Towards a reliable SLAM back-end

Gibson Hu, Kasra Khosoussi and Shoudong Huang

Abstract— **In the state-of-the-art approaches to SLAM, the problem is often formulated as a non-linear least squares. SLAM back-ends often employ iterative methods such as Gauss-Newton or Levenberg-Marquardt to solve that problem. In general, there is no guarantee on the global convergence of these methods. The back-end might get trapped into a local minimum or even diverge depending on how good the initial estimate is. Due to the large noise in odometry data, it is not wise to rely on dead reckoning for obtaining an initial guess, especially in long trajectories. In this paper we demonstrate how M-estimation can be used as a bootstrapping technique to obtain a reliable initial guess. We show that this initial guess is more likely to be in the basin of attraction of the global minimum than existing bootstrapping methods. As the main contribution of this paper, we present new insights about the similarities between robustness against outliers and robustness against a bad initial guess. Through simulations and experiments on real data, we substantiate the reliability of our proposed method.**

I. INTRODUCTION

Most recent works in SLAM have based their notion around a graphical representation of the problem. The nodes in the graph most commonly represent poses or features, and the edges describe a relative measurement between two nodes. The structure of the graph is often obtained by a SLAM front-end, where the nodes and edges are extracted from raw sensor data, and a back-end, where the graph configuration is estimated using the relative measurements. Measurement noise is usually assumed to be Gaussian and independent of each other. Under this assumption, the maximum likelihood estimate of the nodes is obtained through solving a non-linear least squares problem.

Since the seminal work of Lu and Milios [1] and especially in the past 8 years, various techniques have been employed to solve this non-linear least squares problem. Due to the existence of local minima, an initial estimate sufficiently close to the global minimum is crucial to almost all of these methods. The initial guess is usually obtained using the odometry data. However, due to the accumulative nature of error in dead reckoning, this initial estimate can easily lead to a local minimum or even divergence (especially true when the trajectory is long). This makes finding a good initial guess very attractive to SLAM researchers.

The term *bootstrapping* refers to finding a good initial estimate from which iterative methods such as Gauss-Newton are likely to converge to the global minimum. This problem has been acknowledged before [2]–[4]. The application of these bootstrappers is limited to pose-graphs, while feature-based SLAM (and bundle adjustment) is being used in many real-world applications. In this paper we demonstrate the advantage of using robust M-estimation techniques, in particular the Cauchy function [5], to bootstrap the Gauss-Newton algorithm under large noise. These techniques are originally designed to make the estimation process robust to outliers by damping or bounding their influence. We show how this situation is similar to the case where, in the absence of outliers (i.e., wrong data association), the optimization process is started from a bad initial estimate (e.g., noisy odometry) in SLAM. This novel insight is the main contribution of this paper.

We compare the robustness of the proposed technique to the other popular bootstrapper, Linear Approximation for Graph Optimization (LAGO) [2]. We also provide a comparison between our results and the solution obtained using the Tree-based Network Optimizer (TORO) [3] because TORO is known to be robust against bad initial estimate. Both of these methods are limited to pose-graphs, while LAGO [2] can only be applied to 2D problems. As an advantage, our method can be applied to different variants of SLAM problem (pose-graphs, feature-based and bundle adjustment) in both 2D and 3D problems.

This paper is divided into the following. Section II is related work, Section III describes the methodology. Experiments and results are provided in Section IV. More insights surrounding our approach are given in Section V. Finally Section VI is the conclusion.

II. RELATED WORK

In this section we briefly review some of the related works. First we introduce the most relevant bootstrappers in SLAM, and then discuss the application of robust M-estimation techniques in robust SLAM back-ends against wrong data association.

A. Bootstrapping Methods in SLAM

Carlone et al. [2] proposed an approximate solution for 2D pose-graphs called LAGO which can be used as a bootstrapping technique. Each pose-to-pose constraint in pose-graphs consists of a relative position part (x and y components in 2D) and a relative orientation part. The latter part is a linear function of robot orientations. Therefore by ignoring the effect of the first part of each constraint on robot orientations, a suboptimal estimate for robot orientations can be obtained through solving a linear least squares problem. After obtaining an estimate of robot orientations, observations become linear in x and y components of robot

The authors are with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia. {Gibson.Hu, Kasra.Khosoussi, Shoudong.Huang}@uts.edu.au

poses and they can be approximated by solving another linear least squares problem. Finally they perform a Gauss-Newton iteration using the computed suboptimal estimates as the initial guess. It is reasonable to consider the approximate solution of LAGO before the single Gauss-Newton iteration at the end as its proposed initial guess for Gauss-Newton iterations (see Phase 2 and Phase 3 of the algorithm in [2]).

By exploiting the structure of 2D pose-graphs, LAGO is able to produce a good initial guess. However, for the same reason, it is closely dependent on the problem formulation, and any extension to other formulations such as feature-based or 3D problems seems to be difficult. The covariance matrix of measurement (both loop closing and odometry) noise must be block diagonal. The quality of LAGO's solution depends on the ratio between the variances of the x and y parts, and the variance of the orientation part in the measurement noise: If this ratio is small then it is impossible to ignore the effect of the x and y parts of the measurements in estimating the robot orientation.

Olson et al. [6] introduced an iterative method for optimizing pose-graphs based on Stochastic Gradient Decent (SGD) which was further improved by Grisetti et al. in TORO [3]. The tree parametrization in TORO makes it more efficient computationally than Olson's SGD approach. Both of these methods are known to be robust against bad initial guess. According to [6], due to the approximations involved, their method is unable to converge to the exact maximum likelihood estimate. They suggest that their final solution can be used to bootstrap other methods. For this reason we provide a comparison between our approach and TORO.

Finally it is important to note that incremental methods such as Incremental Smoothing and Mapping (iSAM) [7], have a natural advantage over batch approaches in that the initial estimate at time t consists of the maximum likelihood estimate of the graph at time $t-1$. This makes the incremental approaches naturally more "robust" to local minima. In Section V we present an interesting interpretation of our approach from this perspective.

B. Robust Back-Ends and M-Estimation

As it was mentioned earlier, the maximum likelihood estimate under the assumption of Gaussian noise is obtained by minimizing the (weighted) sum of squared residuals. Due to the quadratic growth of least squares objective function, any outlier with a large residual has a strong impact on the final estimate. Robust M-estimators (maximum likelihood type estimators) are originally designed to make the process of finding the maximum likelihood estimate robust against the *influence* of those outliers by minimizing a different cost function with a slower growth [8]. The alternative cost functions are either constructed heuristically (e.g., using heavy-tailed probability distributions) or obtained by analysing the true distribution of the noise [9]. Robust estimators have been used in many SLAM and bundle adjustment applications [10], [11], with many approaches taking advantage of their ability to control the impact of wrong data association.

III. Robust Bootstrapping

In this section we first give a mathematical formulation of the SLAM problem. Then we discuss the motivation and the idea behind this work. Finally we show how M-estimators can be used as bootstrappers.

A. Problem Formulation

Let the directed graph $G = (V, E)$ denote the graphical representation of SLAM. Each vertex $x_i \in V$ corresponds to a robot pose or a feature position. An edge from x_i to x_j in the graph represents a relative observation z_{ij} from x_i to x_j. For the measurement between x_i and x_j we have:

$$z_{ij} = h_{ij}(x_i, x_j) + w_{ij} \tag{1}$$

where the measurement function $h_{ij}(\cdot,\cdot)$ is a non-linear function, $w_{ij} \sim \mathcal{N}(\mathbf{0}, \Omega_{ij}^{-1})$ denotes the noise variable and $\forall (i_1, j_1) \neq (i_2, j_2)$ we have: $\mathbb{E}\left[w_{i_1 j_1} w_{i_2 j_2}^\top\right] = \mathbf{0}$. Then the maximum likelihood estimate for the parameters $x \triangleq (x_1^\top, \ldots, x_n^\top)^\top$ is denoted by $x^\star$ and can be obtained by minimizing the negative log-likelihood function:

$$\begin{aligned} x^\star &= \underset{x}{\operatorname{argmin}} \sum_{(i,j)\in E} f_{ij}(x) \qquad (2) \\ &= \underset{x}{\operatorname{argmin}} \\ &\quad \sum_{(i,j)\in E} \left(z_{ij} - h_{ij}(x_i, x_j)\right)^\top \Omega_{ij} \left(z_{ij} - h_{ij}(x_i, x_j)\right) \end{aligned}$$

Given a good initial estimate $x^{(0)}$, the non-linear least squares problem (2) can be solved using iterative methods such as Gauss-Newton.

B. Motivation

As mentioned earlier, to solve any non-linear least squares problem with Gauss-Newton it is crucial to have an initial estimate that is *sufficiently close* to the global minimum. The basin of attraction of the global minimum in SLAM depends on various factors such as the noise level, the structure of the graph, etc (see [12]–[14] for an analysis on the effect of these factors on the convergence of Gauss-Newton in SLAM). Generally speaking, it is not wise to trust the odometry as an initial estimate. Even for a fixed low noise level, as the length of the traversed trajectory increases, due to the accumulation of error, the difference between the maximum likelihood estimate and the initial estimate obtained from odometry gets larger and larger.

The main idea behind our approach to bootstrapping, is to design and solve a sequence of intermediate optimization problems $\mathcal{P}_1, \ldots, \mathcal{P}_N$ such that:

(C1) The initial guess obtained from odometry is within the basin of attraction of the global minimum of $\mathcal{P}_1$.

(C2) The solution of each problem $\mathcal{P}_k$ is within the basin of attraction of the global minimum of the next problem $\mathcal{P}_{k+1}$.

(C3) The solution of the final problem $\mathcal{P}_N$ is within the basin of attraction of the global minimum of the original non-linear least squares problem (2).

Then by our definition, starting from odometry as the initial guess for $\mathcal{P}_1$ and using the solution of $\mathcal{P}_k$ as the initial estimate in $\mathcal{P}_{k+1}$ we can obtain the maximum likelihood estimate. This idea is related to the idea of Graduated Non-Convexity [10]. In general, due to the various factors involved, it is difficult to design a finite sequence $\{\mathcal{P}_k\}_{k=1}^N$ that is guaranteed to satisfy these 3 conditions. Instead we can design $\{\mathcal{P}_k\}_{k=1}^N$ using approximation and based on heuristics. In this case, it is of utmost importance to support that heuristic principle with an extensive Monte Carlo study in order to verify that the proposed sequence can handle a broad range of realistic scenarios (e.g., different noise levels, graph structure, graph type, etc).

Now we analyse a few basic choices. Let us start by considering the following subclass for designing $\{\mathcal{P}_k\}_{k=1}^N$: $\mathcal{P}_k$ is defined as the non-linear least squares corresponding to the maximum likelihood estimation of x on a (connected) spanning subgraph of G like $G_k = (V, E_k)$ (i.e., $E_k \subseteq E$):

$$x^\star_{(k)} = \underset{x}{\operatorname{argmin}} \sum_{(i,j)\in E_k} f_{ij}(x) \tag{3}$$

In this case, the difference between $\mathcal{P}_{k-1}$ and $\mathcal{P}_k$ is determined by the selection of E_k. Let us further limit this subclass such that for any k we have $E_k \subset E_{k+1}$. To satisfy (**C2**), the initial estimate at each iteration like k, i.e., $x^\star_{(k-1)}$, must be *sufficiently* close to the global minimum of $\mathcal{P}_k$, i.e., $x^\star_{(k)}$. Roughly speaking, this means that the new edges introduced in E_k (i.e., $E_k \setminus E_{k-1}$) should be *sufficiently consistent* with the edges in E_{k-1}. In other words, closing a (new) big loop in E_k might violate (**C2**) and consequently increase the risk of converging to a local minimum. On the other hand, it is obvious that ignoring measurements is not an option: in order to satisfy (**C3**) and make $x^\star_{(N)}$ close enough to $x^\star$, we need to use as many measurements as we can in the process of bootstrapping (i.e., solving $\{\mathcal{P}_k\}_{k=1}^N$). Thus intuitively, what we need is a *gradual* process of incorporating these new measurements into the optimization process so we can control their sudden *influence*.

A simple way to achieve this is to assign (additional) weights $w_{ij}^{(k)}$ to the measurements such that edges with large residuals will get smaller weights. So we can extend the subclass in (3) to:

$$x^\star_{(k)} = \underset{x}{\operatorname{argmin}} \sum_{(i,j)\in E} w_{ij}^{(k)} f_{ij}(x) \tag{4}$$

It is important to note that in this case, unlike (3), in the k^{th} intermediate optimization problem $\mathcal{P}_k$ we are using all of the edges. Nevertheless, weights $w_{ij}^{(k)}$ are chosen such that (relatively) large residuals do not have a considerable influence on the solution of $\mathcal{P}_k$.

Equation (4) is similar to applying iterative re-weighted least squares in M-estimation for controlling the *influence* of outliers on the solution. Large residuals in that problem correspond to (potential) outliers, while in our case, they correspond to measurements that are not consistent with the initial guess used in that intermediate optimization problem (e.g., closing a big loop). This new insight is our motive to use robust M-estimators in order to bootstrap the Gauss-Newton algorithm in SLAM. In the remaining parts of this section we show how M-estimators are related to the idea of (4) and its weight function.

C. M-estimation

In this paper we propose to use the final solution of an M-estimator with a re-descending influence function [8] (e.g., Cauchy function) as the initial estimate for Gauss-Newton. Using the square root of $\Omega_{ij} = \Omega_{ij}^{\frac{1}{2}}\Omega_{ij}^{\frac{1}{2}}$ we can define the normalized error vector as:

$$e_{ij} \triangleq \Omega_{ij}^{\frac{1}{2}}(z_{ij} - h_{ij}(x_i, x_j)) \tag{5}$$

Then we propose to use the following $x^{(0)}$ as the initial estimate:

$$x^{(0)} = \underset{x}{\operatorname{argmin}} \sum_{(i,j)\in E} \rho(r_{ij}) \tag{6}$$

where $r_{ij} \triangleq \|e_{ij}\|_2$ denotes the ℓ_2-norm of e_{ij} and $\rho(\cdot)$ is the cost function of the chosen M-estimator. For the Cauchy cost function we have [5]:

$$\rho(r) \triangleq \frac{c^2}{2}\log(1 + (\frac{r}{c})^2) \tag{7}$$

where c is a constant parameter. See [5] and [9] for other robust cost functions.

Depending on the choice of $\rho(\cdot)$, the optimization problem in (6) may seem difficult to solve. However it can be reformulated and solved iteratively as an iterative re-weighted least squares (IRLS). Following the notation of [5] we start by computing the gradient of the objective function in (6) w.r.t. x and setting it to zero:

$$\sum_{(i,j)\in E} \psi(r_{ij})\,\frac{\partial r_{ij}}{\partial x_t} = \mathbf{0}, \quad \text{for } t = 1,\ldots,n \tag{8}$$

where $\psi(r) \triangleq d\rho(r)/dr$ is known as the *influence* function of the M-estimator. Now by defining the *weight* function $w(r) \triangleq \psi(r)/r$ we can rewrite (8) as:

$$\sum_{(i,j)\in E} w(r_{ij})\, r_{ij}\,\frac{\partial r_{ij}}{\partial x_t} = \mathbf{0}, \quad \text{for } t = 1,\ldots,n \tag{9}$$

The LHS of (9) can be inferred as the gradient of the cost function in the k^{th} iteration of the following IRLS problem:

$$\text{minimize } \frac{1}{2}\sum_{(i,j)\in E} w(r_{ij}^{(k-1)})r_{ij}^2 \tag{10}$$

where $r_{ij}^{(k-1)}$ is the residual computed using the latest estimate. For each iteration like k we need to compute the new weights according to the residuals and solve (10) using Gauss-Newton (iteratively) with the same set of weights. The weights will be updated after the convergence of these Gauss-Newton iterations. In practice we observed that comparable results can be obtained by performing even only a single Gauss-Newton iteration for a fixed set of weights. Finally note that (10) is consistent with the underlying idea of (4).

D. Selecting the M-estimator

It is clear that the performance of our approach depends on the choice of the influence function $\psi(\cdot)$. The influence function has to exhibit the properties derived in Section III-B. Figure 1 shows the influence functions of some of the well-studied and popular M-estimators (see [5] for more details). M-estimators can be categorized based on their influence function. As you can see in Figure 1, the influence function in normal least squares is not bounded and that is why normal least squares is not robust to outliers (or in our case, bad initial value). The influence function for Huber M-estimator becomes constant beyond a threshold. Finally Cauchy and Geman-McClure have re-descending influence functions: the influence re-descend to zero for large residuals $\lim_{|r|\to\infty}\psi(r) = 0$.

Re-descending influence functions fit better with our expectations in Section III-B since they incorporate the measurements with larger residuals gradually and smoothly over intermediate iterations. Intuitively speaking, with a re-descending influence function, a measurement will have a considerable effect on the solution only when it becomes sufficiently consistent with the initial estimate. Our initial experiments also confirmed this fact. A potential problem with re-descending influence functions is that IRLS might converge to local minima. According to our Monte Carlo study in Section IV, most of the time this problem is not critical: i.e. IRLS converges either to the global minimum or to a nearby local minimum such that Gauss-Newton is able to converge to $x^\star$ using that as its initial estimate.

The descent rate is another important factor in selecting the right M-estimator. As can be seen in Figure 1, the Geman-McClure influence function descends much faster than the Cauchy influence function. If the descent phase is too quick, then loop-closing edges might not have a chance to affect the bootstrap solution and we might violate (**C3**). On the other hand, descending too slowly will reduce the robustness and might violate (**C2**).

To find the most suitable influence function we define the following family of re-descending influence functions with different rate of descent determined by α:

$$\psi_\alpha(r) \triangleq \frac{r}{(1+r^2)^\alpha} \quad , \quad w_\alpha(r) \triangleq \frac{1}{(1+r^2)^\alpha} \tag{11}$$

For $\alpha = 1$ we get Cauchy, while $\alpha = 2$ is equivalent to Geman-McClure [5]. It is interesting to see that for $\alpha = 0.5$ we get a similar influence function to Huber, and by reducing α further to zero, it becomes the normal least squares. In general, the "optimal" value for α can depend on the nature of the measurements and the noise level.

We conducted a series of Monte Carlo simulations to test the behavior of M-estimators for different values of α. In these experiments the final solution of the M-estimator is used as the initial value for the Gauss-Newton algorithm. The *success rate* in Table I shows how often the final value of the (least squares) objective function is less or equal to the best achievable value if we start from the ground truth. Table I shows the result of 100 Monte Carlo simulations of

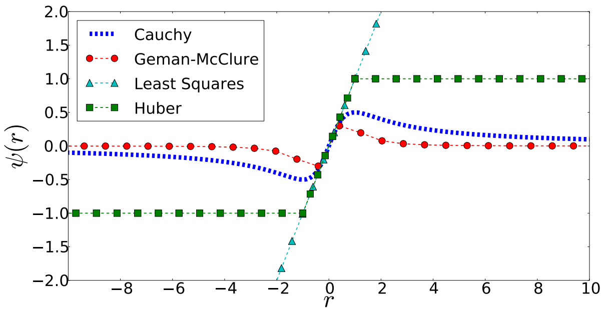


Fig. 1. Influence functions of different M-estimators

Algorithm 1: Cauchy Bootstrapper

```
**Input**: initial estimate from odometry: $x^{(0)}_{\text{odo}}$
**Output**: bootstrapper's initial estimate: $x^{(0)}$
$\alpha \longleftarrow 1$ // Cauchy M-estimator
$k \longleftarrow 1$
$x^\star_{(0)} \longleftarrow x^{(0)}_{\text{odo}}$
repeat
    foreach edge $(i,j) \in E$ do
        Compute $r^{(k-1)}_{ij}$ using $x^\star_{(k-1)}$
        $w^{(k)}_{ij} = w_\alpha(r^{(k-1)}_{ij})$
    end
    $x^\star_{(k)} = \text{GN}\left(\sum_{(i,j)\in E} w^{(k)}_{ij} f_{ij}(x)\ ,\ x^\star_{(k-1)}\right)^1$
    $k \longleftarrow k+1$
until $\|w^{(k)} - w^{(k-1)}\|_2 \leq \epsilon$
$\hat{x}^{(0)} \longleftarrow x^\star_{(k)}$
return $x^{(0)}$
```

Manhattan dataset at three noise levels. According to our results in Table I, the best performance happens close to $\alpha = 1$ (i.e., Cauchy M-estimator (7) with constant $c = 1$). Smaller α values tend to only perform poorly under larger noises and larger values of α do not have robust convergence at all. Note that for $\alpha = 0$ we get the original non-linear least squares problem. In the following sections, by robust estimation we refer to $\alpha = 1$, i.e., the Cauchy M-estimator. Our approach is summarized in Algorithm 1.

IV. Experiments and Results

A. Benchmarking

Gauss-Newton can be considered as the standard approach in SLAM back-ends, although other techniques such as gradient descent, Levenberg-Marquardt might perform better under some conditions. To find the global minimum we use Gauss-Newton algorithm initiated from the ground truth. `GT+GN` in Table II refers to this process. This is arguably the most reliable way to obtain the global minimum if the ground truth is available. Therefore if a bootstrapper were to achieve the same value or lower, its solution can be considered as the maximum likelihood estimate $x^\star$. The `+GN` postfix implies that the result of the bootstrapper has been used as the initial value of Gauss-Newton. That being said,

[1] `GN(·,·)` refers to the Gauss-Newton function which accepts two arguments: the first argument is a (nonlinear least squares) objective function and the second one is the initial estimate.

TABLE I

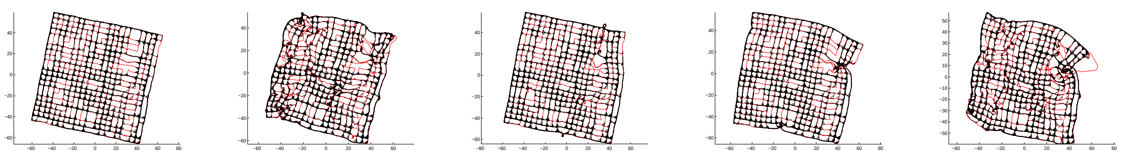
SUCCESS RATE (%) OF $\psi_\alpha(\cdot)$ IN 100 MONTE CARLO SIMULATIONS FOR DIFFERENT α VALUES IN MANHATTEN DATASET

Noise $(\sigma_x, \sigma_y, \sigma_\theta)$	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 1$	$\alpha = 1.25$	$\alpha = 1.5$	$\alpha = 1.75$	$\alpha = 2$
(0.1,0.1,0.1)	54%	94%	100%	98%	78%	14%	4%
(0.2,0.2,0.2)	16%	94%	98%	88%	22%	0%	0%
(0.3,0.3,0.3)	0%	58%	74%	60%	2%	0%	0%

TABLE II

CONVERGENCE RATE (%) FOR 50 MONTE CARLO SIMULATIONS (AVERAGE OF THE OBTAINED REDUCED χ^2)

Dataset	Noise $(\sigma_x, \sigma_y, \sigma_\theta)$	CAUCHY+GN	LAGO+GN	TORO+GN	SpanningTree+GN	Odometry+GN	GT+GN
Manhattan3500	0.05,0.05,0.05	100 (1.002)	50 (6.64)	100 (1.002)	100 (1.002)	50 (5.63)	(1.002)
	0.1,0.1,0.1	100 (1.001)	2 (1.29e+4)	100 (1.001)	90 (1.073)	2 (5.04e+5)	(1.001)
	0.2,0.2,0.2	98 (0.99)	0 (1.28e+4)	70 (1.14)	10 (2.11e+5)	0 (2.05e+3)	(0.99)
	0.3,0.3,0.3	80 (90.03)	0 (4.60e+3)	40 (2.0e+2)	0 (1.05e+4)	0 (1.07e+3)	(0.99)
	0.05,0.05,0.2	96 (1.046)	0 (1.16e+4)	74 (1.33e+2)	28 (3.21e+5)	0 (6.32e+5)	(1.00)
	0.2,0.2,0.05	100 (1.002)	46 (1.62e+3)	100 (1.002)	100 (1.002)	0 (3.81e+2)	(1.002)
	0.1,0.1,0.1 (correlated)	86 (1.15e+3)	4 (1.63e+6)	0 (3.80e+3)	90 (1.03)	0 (8.28e+6)	(1.002)
	0.2,0.2,0.2 (correlated)	78 (1.008)	0 (1.81e+4)	0 (2.15e+3)	12 (6.51e+5)	0 (1.26e+6)	(0.99)
City10000	0.05,0.05,0.05	96 (1.04)	2 (54.74)	94 (1.11)	100 (1.00)	0 (2.78e+2)	(1.00)
	0.1,0.1,0.1	100 (1.00)	0 (22.16)	82 (1.069)	94 (1.01)	0 (68.52)	(1.00)
	0.2,0.2,0.2	98 (1.00)	0 (7.50)	8 (1.23)	0 (1.29)	0 (19.3)	(1.00)
	0.3,0.3,0.3	92 (1.00)	0 (4.49)	0 (1.30)	0 (1.57)	0 (8.82)	(1.00)
	0.05,0.05,0.2	70 (1.31)	0 (16.44)	20 (8.1737)	4 (2.59)	0 (50.86)	(1.00)
	0.2,0.2,0.05	100 (1.00)	0 (96.90)	94 (1.026)	100 (1.00)	0 (361.32)	(1.00)
	0.1,0.1,0.1 (correlated)	92 (1.03)	0 (32.05)	0 (43.99)	96 (1.01)	0 (112.65)	(1.00)
	0.2,0.2,0.2 (correlated)	90 (1.00)	0 (8.32)	0 (40.95)	0 (1.43)	0 (24.60)	(1.00)



(a) Cauchy+GN (b) LAGO+GN (c) TORO+GN (d) SpanningTree+GN (e) Odometry+GN

Fig. 2. A single run at noise level (0.2,0.2,0.2) with correlated noise components for City10000

it is important to note that for large noise levels, $x^\star$ can be very different from the ground truth, and even GT+GN might fail to converge to $x^\star$. We use the so-called reduced (also known as normalized) χ^2 to verify the obtained solutions. It is well-known that for the measurement model defined in (1), if the measurement function $h_{ij}(\cdot,\cdot)$ is linear in x_i and x_j, then $f^\star \sim \chi^2_\nu$, in which $f^\star$ denotes the value of objective function (2) at $x^\star$ and $\nu \triangleq 3(m-n)$ denotes the number of degrees of freedom[2]. Therefore the expected value and variance of $f^\star$ (over all possible measurements) are equal to ν and 2ν, respectively.

It is common to extend this result to non-linear measurement functions and use ν as the *approximate* expected value of $f^\star$ by linearizing the non-linear models around $x^\star$ [15]. This approximate expected value can be used to test and validate the assumptions (e.g., distribution of the noise) and/or the model. One can compare this theoretical approximate expected value to the obtained minimum in order to verify that the obtained solution is in fact $x^\star$, if the assumptions in Section III-A can be trusted. The value of ν depends on the number of edges and vertices of the network. Therefore, to evaluate the performance across different datasets, it is more convenient to normalize $f^\star$ and report the value of $f^\star/\nu$ instead. Note that the expected value and variance of the reduced χ^2 are equal to 1 and $2/\nu$, respectively. Therefore for sufficiently large ν, we can just compute $f^\star/\nu$ for the obtained solution and see if it is close enough to 1 or not. In Table II, the average of reduced χ^2 over 50 Monte Carlo simulations is reported in parenthesis for different noise levels, datasets and algorithms. This value for GT+GN was very close to 1 in all of our simulations. Therefore we can trust the solution of GT+GN as the true maximum likelihood estimate.

[2]This value is only for 2D pose-graphs. In general $\nu = \dim(z) - \dim(x)$. Here z denotes the vector of all measurements and $\dim(\cdot)$ returns the size of the given vector.

B. Comparison Between Alternative Bootstrapping Methods

We compare our bootstrapping algorithm (Cauchy+GN) to other popular bootstrapping techniques: TORO+GN, LAGO+GN and SpanningTree+GN (a simple heuristic bootstrapper for pose-graphs [16]). For TORO and LAGO we use the code published by their original authors. All of the other experiments and simulations (including the Gauss-Newton implementation) have been done in g^{2}o [11]. The simulation datasets we have chosen are Manhattan3500 by

Olson [17] and City10000. For each dataset and noise level we have generated 50 Monte Carlo simulations. Noise levels were chosen carefully to cover all of the possible cases. In Table II there are three types of noise covariance matrices Ω_{ij}^{-1}:

1) A scalar multiple of the identity matrix.
2) Diagonal and $\sigma_x = \sigma_y \neq \sigma_\theta$.
3) Full with correlation coefficients:
   $\rho_{x,y} = \rho_{x,\theta} = \rho_{y,\theta} = 0.5$.

The units used in Table II for the standard deviation of noise are in metre (σ_x, σ_y) and radian (σ_θ). In `TORO+GN` the odometry is used as the initial value for `TORO` (only in pose-graphs can one use a spanning tree as a better initial value for both `Cauchy` and `TORO`). For each case we report the *success rate* (as defined in Section III-D) of different methods. Additionally we report (in parenthesis) the average of the obtained normalized χ^2 over Monte Carlo simulations to verify if the obtained solution is in fact $x^\star$. The results are summarised in Table II.

It is clear that `Cauchy+GN` has a very good *success rate*; in fact according to Table II, it is the only algorithm capable of handling all noise levels. Except for two cases, the average of the normalized χ^2 for `Cauchy+GN` is always close to 1, although its *success rate* might be lower than 100%. For the other methods, `SpanningTree+GN` has a high *success rate* only when the noise is small, while `TORO+GN` completely fails when the noise components are correlated. `LAGO+GN` in general, performs poorly especially if the noise components are correlated and/or σ_θ is larger than σ_x and σ_y (this behaviour was predicted in Section II-A). Finally the total failure of `Odometry+GN` underlines the importance of using a robust bootstrapper in SLAM.

It is crucial to note that in many practical scenarios, the noise components may be correlated. For instance if feature matching is used to obtain a relative pose measurement or if the motion model belongs to a non-holonomic vehicle, then the correlation will naturally exist between the x, y and θ components of the noise. Inconsistency would arise if these covariance matrices were to be approximated with a diagonal matrix. From Figure 2, it is visually clear that our results are the most consistent of all the maps. The obtained solutions by `Cauchy+GN` are the maximum likelihood estimates.

The computation time of each technique for a single run is reported in Table III. For a fair comparison of computation time between different techniques we have to make sure that all of the algorithms are converging to the true maximum likelihood estimate $x^\star$. It is very difficult to generate a series of (realistic) Monte Carlo simulations with this property. Therefore we decided to report the computation time of each method for a single run. Unlike Table II, here we do not generate our noise samples; instead we use the original (noisy) datasets shipped with g^{2}o. Empty entries in Table III denote N/A cases (e.g., number of bootstrapping iterations in `Odometry`). In Table III, "`Cauchy` (single `GN`)" refers to the case in which for each set of weights, instead of solving that intermediate non-linear least squares fully using Gauss-Newton, we only perform a single Gauss-Newton step. As it was mentioned in Section III-C, in practice the performance of "`Cauchy` (single `GN`)" is close to the original `Cauchy` method, while it is usually faster. According to Table III, our proposed method does not increase the computational complexity per iteration of a standard SLAM back-end (we just need to compute the weights); only the total number of iterations will change. The total computation time of our proposed method is comparable to that of alternative methods.

C. Real Datasets

In addition, we have tested the proposed method on real datasets. For MIT Killian Court, `Cauchy+GN` was able to achieve the maximum likelihood estimate with $f^\star = 39.6$ (Figure 3(a)), while `Odometry+GN` converges to a local minimum with objective value of 769.70. We also tested `Cauchy+GN` on feature-graphs and bundle adjustment problems. Our method is the only bootstrapper capable of solving these variants of SLAM. For the Victoria Park dataset, `Cauchy+GN` converges to the global minimum with $f^\star = 9012$ (Figure 3(b)) while `Odometry+GN` converges to a local minimum with χ^2 value of 2327481. We used the Malaga PARKING-6L dataset [18] to evaluate the performance of the proposed algorithm in bundle adjustment. We have chosen to employ parallax angle parametrization [19] to avoid potential instabilities when using Gauss-Newton. `Cauchy+GN` is able to achieve $f^\star = 14748$ (Figure 3(c)), while the χ^2 value for `VisualOdometry+GN` is 277370.

V. Discussion

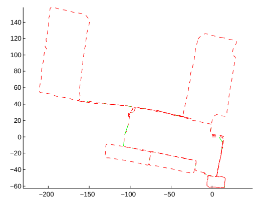
Table II clearly shows how unreliable `Odometry+GN` can be even for a low noise level. It is of utmost importance to conduct a Monte Carlo study when evaluating SLAM algorithms. Failure to do so may result in drawing wrong conclusions about the reliability of those methods. According to Table II, for the lowest noise level in Manhattan3500 dataset, `Odometry+GN` can achieve the maximum likelihood estimate in 50% of the simulations. This means a single successful realization of noise can be misleading from the overall perspective. In other words, without a proper Monte Carlo study, one is not able to take into account the failures of `Odometry+GN` in the remaining noise instances/levels and/or datasets. Therefore using a good bootstrapper is essential to the success of SLAM back-ends.

The proposed framework in Section III-B has a general form and only describes the properties of an *ideal* bootstrapper. In this paper we showed that the Cauchy M-estimator is an effective realization of this general idea. Many of the alternative algorithms can be viewed as special (or extreme) cases of our general framework. For example LAGO can be viewed as an extreme instance of this general idea in which $N = 1$ and $\mathcal{P}_1$ is a (linear) least squares problem constructed using the linear approximation method explained in Section II-A. In [12] Carlone proposes to extend LAGO by increasing the number of these intermediate steps (which is equivalent to increasing N in our framework). Incremental methods such as iSAM also fit within this framework: the

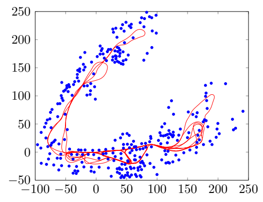
TABLE III

COMPUTATION TIME FOR A SINGLE RUN OF EACH BOOTSTRAPPER ON AN INTEL COREI5-2400 RUNNING AT 3.10GHZ

Dataset	Bootstrapper	# Bootstrapping Iterations	Bootstrapping Time (s)	# +GN Iterations	GN Time (s)	Total Time (s)
City10000	`Cauchy` (single `GN`)	11	1.620	3	0.48	2.1000
	`Cauchy`	64	9.144	3	0.48	9.6240
	`TORO`	100	11.95	3	0.48	12.4300
	`LAGO`	-	0.35	2	0.33	0.6800
	`SpanningTree`	-	≈ 0	4	0.634	0.634
	`Odometry`	-	≈ 0	7	1.069	1.069



(a) MIT Killian Court (pose-graph)



(b) Victoria Park (feature-based)

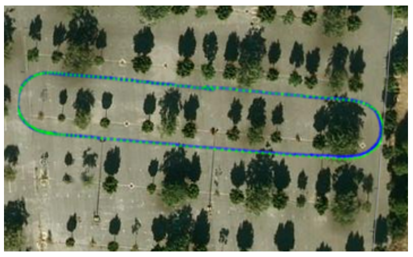

(c) Malaga (bundle adjustment): GPS (green), Our method (blue)

Fig. 3. Results of `Cauchy+GN` for different variants of SLAM

$\mathcal{P}_k$ in incremental approaches is the non-linear least squares problem that arises in the process of obtaining the maximum likelihood estimate of the graph vertices at time step k using all the available edges up to that time. Finally note that submapping approaches are also consistent with our general framework: intermediate optimization problems are the maximum likelihood estimation problems in each submap (i.e., disjoint subgraphs of G).

VI. CONCLUSION

In this paper we have demonstrated the importance of using a reliable bootstrapper in graph-based approaches to SLAM. A general framework for the *ideal* bootstrapper was developed. After discussing a number of heuristic realizations for that *ideal* framework, we illustrated the connection between this problem and robustness against outliers, and proposed to use M-estimators with re-descending influence functions as our bootstrapper. Our extensive Monte Carlo study revealed that the proposed method outperforms existing methods with a comparable computation time. Furthermore, unlike the alternative methods, the proposed algorithm is capable of handling different noise levels, graph types and formulations (e.g., feature-based, bundle adjustment, 3D, etc). In future work we plan to investigate alternative forms of the influence function. For instance a dynamic influence function may be a better realization for our ideal bootstrapper.

REFERENCES

[1] F. Lu and E. Milios, “Robot pose estimation in unknown environments by matching 2d range scans,” *Journal of intelligent & robotic systems*, vol. 18, no. 3, pp. 249–275, 1997.

[2] L. Carlone, R. Aragues, J. Castellanos, and B. Bona, “A linear approximation for graph-based simultaneous localization and mapping,” in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[3] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard, “A tree parameterization for efficiently computing maximum likelihood maps using gradient descent,” in *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[4] M. Liu, S. Huang, G. Dissanayake, and H. Wang, “A convex optimization based approach for pose slam problems,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 1898–1903.

[5] Z. Zhang, “Parameter estimation techniques: A tutorial with application to conic fitting,” *Image and vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.

[6] E. Olson, J. Leonard, and S. Teller, “Fast iterative optimization of pose graphs with poor initial estimates,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006, pp. 2262–2269.

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

[8] P. Huber, *Robust Statistics*, ser. Wiley Series in Probability and Statistics. Wiley, 2005.

[9] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[10] A. Blake and A. Zisserman, *Visual reconstruction*, 1987.

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

[12] L. Carlone, “A convergence analysis for pose graph optimization via gauss-newton methods,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

[13] S. Huang, H. Wang, U. Frese, and G. Dissanayake, “On the number of local minima to the point feature based slam problem,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2074–2079.

[14] H. Wang, G. Hu, S. Huang, and G. Dissanayake, “On the structure of nonlinearities in pose graph SLAM,” in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[15] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press, 2007.

[16] K. Konolige, G. Grisetti, R. Kummerle, W. Burgard, B. Limketkai, and R. Vincent, “Efficient sparse pose adjustment for 2d mapping,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 22–29.

[17] E. Olson, “Robust and efficient robotic mapping,” Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, June 2008.

[18] J.-L. Blanco, F.-A. Moreno, and J. Gonzalez, “A collection of outdoor robotic datasets with centimeter-accuracy ground truth,” *Autonomous Robots*, vol. 27, no. 4, pp. 327–351, 2009.

[19] L. Zhao, S. Huang, L. Yan, and G. Dissanayake, “Parallax angle parametrization for monocular slam,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3117–3124.