63% of keyframes as the method did not terminate in reasonable time.

### C. Ablation Studies

In Sec. III-E we claim that that Dog-Leg Line Search is more effective than RISE [21] in riSAM and in Sec. III-F we claim that updating the per-factor $\mu_{init}$ improves long-term efficiency. In the following studies we validate these claims.

*1) Ablation 1 - Update Algorithm:* In this study we evaluate our riSAM algorithm against a variant that uses the RISE algorithm [21]. We compare using a single trial of the Manhattan 3500 dataset with 30% outliers [28], [26]. We report the incremental metrics in Table II which show that the use of our novel Dog-Leg Line Search algorithm greatly improves riSAM's recall and trajectory error.

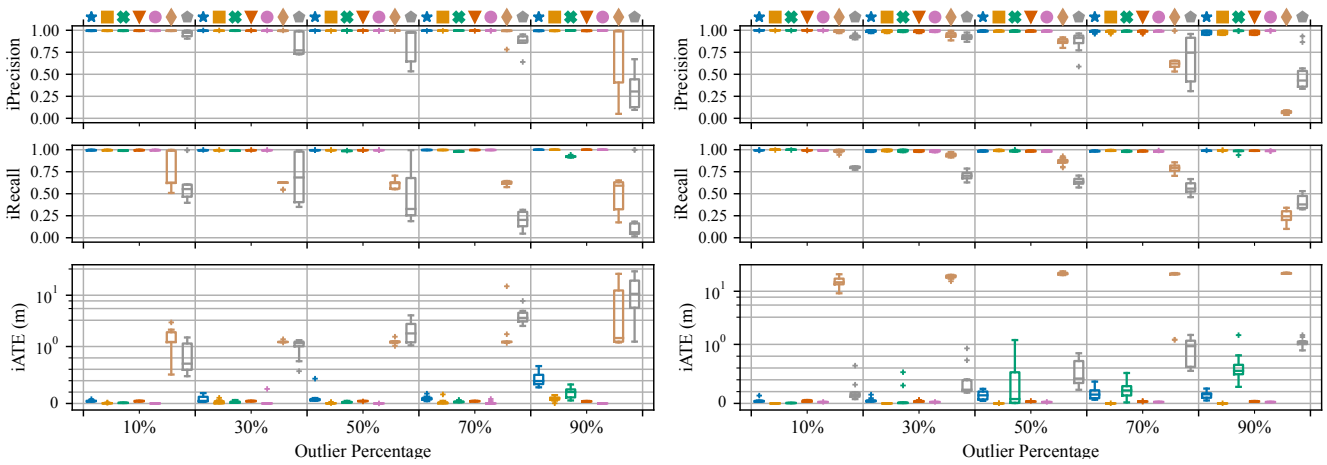
| | iPrecision | iRecall | iATE (m) |
|---|---|---|---|
| Ours (Alg.3) | 1.0 | 0.99 | 0.56 |
| RISE [21] | 0.99 | 0.87 | 11.09 |

**TABLE II:** Comparison between riSAM using RISE [21] and Dog-Leg Line Search (Alg. 3). Dog-Leg Line Search produces better results by all metrics.

*2) Ablation 2 - Initial Control Parameter Adjustment:* In this study we compare our riSAM algorithm against a variant that omits updates to $\mu^i_{init}$. We evaluate both versions on the City10k dataset [29]. Results are reported in Table III which shows that adjusting the per-factor $\mu^i_{init}$ does not degrade the performance of riSAM by any metric. Additionally, adjusting $\mu^i_{init}$ results in a **24% faster** total runtime

| | iPrecision | iRecall | iATE (m) |
|---|---|---|---|
| Adjust $\mu^i_{init}$ | 1.0 | 0.99 | 0.22 |
| Constant $\mu_{init}$ | 1.0 | 0.99 | 0.47 |

**TABLE III:** Comparison of riSAM with and without $\mu^i_{init}$ adjustment. Adjustment does not decrease performance of riSAM and provides a computational speed-up.




**(a)** Dataset: Intel **(b)** Dataset: CSAIL

**Fig. 5:** Performance of our incremental algorithm riSAM (★), prior batch methods GNC (■) and PCM (✖), and prior incremental methods Geman-McClure (▼), Max-Mixture (●), Huber (♦), and Switchable Constraints (⬟) evaluated on the Intel (5a) and CSAIL (5b) datasets corrupted with outlier measurements. Our incremental method achieves comparable performance to its batch variant and performs well even with very large numbers of outlier measurements. *Note: iATE plot uses linear scale up to* $10^0$ *and log scale above.*

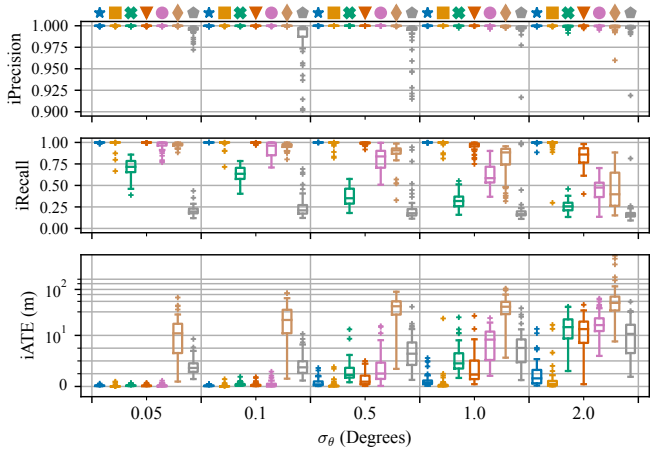



**Fig. 6:** Performance of our incremental algorithm riSAM (★), prior batch methods GNC (■) and PCM (✖), and prior incremental methods Geman-McClure (▼), Max-Mixture (●), Huber (♦), and Switchable Constraints (⬟) evaluated on random grid world trajectories generated with increasing levels of odometric noise. Our method, riSAM, outperforms all others methods as the initialization quality decreases with larger odometric noise. *Note: iATE plot has linear scale up to* $10^2$ *and log scale above.*

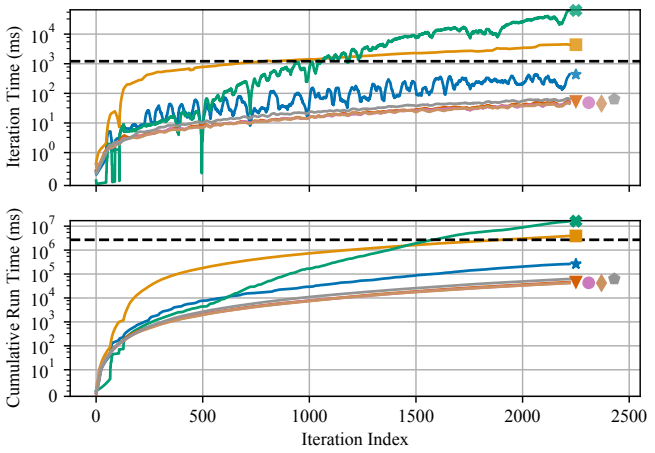



**Fig. 7:** Runtime comparison between our incremental algorithm riSAM (★), prior batch methods GNC (■) and PCM (✖), and prior incremental methods Geman-McClure (▼), Max-Mixture (●), Huber (♦), and Switchable Constraints (⬟) on the Intel dataset. The batch methods do not meet real-time requirements (shown by horizontal dashed lines) while our algorithm and prior incremental methods satisfy online constraints. *Note: Plots use a linear scale on the y-axis up to* $10^0$ *and* $10^1$ *respectively and log scale above those thresholds.*

## V. Discussion and Future Work

The results in Sec. IV demonstrate the capabilities of riSAM. In Fig. 5 and Fig. 6 we can see that riSAM is able to achieve iPrecision and iRecall statistics as good as batch GNC. riSAM often has slightly poorer iATE than GNC as intermediate results are sometimes taken when riSAM has not yet converged due to the approximation in Alg. 1. Further, in Fig. 7 we observe that, unlike batch GNC, riSAM can achieve online efficiency on large scale problems. Looking holistically at robustness to both outliers and initialization, riSAM outperforms all other methods evaluated.

Some incremental methods (Switchable Constraints and Huber) perform poorly under all conditions as they can admit arbitrary influence from outliers. While this behavior was expected from the Huber kernel it is a surprise from Switchable Constraints which we expected to match the performance of a Geman-McClure kernel. The cause of this behavior is that, unlike M-Estimators, Switchable Constraints initializes switch variables to indicate that new measurements are inliers. This initialization effectively convexifies new measurements. However, unlike our method, the convexification is only done for new factors giving them disproportionate influence and resulting in poor performance.

Other incremental methods (Max-Mixtures and Geman-McClure) and surprisingly the batch method, PCM, appear to perform very well in the first experiment (Fig. 5), when initialization is good, but break down when odometric drift results in poor initialization in the second experiment (Fig. 6). For the kernel methods this behavior is a result of the fact that they are effectively asymptotically constant kernels for the given parameters and thus sensitive to initialization. For PCM, we hypothesize that this behavior is caused by the assumptions made by the algorithm for tractability.

Currently, riSAM's largest drawback is its dependence on user defined parameters. Future work should investigate the affect of these parameters on the algorithm's behavior. Future work could also include extending riSAM for distributed applications and running field-trials for additional validation.

## REFERENCES


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

[2] P. Lajoie, S. Hu, G. Beltrame, and L. Carlone, "Modeling perceptual aliasing in SLAM via discrete–continuous graphical models," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 2, pp. 1232–1239, 2019.

[3] F. Dellaert and M. Kaess, "Factor graphs for robot perception," *Foundations and Trends in Robotics (FNT)*, vol. 6, no. 1-2, pp. 1–139, 2017.

[4] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *Intl. J. of Robotics Research (IJRR)*, vol. 31, no. 2, pp. 216–235, 2012.

[5] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 1127–1134, 2020.

[6] J. Neira and J.D., "Data association in stochastic mapping using the joint compatibility test," *IEEE Trans. on Robotics (TRO)*, vol. 17, no. 6, pp. 890–897, 2001.

[7] J. Mangelson, D. Dominic, R. Eustice, and R. Vasudevan, "Pairwise consistent measurement set maximization for robust multi-robot map merging," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Brisbane, AU, 2018, pp. 2916–2923.

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

[9] ——, "Realizing, reversing, recovering: Incremental robust loop closing over time using the iRRR algorithm," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Vilamoura, PT, 2012, pp. 4211–4217.

[10] Z. Zhang, "Parameter estimation techniques: a tutorial with application to conic fitting," *Image and Vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.

[11] K. Aftab and R. Hartley, "Convergence of iteratively re-weighted least squares to robust m-estimators," in *IEEE Winter Conference on Applications of Computer Vision*, Waikoloa, USA, 2015, pp. 480–487.

[12] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard, "Robust map optimization using dynamic covariance scaling," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Karlsruhe, DE, 2013, pp. 62–69.

[23] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Paris, FR, 2020, pp. 1689–1696.

[13] E. Olson and P. Agarwal, "Inference on networks of mixtures for robust robot mapping," *Intl. J. of Robotics Research (IJRR)*, vol. 32, no. 7, pp. 826–840, 2013.

[14] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, 2nd ed. Cambridge University Press, 2003.

[15] N. Sünderhauf and P. Protzel, "Switchable constraints for robust pose graph SLAM," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Vilamoura, PT, 2012, pp. 1879–1884.

[16] M. Black and A. Rangarajan, "On the unification of line processes, outlier rejection, and robust statistics with applications in early vision," *Intl. J. of Computer Vision*, vol. 19, no. 1, pp. 57–91, 1996.

[17] K. Doherty, Z. Lu, K. Singh, and J. Leonard, "Discrete-continuous smoothing and mapping," arXiv preprint arXiv:2204.11936v2 [cs], 2022.

[18] N. Chebrolu, T. Läbe, O. Vysotska, J. Behley, and C. Stachniss, "Adaptive robust kernels for non-linear least squares problems," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 2, pp. 2240–2247, 2021.

[19] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Technical Report, 2012.

[20] L. Carlone, "Estimation contracts for outlier-robust geometric perception," arXiv preprint arXiv:2208.10521v1 [cs, stat], 2022.

[21] D. Rosen, M. Kaess, and J. Leonard, "RISE: An incremental trust-region method for robust online sparse least-squares estimation," *IEEE Trans. on Robotics (TRO)*, vol. 30, no. 5, pp. 1091–1108, 2014.

[22] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[24] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018, pp. 7244–7251.

[25] L. Carlone, R. Aragues, J. Castellanos, and B. Bona, "A fast and accurate approximation for planar pose graph optimization," *Intl. J. of Robotics Research (IJRR)*, vol. 33, no. 7, pp. 965–987, 2014.

[26] L. Carlone and A. Censi, "From angular manifolds to the integer lattice: Guaranteed orientation estimation with application to pose graph optimization," *IEEE Trans. on Robotics (TRO)*, vol. 30, no. 2, pp. 475–492, 2014.

[27] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert, "Initialization techniques for 3D SLAM: A survey on rotation estimation and its use in pose graph optimization," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Seattle, USA, 2015, pp. 4597–4604.

[28] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Orlando, USA, 2006, pp. 2262–2269.

[29] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. on Robotics (TRO)*, vol. 24, no. 6, pp. 1365–1378, 2008.