Initialization Techniques for 3D SLAM: a Survey on Rotation Estimation and its Use in Pose Graph Optimization

Luca Carlone, Roberto Tron, Kostas Daniilidis, and Frank Dellaert

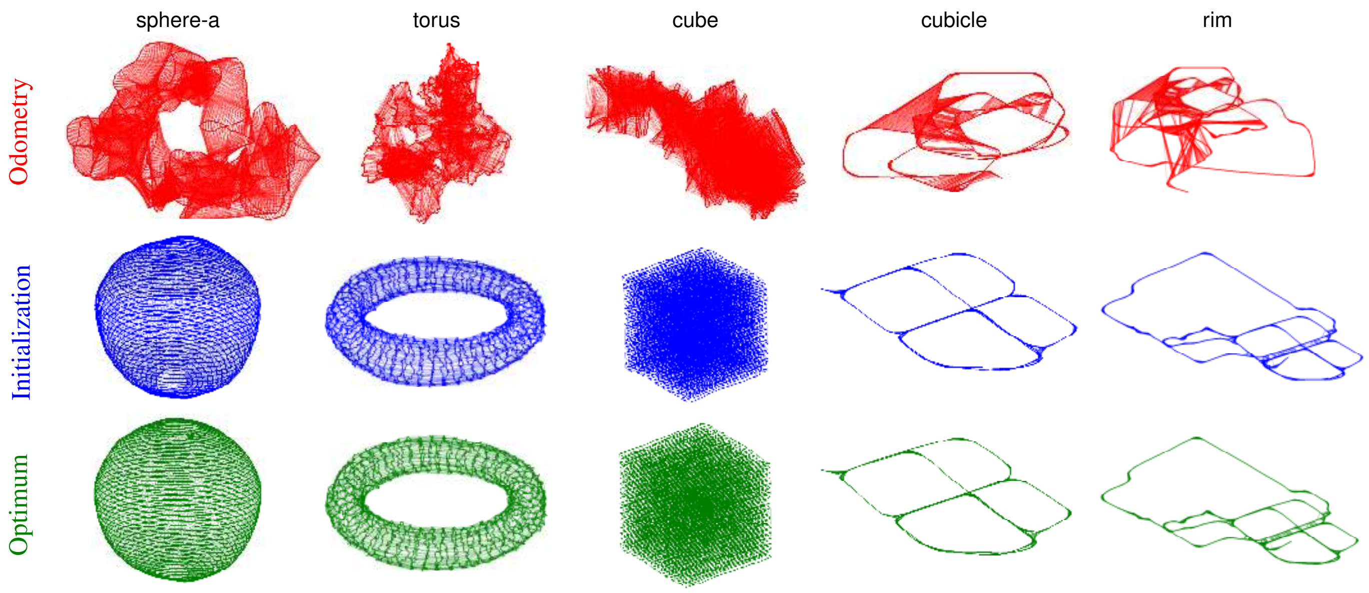


Fig. 1. State-of-the-art techniques for SLAM optimize robot trajectory via iterative methods (e.g. Gauss-Newton), starting from the odometric estimate (**red**). This strategy is doomed to fail when odometry is inaccurate. In this paper we show that if we solve for rotations first, and then use this estimate as initialization for iterative methods, we have an astonishing boost in robustness and speed: the initialization (**blue**) is visually correct and very close to the optimal solution (**green**). For 3D rotation estimation, we leverage results from related work; for instance, the initialization in the figure relies on the *chordal relaxation* from Martinec and Pajdla [1].

***Abstract*— Pose graph optimization is the non-convex optimization problem underlying pose-based Simultaneous Localization and Mapping (SLAM). If robot orientations were known, pose graph optimization would be a linear least-squares problem, whose solution can be computed efficiently and reliably. Since rotations are the actual reason why SLAM is a difficult problem, in this work we survey techniques for 3D rotation estimation. Rotation estimation has a rich history in three scientific communities: robotics, computer vision, and control theory. We review relevant contributions across these communities, assess their practical use in the SLAM domain, and benchmark their performance on representative SLAM problems (Fig. 1). We show that the use of rotation estimation to bootstrap iterative pose graph solvers entails significant boost in convergence speed and robustness.**

I. INTRODUCTION

Pose graph optimization is a state-of-the-art formulation for SLAM: robot poses are estimated by solving the non-convex optimization resulting from *maximum a-posteriori* estimation. Pose graph solvers rely on nonlinear optimization techniques (e.g., Gauss-Newton method), which iteratively refine the trajectory estimate, starting from an initial guess.

L. Carlone and F. Dellaert are with the College of Computing, Georgia Institute of Technology, USA. luca.carlone@gatech.edu, frank@cc.gatech.edu

R. Tron and K. Daniilidis are with the Department of Computer and Information Science, University of Pennsylvania, USA. {tron,kostas}@seas.upenn.edu

A good initial guess has two merits. First, initializing the estimate near the optimal solution enables fast convergence. Second, a good initialization wards off the risk of convergence to local minima, which imply large estimation errors.

Related work in robotics tackles local convergence by resorting to iterative techniques with larger basin of convergence (e.g., Levenberg-Marquardt, stochastic gradient descent [2], [3]), or exploiting robust kernels [4]. These techniques are usually slow as the improved convergence results from more conservative updates. For this reason, recent interest from the robotics community has been devoted to the computation of a good initial guess (the *initialization* problem), including contributions on 2D SLAM [5], [6], [7], visual-inertial navigation [8], [9], [10], and calibration [11].

In this work we address the initialization problem for 3D pose graph optimization. Standard approaches for batch pose graph optimization commonly use robot odometry as initial guess. As shown in this work, in most cases, this is not a convenient choice. As specified in the title, the initialization techniques we discuss in this paper leverage results on rotation estimation. The interest towards rotation estimation stems from the fact that, if robot rotations were known, pose graph optimization would be a linear least-squares problem, whose global minimizer can be computed efficiently. Recent work [5], [6], [7] showed that estimating rotations first, and then using the rotation estimate to *initialize* 2D pose graph optimization entails consistent advantages in terms of

Preprint submitted to 2015 IEEE International Conference on Robotics and Automation. Received September 30, 2014.

computation and robustness. In this work we show that this initialization is beneficial in the 3D case as well (Fig. 1). While in 2D it is possible to devise exact closed-form solutions for rotation estimation [7], no closed-form solution is known in the 3D case (beside the simple case of pose graphs with a single cycle). However, related work offers many approaches that work well in practice.

Our survey spans contributions to 3D rotation estimation across three research communities. First, rotation estimation (a.k.a. *rotation averaging*) has been studied in computer vision, where accurate camera orientation estimation is critical to solve Bundle Adjustment in Structure from Motion [12], [1], [13], [14], [15], [16], [17]. Second, rotation estimation has been investigated in the control theory community, where it finds application to vehicle coordination [18], sensor network localization and camera network calibration [19], [20], attitude synchronization [21], [22], and distributed consensus on manifold [23], [24]. Third, techniques to solve for rotations have been studied in robotics [18], [7], [25].

Since our goal is to initialize 3D SLAM, we omit planar approaches. Moreover, we exclude techniques based on discretization [26], since these techniques usually have poor scalability [27]. Finally, we purposely avoid the problem of outlier rejection, and we assume that gross outliers have been removed using suitable techniques, e.g., [28], [29].

The paper is organized as follows. Section II introduces pose graph optimization and discusses the importance of rotation estimation. Section III surveys five technique for rotation estimation. In particular, Section III-A reviews the closed-form solution for graphs with a single cycle, proposed by Sharp *et al.* in [15], and further studied by Dubbelman *et al.* [25], and Peters *et al.* [30]. Section III-B reviews the chordal relaxation of Martinec and Pajdla [1]. Section III-C reviews the quaternion relaxation of Govindu [12], and the recent analysis of Hartley *et al.* [16]. Section III-D reviews the semidefinite programming relaxation of Fredriksson and Olsson [17]. Section III-E discusses the gradient descent technique of Tron and Vidal [31]. Section IV provides numerical comparisons and elucidates on the use of these techniques in SLAM. Section V concludes the paper.

Beyond the survey contribution, we propose three minor contributions. First, we extend the technique of Section III-B to incorporate vertical direction measurements; this is important when rotation estimation can be informed by gravity measurements from an IMU. Second, we show how to exploit rotation estimation in SLAM and we compare the surveyed techniques in simulated and real robotics benchmarking problems. Third, we release an open source implementation of the best performing techniques as part of the gtsam suite [32], which is a widely used library for SLAM.

Note that computer vision literature offers an excellent survey on rotation averaging [16]. In this work, we complement [16] by covering other techniques (3 of the 5 techniques reviewed in this paper are not discussed in [16]) and by presenting numerical evaluation on robotic problems.

II. Why is Rotation Estimation Important?

In this section we remark why rotation estimation is central to pose graph optimization (Section II-A) and we introduce standard distance metrics in $\mathrm{SO}(3)$ (Section II-B).

A. Pose Graph Optimization and Rotation Initialization

Pose graph optimization estimates n robot poses from m relative pose measurements. Both robot poses and relative measurements are quantities in $\mathrm{SE}(3) \doteq \{(\boldsymbol{R},\boldsymbol{t}) : \boldsymbol{R} \in \mathrm{SO}(3), \boldsymbol{t} \in \mathbb{R}^3\}$. $\mathrm{SO}(3)$ is the set of 3D rotations which is formally defined as $\mathrm{SO}(3) \doteq \{\boldsymbol{R} \in \mathbb{R}^{3\times 3} : \boldsymbol{R}^\mathsf{T}\boldsymbol{R} = \mathbf{I}_3, \det(\boldsymbol{R}) = 1\}$, where $\mathbf{I}_3$ is the 3×3 identity matrix and $\det(\cdot)$ is the matrix determinant.

The problem can be easily visualized as a directed graph, in which nodes correspond to robot poses (to be estimated) while edges $\mathcal{E}$ correspond to relative measurements. An edge $(i,j) \in \mathcal{E}$ encodes a relative pose measurement between pose i and j. Each relative pose measurement includes a relative rotation $\boldsymbol{R}_{ij}$ and a relative translation $\boldsymbol{t}_{ij}$:

$$\boldsymbol{t}_{ij} = \boldsymbol{R}_i^\mathsf{T}(\boldsymbol{t}_j - \boldsymbol{t}_i) + \boldsymbol{t}_{ij}^\epsilon, \qquad \boldsymbol{R}_{ij} = \boldsymbol{R}_i^\mathsf{T}\boldsymbol{R}_j\boldsymbol{R}_{ij}^\epsilon, \quad (1)$$

where the pair $(\boldsymbol{R}_i,\boldsymbol{t}_i)$ defines the pose of node i (resp. j), and $\boldsymbol{t}_{ij}^\epsilon \in \mathbb{R}^3$, $\boldsymbol{R}_{ij}^\epsilon \in \mathrm{SO}(3)$ denote measurement noise.

Pose graph optimization estimates robot positions $\{\boldsymbol{t}_i\}$ and rotations $\{\boldsymbol{R}_i\}$ by solving the optimization problem

$$\min_{\substack{\{\boldsymbol{R}_i\}\in\mathrm{SO}(3)\\ \{\boldsymbol{t}_i\}\in\mathbb{R}^3}} \sum_{(i,j)\in\mathcal{E}} \mathrm{d}_{\mathbb{R}^3}\left(\boldsymbol{t}_{ij}, \boldsymbol{R}_i^\mathsf{T}(\boldsymbol{t}_j-\boldsymbol{t}_i)\right)^2 + \mathrm{d}_{\mathrm{SO}(3)}\left(\boldsymbol{R}_{ij}, \boldsymbol{R}_i^\mathsf{T}\boldsymbol{R}_j\right)^2 \quad (2)$$

where $\mathrm{d}_{\mathbb{R}^3}(\boldsymbol{t}_a,\boldsymbol{t}_b)$ denotes the Euclidean distance between two vectors $\boldsymbol{t}_a,\boldsymbol{t}_b \in \mathbb{R}^3$, while $\mathrm{d}_{\mathrm{SO}(3)}(\boldsymbol{R}_a,\boldsymbol{R}_b)$ denotes a distance metric between two rotations in $\mathrm{SO}(3)$. Roughly speaking, Problem (2) looks for the estimates $(\boldsymbol{R}_i,\boldsymbol{t}_i)$, $i = 1,\ldots,n$ that minimize the mismatch with respect to the measurements $(\boldsymbol{t}_{ij},\boldsymbol{R}_{ij})$, $\forall(i,j) \in \mathcal{E}$, according to the distance metrics $\mathrm{d}_{\mathbb{R}^3}(\cdot,\cdot)$ and $\mathrm{d}_{\mathrm{SO}(3)}(\cdot,\cdot)$.

The Euclidean distance $\mathrm{d}_{\mathbb{R}^3}(\cdot,\cdot)$ is simply:

$$\mathrm{d}_{\mathbb{R}^3}\left(\boldsymbol{t}_{ij}, \boldsymbol{R}_i^\mathsf{T}(\boldsymbol{t}_j-\boldsymbol{t}_i)\right) \doteq \left\|\boldsymbol{R}_i^\mathsf{T}(\boldsymbol{t}_j-\boldsymbol{t}_i)-\boldsymbol{t}_{ij}\right\| = \left\|\boldsymbol{t}_j-\boldsymbol{t}_i-\boldsymbol{R}_i\boldsymbol{t}_{ij}\right\|, \quad (3)$$

while different choices for the distance $\mathrm{d}_{\mathrm{SO}(3)}(\cdot,\cdot)$ are discussed in Section II-B.[1]

The following observations motivate our interest in rotation estimation. First, if rotations were known, say $\boldsymbol{R}_i = \hat{\boldsymbol{R}}_i$, $\forall i = 1\ldots,n$, Problem (2) would simplify to:

$$\min_{\{\boldsymbol{t}_i\}\in\mathbb{R}^3} \sum_{(i,j)\in\mathcal{E}} \left\|\boldsymbol{t}_j - \boldsymbol{t}_i - \hat{\boldsymbol{R}}_i\boldsymbol{t}_{ij}\right\|^2 \quad (4)$$

which is a linear least squares problem, hence easy to solve. Second, translations appear linearly in the residual errors in (3), and this implies that the initial guess for translations is irrelevant. Third, in common SLAM problems, the first term in (2) has a minor influence on the rotation estimate, and an accurate rotation initialization can be computed by minimizing only the second term:

$$\mathcal{P}: \quad \min_{\{\boldsymbol{R}_i\}\in\mathrm{SO}(3)} \sum_{(i,j)\in\mathcal{E}} \mathrm{d}_{\mathrm{SO}(3)}\left(\boldsymbol{R}_{ij}, \boldsymbol{R}_i^\mathsf{T}\boldsymbol{R}_j\right)^2 \quad (5)$$

Therefore, in this paper we propose to solve (5) to compute a good rotation estimate, and then use this rotation estimate to bootstrap (standard) iterative solvers that minimize (2).

[1]Note that we consider isotropic distances. One may use anisotropic distances (i.e., nondiagonal covariance matrices) in the nonlinear refinement that usually follows the initialization techniques discussed in this paper.

Preprint submitted to 2015 IEEE International Conference on Robotics and Automation. Received September 30, 2014.

The same insight was exploited in [6] to devise fast solutions to 2D pose graph optimization.

Towards this goal, Section III reviews existing techniques to solve or approximate the solution of Problem $\mathcal{P}$ in (5), for different choices of the distance metric.

B. Distance Metrics in SO(3)

We consider three different distance metrics between two rotations $\boldsymbol{R}_a$ and $\boldsymbol{R}_b$ in $\mathrm{SO}(3)$:

- *Angular distance*: it is the rotation angle corresponding to the relative rotation $\boldsymbol{R}_a^{\mathsf{T}}\boldsymbol{R}_b$. More formally:

$$\mathrm{d}_{\mathrm{ang}}(\boldsymbol{R}_a,\boldsymbol{R}_b) = \left\|\mathrm{Log}\left(\boldsymbol{R}_a^{\mathsf{T}}\boldsymbol{R}_b\right)\right\| = \left\|\mathrm{Log}\left(\boldsymbol{R}_b^{\mathsf{T}}\boldsymbol{R}_a\right)\right\|$$

  where $\mathrm{Log}(\boldsymbol{R})$ denotes the *logarithm map* (at the identity) for $\mathrm{SO}(3)$. In this paper, $\mathrm{Log}(\boldsymbol{R}) = \theta\boldsymbol{u}$, where $\boldsymbol{u}$ is a unit vector corresponding to the rotation axis of $\boldsymbol{R}$, and $\theta \in [0,\pi]$ is the corresponding rotation angle[2].
- *Chordal distance*: it is the Frobenius norm of $\boldsymbol{R}_a - \boldsymbol{R}_b$

$$\mathrm{d}_{\mathrm{chord}}(\boldsymbol{R}_a,\boldsymbol{R}_b) = \|\boldsymbol{R}_a - \boldsymbol{R}_b\|_{\mathrm{F}} = \left\|\boldsymbol{R}_a^{\mathsf{T}}\boldsymbol{R}_b - \mathbf{I}_3\right\|_{\mathrm{F}}$$

- *Quaternion distance*: If we call $\boldsymbol{q}_a$ (resp. $\boldsymbol{q}_b$) the unit quaternion representation of the rotation matrix $\boldsymbol{R}_a$ (resp. $\boldsymbol{R}_b$), the quaternion distance is:

$$\mathrm{d}_{\mathrm{quat}}(\boldsymbol{R}_a,\boldsymbol{R}_b) = \min\left(\|\boldsymbol{q}_a - \boldsymbol{q}_b\|\ ,\ \|\boldsymbol{q}_a + \boldsymbol{q}_b\|\right) \quad (6)$$

  The $\min$ operator in the definition (6) is used to solve the sign ambiguity, since the quaternions $\boldsymbol{q}_a$ and $-\boldsymbol{q}_a$ represent the same rotation (for this reason, unit quaternions constitute a *double cover* of $\mathrm{SO}(3)$).

If we call θ the rotation angle of $\boldsymbol{R}_a^{\mathsf{T}}\boldsymbol{R}_b$, the following equalities hold [16]:

$$\begin{aligned}
\mathrm{d}_{\mathrm{ang}}(\boldsymbol{R}_a,\boldsymbol{R}_b) &= \theta && (7)\\
\mathrm{d}_{\mathrm{chord}}(\boldsymbol{R}_a,\boldsymbol{R}_b) &= 2\sqrt{2}\sin(\theta/2) \approx \sqrt{2}\,\theta && (8)\\
\mathrm{d}_{\mathrm{quat}}(\boldsymbol{R}_a,\boldsymbol{R}_b) &= 2\sin(\theta/4) \approx \theta/2 && (9)
\end{aligned}$$

where after the sign $\approx$ we report the first-order approximations. The equalities ensure that the metrics are essentially the same (up to a constant factor) for small rotation errors.

III. Techniques for 3D Rotation Estimation

Sections III-A to III-E describe 5 different techniques to solve or approximate the solution of Problem $\mathcal{P}$ in (5), for some choice of the distance metric $\mathrm{d}_{\mathrm{SO}(3)}(\cdot,\cdot)$.

A. Single Loop Solution [15], [25], [30]

This technique returns the optimal solution of Problem $\mathcal{P}$ with the angular distance and for the specific case of graphs with a single loop (Fig. 2). While this technique first appeared in computer vision [15], it is known in robotics as *trajectory bending* [33], [25]. It also appeared recently in [30]. Let us define the *ordered* set $\mathcal{L} = \{(1,2),(2,3),\ldots,(n-1,n),(n,1)\}$, which collects the edges along the loop. Using this definition Problem $\mathcal{P}$ becomes:

$$\min_{\{\boldsymbol{R}_i\}\in\mathrm{SO}(3)} \sum_{(i,j)\in\mathcal{L}} \left\|\mathrm{Log}\left(\boldsymbol{R}_{ij}^{\mathsf{T}}\boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{R}_j\right)\right\|^2 \quad (10)$$

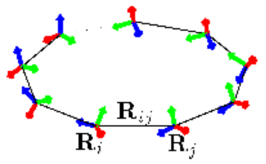


Fig. 2. Single-loop pose graph with odometric edges (solid line) and a single loop closure (dotted line).

The intuition to solve (10) is that relative rotations should compose to the identity along the loop. Since the measurements are noisy, the measured rotations do not compose to the identity and the rotation estimator has to optimally distribute this rotation "excess" among the edges.

To exploit this insight, we re-parametrize (10) in terms of *relative*, rather than *absolute* rotations. Let us define:

$$\tilde{\boldsymbol{R}}_{ij} \doteq \boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{R}_j, \quad (i,j)\in\mathcal{L} \quad (11)$$

By construction, the rotations $\tilde{\boldsymbol{R}}_{ij}$ in (11), has to compose to the identity along the loop. Therefore, we write (10) as:

$$\begin{aligned}
\min_{\{\tilde{\boldsymbol{R}}_{ij}\}\in\mathrm{SO}(3)} \quad & \textstyle\sum_{(i,j)\in\mathcal{L}} \left\|\mathrm{Log}\left(\boldsymbol{R}_{ij}^{\mathsf{T}}\tilde{\boldsymbol{R}}_{ij}\right)\right\|^2 \\
\text{subject to} \quad & \textstyle\prod_{(i,j)\in\mathcal{L}} \tilde{\boldsymbol{R}}_{ij} = \mathbf{I}_3, \qquad (12)
\end{aligned}$$

where the product $\prod_{(i,j)\in\mathcal{L}}$ is *ordered* according to the set $\mathcal{L}$ (3D rotations do not commute). Eq. (12) has a very intuitive explanation: we look for rotations $\tilde{\boldsymbol{R}}_{ij}$ that are close to the measurements $\boldsymbol{R}_{ij}$ (in the sense of the angular distance), and that compose to the identity along the loop.

Our second change of variable is:

$$\tilde{\boldsymbol{E}}_{ij} \doteq \boldsymbol{R}_{ij}^{\mathsf{T}}\tilde{\boldsymbol{R}}_{ij}, \quad (i,j)\in\mathcal{L}, \quad (13)$$

that, applied to (12), gives:

$$\begin{aligned}
\min_{\{\tilde{\boldsymbol{E}}_{ij}\}\in\mathrm{SO}(3)} \quad & \textstyle\sum_{(i,j)\in\mathcal{L}} \left\|\mathrm{Log}\left(\tilde{\boldsymbol{E}}_{ij}\right)\right\|^2 \\
\text{subject to} \quad & \textstyle\prod_{(i,j)\in\mathcal{L}} \boldsymbol{R}_{ij}\tilde{\boldsymbol{E}}_{ij} = \mathbf{I}. \qquad (14)
\end{aligned}$$

Also here the interpretation is simple: we look for small *corrections* $\tilde{\boldsymbol{E}}_{ij}$ that can help to satisfy the loop constraint in (14). Rearranging the rotations, the constraint in (14) can be written as (see [15] for a complete derivation):

$$\prod_{(i,j)\in\mathcal{L}} \boldsymbol{S}_j\tilde{\boldsymbol{E}}_{ij}\boldsymbol{S}_j^{\mathsf{T}} = \boldsymbol{S}_{\mathcal{L}}^{\mathsf{T}}, \quad (15)$$

where $\boldsymbol{S}_j \doteq \prod_{(k-1,k)\in\mathcal{L},k\leq j}\boldsymbol{R}_{ij}$, and $\boldsymbol{S}_{\mathcal{L}} \doteq \prod_{(i,j)\in\mathcal{L}}\tilde{\boldsymbol{R}}_{ij}$; $\boldsymbol{S}_{\mathcal{L}}$ represents the total rotation "excess" we need to compensate, while each term in the product of (15) represents the error on each edge, all in the reference frame of node 1.

This justifies our last change of variables:

$$\tilde{\boldsymbol{T}}_{ij} \doteq \boldsymbol{S}_j\tilde{\boldsymbol{E}}_{ij}\boldsymbol{S}_j^{\mathsf{T}}, \quad (i,j)\in\mathcal{L}. \quad (16)$$

Substituting (16) and (15) in (14), and recalling that $\|\mathrm{Log}(\boldsymbol{S}_j^{\mathsf{T}}\tilde{\boldsymbol{T}}_{ij}\boldsymbol{S}_j)\| = \|\boldsymbol{S}_j^{\mathsf{T}}\mathrm{Log}(\tilde{\boldsymbol{T}}_{ij})\| = \|\mathrm{Log}(\tilde{\boldsymbol{T}}_{ij})\|$, we get:

$$\begin{aligned}
\min_{\{\tilde{\boldsymbol{T}}_{ij}\}\in\mathrm{SO}(3)} \quad & \textstyle\sum_{(i,j)\in\mathcal{L}} \left\|\mathrm{Log}\left(\tilde{\boldsymbol{T}}_{ij}\right)\right\|^2 \\
\text{subject to} \quad & \textstyle\prod_{(i,j)\in\mathcal{L}} \tilde{\boldsymbol{T}}_{ij} = \boldsymbol{S}_{\mathcal{L}}^{\mathsf{T}} \qquad (17)
\end{aligned}$$

[2]Formally, the logarithm map returns an element of the tangent space (a skew symmetric matrix), whose exponential is $\boldsymbol{R}$. Since every 3×3 skew symmetric matrix can be unequivocally mapped to a vector in $\mathbb{R}^3$ (via the *vee* operator [24]), our notation comes without loss of generality.

Preprint submitted to 2015 IEEE International Conference on Robotics and Automation. Received September 30, 2014.

which essentially requires to find rotations $\tilde{\boldsymbol{T}}_{ij}$ whose composition is a known rotation $\boldsymbol{S}_{\mathcal{L}}^{\mathsf{T}}$, and such that the rotation angles are small, in the sense of the angular norm $\|\mathrm{Log}(\tilde{\boldsymbol{T}}_{ij})\|$.

It is possible to show [15] that the minimum of (17) is attained when the rotations $\tilde{\boldsymbol{T}}_{ij}$ have the same rotation axis of $\boldsymbol{S}_{\mathcal{L}}^{\mathsf{T}}$ and rotation angle equal to $\|\mathrm{Log}\left(\boldsymbol{S}_{\mathcal{L}}^{\mathsf{T}}\right)\|/m$:

$$\tilde{\boldsymbol{T}}_{ij}^{\star} = \mathrm{Exp}\left(\frac{\mathrm{Log}\left(\boldsymbol{S}_{\mathcal{L}}^{\mathsf{T}}\right)}{m}\right) \tag{18}$$

where $\mathrm{Exp}\,(\cdot)$ is the exponential map for $\mathrm{SO}(3)$.

Substituting $\tilde{\boldsymbol{T}}_{ij}^{\star}$ back into (16) and (13), we get the optimal relative rotations. We then retrieve the desired absolute rotations from (11), by chaining the relative rotations.

Related work [15], [34], [30] also discusses extensions to the multi-loop case; these extensions are iterative in nature and have no global convergence guarantees.

B. Chordal Relaxation [1]

This section reviews the approach proposed by Martinec and Pajdla [1]. This technique does not return the optimal solution of Problem $\mathcal{P}$ in general, but, as shown in Section IV, it performs astonishingly well in practice.

Let us use the chordal distance in Problem $\mathcal{P}$:

$$\min_{\{\boldsymbol{R}_i\}\in\mathrm{SO}(3)} \sum_{(i,j)\in\mathcal{E}} \left\|\boldsymbol{R}_{ij} - \boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{R}_j\right\|_{\mathrm{F}}^2 =$$
$$\min_{\{\boldsymbol{R}_i\}\in\mathrm{SO}(3)} \sum_{(i,j)\in\mathcal{E}} \left\|\boldsymbol{R}_{ij}\boldsymbol{R}_j^{\mathsf{T}} - \boldsymbol{R}_i^{\mathsf{T}}\right\|_{\mathrm{F}}^2 . \tag{19}$$

If we call $\boldsymbol{r}_i^k$ the k-th row of $\boldsymbol{R}_i$ ($k = 1,2,3$), and we write each row as a column vector, eq. (19) becomes:

$$\begin{aligned} \min_{\{\boldsymbol{r}_i^k\}} \quad & \textstyle\sum_{(i,j)\in\mathcal{E}} \sum_{k=1,2,3} \left\|\boldsymbol{R}_{ij}\boldsymbol{r}_i^k - \boldsymbol{r}_j^k\right\|^2 \\ \text{subject to} \quad & \left[\boldsymbol{r}_i^1\ \boldsymbol{r}_i^2\ \boldsymbol{r}_i^3\right]^{\mathsf{T}} \in \mathrm{SO}(3), \quad i = 1,\ldots,n, \end{aligned} \tag{20}$$

where the constraint restricts the choices of $\boldsymbol{r}_i^k$ to vectors that form meaningful rows of a rotation matrix (essentially, orthonormal vectors that follow the right-hand rule).

The idea behind the second technique is a very simple one. Rather than solving directly problem (20), one first solves an unconstrained version of (20):

$$\min_{\{\boldsymbol{r}_i^k\}} \sum_{(i,j)\in\mathcal{E}} \sum_{k=1,2,3} \left\|\boldsymbol{R}_{ij}\boldsymbol{r}_i^k - \boldsymbol{r}_j^k\right\|^2 \tag{21}$$

and obtains n matrices $\boldsymbol{M}_i \doteq \left[\boldsymbol{r}_i^1\ \boldsymbol{r}_i^2\ \boldsymbol{r}_i^3\right]^{\mathsf{T}}$ (which are not rotations in general). Each rotation is then computed as:

$$\boldsymbol{R}_i^{\star} = \operatorname*{arg\,min}_{\boldsymbol{R}_i\in\mathrm{SO}(3)} \|\boldsymbol{M}_i - \boldsymbol{R}_i\|_{\mathrm{F}}^2 \tag{22}$$

which looks for the closest rotation matrix (in the Frobenius norm sense) to $\boldsymbol{M}_i$. The advantage is that (21) is a linear least-squares problem. Moreover, problem (22) admits a closed-form solution [16]: if we compute the singular value decomposition $\boldsymbol{M}_i = \boldsymbol{S}\boldsymbol{D}\boldsymbol{V}^{\mathsf{T}}$, then:

$$\boldsymbol{R}_i^{\star} = \boldsymbol{S}\ \mathrm{diag}\left([1\ 1\ \det(\boldsymbol{S}\boldsymbol{V}^{\mathsf{T}})]\right)\ \boldsymbol{V}^{\mathsf{T}}. \tag{23}$$

Remark 1 (Homogeneous least squares): Problem (20) is a homogeneous least squares problem, hence admits a trivial solution in which the vectors are all zero. This reflects an observability issue as we are trying to estimate global rotations from relative measurements (the global frame is unobservable). We can solve this indetermination by including a prior on a rotation (e.g., the first rotation is $\boldsymbol{R}_1 = \mathbf{I}_3$), or imposing a norm constraint as in [1]. We adopt the first solution as it easily extends to the presence of other priors, such as the one in the following subsection.

1) Including vertical priors: In this subsection we present an original extension of the chordal relaxation technique [1] to include vertical direction measurements. Assume that the robot can measure the vertical direction $\boldsymbol{v}_i$ in the local frame $\boldsymbol{R}_i$. For instance, it can sense the gravity vector using an IMU. In the global frame the vertical direction is $\boldsymbol{g} = [0\ 0\ 1]^{\mathsf{T}}$. The measurement model is:

$$\boldsymbol{v}_i = \boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{g} + \boldsymbol{v}_i^{\epsilon} \tag{24}$$

where $\boldsymbol{v}_i^{\epsilon}$ represents measurement noise and the matrix $\boldsymbol{R}_i^{\mathsf{T}}$ transforms the vector $\boldsymbol{g}$ to the local frame. Exploiting the fact that $\boldsymbol{g} = [0\ 0\ 1]^{\mathsf{T}}$, it is easy to see that $\boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{g} = \boldsymbol{r}_i^3$, i.e., $\boldsymbol{v}_i$ is a noisy measurement of the last row of $\boldsymbol{R}_i$. Therefore, if we have a set of vertical measurements $\mathcal{V}$, problem (21) can be easily extended to:

$$\min_{\{\boldsymbol{r}_i^k\}} \sum_{(i,j)\in\mathcal{E}} \sum_{k=1,2,3} \left\|\boldsymbol{R}_{ij}\boldsymbol{r}_i^k - \boldsymbol{r}_j^k\right\|^2 + \sum_{i\in\mathcal{V}} \left\|\boldsymbol{r}_i^3 - \boldsymbol{v}_i\right\|^2 \tag{25}$$

which is still a linear least squares problem. A small example in which we estimate rotations via (25) is reported in the supplementary material [35].

C. Quaternion Relaxation [12], [16]

This section reviews the rotation estimation approach of Govindu [12] and the recent analysis of Hartley *et al.* [16]. This approach uses the quaternion distance in Problem $\mathcal{P}$:

$$\begin{aligned} \min_{\{\boldsymbol{q}_i\},\{b_{ij}\}} \quad & \sum_{(i,j)\in\mathcal{E}} \left\|\boldsymbol{q}_{ij} - b_{ij}\ \boldsymbol{q}_i^{-1}\cdot\boldsymbol{q}_j\right\|^2 \\ \text{subject to} \quad & \|\boldsymbol{q}_i\|^2 = 1, \quad i = 1,\ldots,n \\ & b_{ij} \in \{-1,+1\}, \quad (i,j)\in\mathcal{E} \end{aligned} \tag{26}$$

where $\cdot$ denotes quaternion multiplication, and we use $b_{ij} \in \{-1,+1\}$ to model the sign ambiguity (compare with (6)).

Problem (26) is hard for the presence of integer variables b_{ij} and because the norm constraints are nonconvex.

Hartley *et al.* [16] propose to solve (26) in two steps. First, determine the signs b_{ij}, and then solve (26) with fixed b_{ij}. This two-stage solution is suboptimal in general, but it works well for low levels of noise (Section IV). Let us review the two steps required to (approximately) solve (26).

1) Computing the signs b_{ij}: Hartley *et al.* [16] propose to determine b_{ij} using a spanning tree of the graph. Here we give a different interpretation, based on the *cycles* of the graph. We believe this interpretation is interesting as (i) it shows that there are only ℓ integer variables to determine, where $\ell = m - n + 1$ is the number of cycles in the graph, and (ii) it draws connections with the planar solution [7].

As we did in Section III-A, we apply a change of variables, so to work on the relative rotations:

$$\tilde{\boldsymbol{q}}_{ij} \doteq b_{ij}\ \boldsymbol{q}_i^{-1}\cdot\boldsymbol{q}_j \tag{27}$$

From the definition (27), the relative rotations $\tilde{\boldsymbol{q}}_{ij}$ satisfy:

$$\prod_{(i,j)\in\mathcal{L}_k} b_{ij}\tilde{\boldsymbol{q}}_{ij} = \mathbf{I}_4, \;\; k=1,\ldots,\ell \tag{28}$$

where $\prod_{(i,j)\in\mathcal{L}_k}$ is the ordered product over the edges within the k-th cycle in the graph, denoted with $\mathcal{L}_k$. This equality imposes that the (to-be-estimated) quaternions have to compose (up to sign) to the identity rotation along loops.

For reasonable measurement noise, the estimates $\tilde{\boldsymbol{q}}_{ij}$ will be close to the measurements $\boldsymbol{q}_{ij}$, and for this reason, we can approximate the constraint in (28) as:

$$\prod_{(i,j)\in\mathcal{L}_k} b_{ij}\boldsymbol{q}_{ij} \approx \mathbf{I}_4, \;\; k=1,\ldots,\ell \tag{29}$$

Since b_{ij} are scalars, the previous expression is the same as:

$$b_k \prod_{(i,j)\in\mathcal{L}_k} \boldsymbol{q}_{ij} \approx \mathbf{I}_4, \;\; k=1,\ldots,\ell \tag{30}$$

where $b_k \doteq \prod_{(i,j)\in\mathcal{L}_k} b_{ij}$. From (30), one can determine the signs as follows: for each cycle, one computes $\prod_{(i,j)\in\mathcal{L}_k} \boldsymbol{q}_{ij}$: if the product is close $\mathbf{I}_4$ then $b_k = 1$; if the product is close to $-\mathbf{I}_4$, one chooses $b_k = -1$. One can build cycles of the graph from a spanning tree, such that each chord of the spanning tree belongs to a single cycle. This explains the approach of [16]: one sets $b_{ij} = +1$ for all edges in the spanning tree, and controls the sign b_k of the k-th cycle, using the sign of the corresponding chord.

Two interesting insights stem from reasoning in terms of cycles. First, we see there are only ℓ signs b_k to determine, rather than m as in Problem (26). Second, for large levels of noise, the product $\prod_{(i,j)\in\mathcal{L}_k} \boldsymbol{q}_{ij}$ can be far from $\pm\mathbf{I}_4$ and can lead to a bad decision on b_k. Since the product $\prod_{(i,j)\in\mathcal{L}_k} \boldsymbol{q}_{ij}$ quantifies the accumulation of measurement error along a cycle, this suggests that, to have better decisions on b_k, we have to choose small cycles. Hence, we arrive to the same conclusion of [7], that tells us that the cycle structure of the graph and the use of a *minimum cycle basis* are the keys of robust (2D) rotation estimation.

2) Solving Problem (26) *with known* b_{ij}*:* After computing the signs b_{ij} problem (26) becomes:

$$\begin{aligned} \min_{\{\boldsymbol{q}_i\}} \quad & \sum_{(i,j)\in\mathcal{E}} \left\|\boldsymbol{q}_{ij}^{+} - \boldsymbol{q}_i^{-1}\cdot\boldsymbol{q}_j\right\|^2 \\ \text{subject to} \quad & \|\boldsymbol{q}_i\|^2 = 1, \quad i=1,\ldots,n \end{aligned} \tag{31}$$

where we denote with $\boldsymbol{q}_{ij}^{+} = b_{ij}\boldsymbol{q}_{ij}$ the sign-corrected measurements. Recalling that the multiplication between two quaternions $\boldsymbol{q}_c = \boldsymbol{q}_a \cdot \boldsymbol{q}_b$ can be computed using standard matrix-vector multiplication, and using compact matrix notation, we rewrite (31) as:

$$\begin{aligned} \min_{\boldsymbol{q}} \quad & \|\boldsymbol{Q}\boldsymbol{q}\|^2 \\ \text{subject to} \quad & \boldsymbol{q}\boldsymbol{N}_i\boldsymbol{q} = 1, \qquad i=1,\ldots,n \end{aligned} \tag{32}$$

where $\boldsymbol{Q}$ is a suitable sparse matrix, $\boldsymbol{q}\in\mathbb{R}^{4n}$ is a vector stacking all (unknown) quaternions, and $\boldsymbol{N}_i$ is a sparse matrix that writes the i-th norm constraint in compact form.

The work [12] relaxes the norm constraint in (32), and reduces (32) to a homogeneous least-squares problem, which can be solved as prescribed in Remark 1. From the relaxed solution we can extract the 4-vectors, corresponding to each rotation: $\boldsymbol{q}_i^\star$, $i=1,\ldots,n$. In general, these vector $\boldsymbol{q}_i^\star$ do not have unit norm, hence they do not represent valid rotations. Therefore, after computing $\boldsymbol{q}_i^\star$ from (32), one has to normalize the resulting vectors to have unit norm.

D. Semidefinite Programming (SDP) Relaxation [17]

This technique has been proposed in [17] and aims at solving (32), without resorting to relaxation of the norm constraints. While the original presentation is based on duality theory, we here provide a simpler explanation that does not require the introduction of the dual problem. Note that the approach [17] implicitly assumes that the signs of the measurements have been corrected (Section III-C.1).

The key observation behind the SDP relaxation is that for any vector $\boldsymbol{y}$ and matrix $\boldsymbol{W}$, it holds: $\boldsymbol{y}^\mathsf{T}\boldsymbol{W}\boldsymbol{y} = \operatorname{tr}\left(\boldsymbol{W}(\boldsymbol{y}\boldsymbol{y}^\mathsf{T})\right)$. This allows rewriting (32) as:

$$\begin{aligned} \min_{\boldsymbol{q}} \quad & \operatorname{tr}\left(\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q}(\boldsymbol{q}\boldsymbol{q}^\mathsf{T})\right) \\ \text{subject to} \quad & \operatorname{tr}\left(\boldsymbol{N}_i(\boldsymbol{q}\boldsymbol{q}^\mathsf{T})\right) = 1, \qquad i=1,\ldots,n \end{aligned} \tag{33}$$

The product $\boldsymbol{q}\boldsymbol{q}^\mathsf{T}$ defines a positive semidefinite matrix with rank 1, i.e., the following sets are identical:

$$\{\boldsymbol{q}\boldsymbol{q}^\mathsf{T} : \boldsymbol{q}\in\mathbb{R}^{4n}\} = \{\boldsymbol{Z}\in\mathbb{R}^{4n\times 4n} : \boldsymbol{Z}\succeq 0, \operatorname{rank}(\boldsymbol{Z}) = 1\}$$

Therefore, problem (33) is the same as:

$$\begin{aligned} \min_{\boldsymbol{Z}\succeq 0} \quad & \operatorname{tr}\left(\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q}\boldsymbol{Z}\right) \\ \text{subject to} \quad & \operatorname{tr}\left(\boldsymbol{N}_i\boldsymbol{Z}\right) = 1, \quad i=1,\ldots,n \\ & \operatorname{rank}(\boldsymbol{Z}) = 1 \end{aligned} \tag{34}$$

Problem (34) is still nonconvex, due to the rank constraint. The idea of the SDP relaxation is to solve (34) without enforcing the rank constraint. The resulting problem is a semidefinite optimization problem (SDP), which can be solved via convex programming. The interesting observation is that, if the solution $\boldsymbol{Z}^\star$ of the SDP has rank 1, then it is also optimal for (33), and it can be factored as $\boldsymbol{Z}^\star = (\boldsymbol{q}^\star)(\boldsymbol{q}^\star)^\mathsf{T}$, which solves the original problem (32). While it is not guaranteed to have a rank-1 $\boldsymbol{Z}^\star$, the work [17] shows that it is often the case in practice, for low levels of noise.

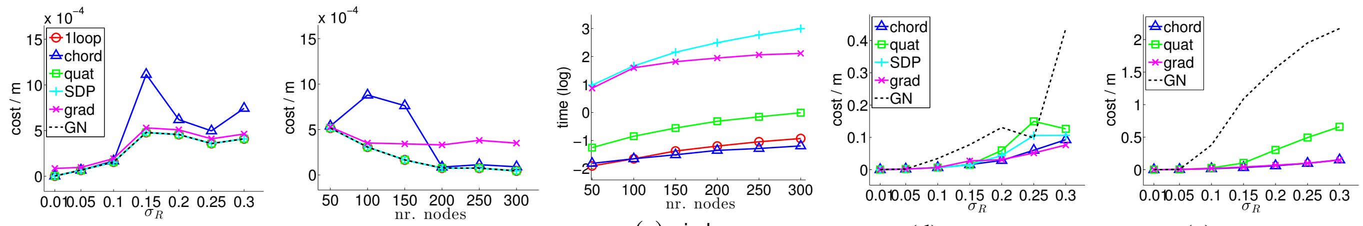
E. Riemannian Gradient Descent [31]

The approach presented in this section has been proposed in [31]; it is iterative in nature and it is included in our survey as it has been shown (in its *consensus* variant [20]) to have global convergence properties. The work [31] shows that, in a noiseless case, Problem $\mathcal{P}$ can be formulated as a consensus problem; however, in presence of noise the equivalence is not exact and the strong convergence result of [20] is not guaranteed for Problem $\mathcal{P}$. For this reason, we will evaluate the convergence properties numerically, in Section IV.

The basic idea is to work on a *reshaped* version of the cost function in Problem $\mathcal{P}$:

$$\min_{\{\boldsymbol{R}_i\}\in\mathrm{SO}(3)} \sum_{(i,j)\in\mathcal{E}} f\left(\mathrm{d}_{\mathrm{SO}(3)}\left(\boldsymbol{R}_{ij}, \boldsymbol{R}_i^\mathsf{T}\boldsymbol{R}_j\right)\right) \tag{35}$$

where $f:[0,+\pi]\mapsto\mathbb{R}$ is a given *reshaping* function. Intuitively, rather than using the distance $\mathrm{d}_{\mathrm{SO}(3)}\left(\boldsymbol{R}_{ij}, \boldsymbol{R}_i^\mathsf{T}\boldsymbol{R}_j\right)$,



(a) circle (b) circle (c) circle (d) torus (e) cube

Fig. 3. (a) Cost VS rotation noise (std σ_R in radians) for the circle scenario and for all techniques; (b) Cost VS number of nodes for the circle scenario; (c) CPU time VS number of nodes for the circle scenario; (d) Cost VS noise for the torus scenario; (e) Cost VS noise for the cube scenario.

that can be prone to convergence to local minima, one works on the function $f\left(\mathrm{d}_{\mathrm{SO}(3)}\left(\boldsymbol{R}_{ij}, \boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{R}_j\right)\right)$, which is well-behaved, if we choose $f(\cdot)$ wisely.

The work [20] proposes to use the angular distance $\theta = \|\mathrm{Log}\left(\boldsymbol{R}_{ij}^{\mathsf{T}}\boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{R}_j\right)\|$ and the following reshaping function:

$$f(\theta) = \frac{\pi^2}{2f_0(\pi)} f_0(\theta), \quad \text{with} \quad f_0(\theta) = \frac{1}{b} - \left(\frac{1}{b} + \theta\right)\exp(-b\theta) \tag{36}$$

where b is a constant. The cost (35) is then optimized using a gradient descent method, which –given the current estimate $\boldsymbol{R}_i^{(t)}$, $i = 1, \ldots, n$– updates the rotations via:

$$\boldsymbol{R}_i^{(t+1)} = \boldsymbol{R}_i^{(t)} \mathrm{Exp}\left(\epsilon\, \boldsymbol{s}^{(t)}\right) \tag{37}$$

where $\boldsymbol{s}^{(t)}$ is the gradient of the cost function evaluated at $\boldsymbol{R}_i^{(t)}$, and ϵ is a given stepsize.

The algorithm is iterative in nature, hence it needs initial guess $\boldsymbol{R}_i^{(0)}$, $i = 1, \ldots, n$. When applied to consensus problems, it guarantees almost sure convergence to a global minimum ([20], Theorem 16), as long as the stepsize satisfies $\epsilon < \frac{2f_0(\pi)}{\pi^2 b \deg(\mathcal{G})}$, where $\deg(\mathcal{G})$ is the maximum node degree of the graph. The basic intuition is that the reshaping function makes local minimizers unstable equilibria points, and the estimate is unlikely to converge to those.

IV. Experimental Evaluation and Comparison

We first test the 5 techniques of Section III on Problem $\mathcal{P}$ (Section IV-A). Then we use the best performing techniques to initialize pose graph optimization (Section IV-B).

A. Comparison on Rotation Estimation

Here we show that the chordal relaxation (Martinec and Pajdla [1]) and the gradient method (Tron and Vidal [31]) outperform the other techniques in solving Problem $\mathcal{P}$.

Compared techniques. We use the following short names for the 5 techniques: 1loop (Section III-A), chord (Section III-B), quat (Section III-C), SDP (Section III-D), and grad (Section III-E). The results of this section are based on a Matlab implementation of the 5 techniques. For the SDP technique, we used CVX/MOSEK [36] as parser/solver. The gradient method is initialized at the odometric trajectory and we set $b = 1$ in eq. (36). When interesting, we include results from a standard Gauss-Newton method (implemented using gtsam [32]) initialized at the odometric trajectory (label: GN).

Benchmarking scenarios. We created different benchmarking scenarios. In the circle scenario (Fig. 2) poses are uniformly spaced along a single loop and a random rotation is assigned to each pose. In the torus scenario (Fig. 1) the trajectory is simulated as the robot were traveling on the surface of a torus. Random loop closures are added between nearby nodes. Similarly, in the cube scenario (Fig. 1) the trajectory is simulated as the robot were traveling on a 3D grid world. We created different instances of these scenarios by changing the number of nodes n, and by simulating different noise levels for the rotation measurements (std σ_R).

Performance metrics. For each technique, we evaluate the cost attained in Problem $\mathcal{P}$, using the angular distance as metric (we are solving a minimization problem hence the lower the better). The angular distance is a common choice in pose graph optimization and Section II-B assures that for small residual errors the distances differ by a constant (that vanishes in the optimization). In the figures we normalize this cost by the number of measurements m, such that the resulting curve describes the average (squared) residual error for each rotation measurement. Timing results are also discussed when relevant. Results are averaged over 10 runs.

Results. Fig. 3(a) shows the cost for the scenario circle with $n = 100$ nodes and for different rotation noise σ_R. The single loop scenario can be managed pretty easily by all techniques. chord performs slightly worse than the others, but –observing the scale of the plot– the difference is small (compare with Fig. 3(d)-(e)). Fig. 3(b) shows cost versus numbers of nodes, for the circle scenario, with $\sigma_R = 0.1$. Also here the differences among the techniques are minor.

Fig. 3(c) shows the CPU time required by each techniques (we excluded GN, that is implemented in c++). The plot is on log scale: while the techniques chord, quat, 1loop are very cheap, SDP and grad require 3 orders of magnitude more time and are impractical, even for small problems.

Fig. 3(d) shows cost versus rotation noise for the torus scenario (200 nodes). Here, we have multiple loops, hence we exclude the 1loop technique. GN has the lowest breakdown point, and easily converges to a local minimum. The quat and SDP techniques are slightly more resilient, but they still have larger errors for $\sigma_R > 0.15$: for large noise, one can select the wrong integers using the approach of Section III-C.1. The grad and chord techniques have a graceful decrease in performance for increasing rotation noise.

Fig. 3(e) shows cost versus rotation noise for the cube scenario (10^3 nodes). This scenario was too large for the SDP approach: while SDPs are convex problems, they do no scale well with problem size [17], and CVX was not able to produce a solution. The remaining techniques have a trend similar to the one of Fig. 3(d): grad and chord are the only techniques that can tolerate large levels of noise.

CPU times for the torus and cube scenarios have the same trend of Fig. 3(c) and are omitted for space reasons: timing plots and extra results can be found at [35].

		odometry	g2o	g2oST	gtsam	chord+gtsam		grad+gtsam		
sphere	Iter.	–	5	5	7	1	4	1	5	
$n = 2500$	Cost	$1.29 \cdot 10^6$	$4.39 \cdot 10^3$	$7.91 \cdot 10^2$	$6.76 \cdot 10^2$	$9.63 \cdot 10^2$	$6.76 \cdot 10^2$	$1.24 \cdot 10^4$	$6.76 \cdot 10^2$	}1435
$m = 4949$	Time	–	1.19	1.23	0.96	0.52	0.85	6.40	6.89	
sphere-a	Iter.	–	5	5	1	1	4	1	1	
$n = 2200$	Cost	$1.34 \cdot 10^8$	$5.32 \cdot 10^{10}$	$5.43 \cdot 10^6$	$5.71 \cdot 10^{10}$	$1.51 \cdot 10^6$	$1.49 \cdot 10^6$	$1.9 \cdot 10^6$	$1.9 \cdot 10^6$	}8155
$m = 8647$	Time	–	2.46	2.48	0.46	0.79	1.28	50.26	50.87	
torus	Iter.	–	5	5	2	1	4	1	12	
$n = 5000$	Cost	$1.99 \cdot 10^6$	$6.04 \cdot 10^8$	$1.27 \cdot 10^4$	$4.71 \cdot 10^{10}$	$1.24 \cdot 10^4$	$1.21 \cdot 10^4$	$5.85 \cdot 10^4$	$2.81 \cdot 10^4$	}1231
$m = 9048$	Time	–	3.90	3.90	0.83	1.18	1.97	11.35	14.45	
cube	Iter.	–	5	5	2	1	4	1	4	
$n = 8000$	Cost	$7.32 \cdot 10^7$	$5.39 \cdot 10^7$	$4.6 \cdot 10^4$	$6.58 \cdot 10^{11}$	$4.51 \cdot 10^4$	$4.22 \cdot 10^4$	$4.61 \cdot 10^4$	$4.22 \cdot 10^4$	}1045
$m = 22236$	Time	–	188.08	187.32	15.80	31.33	53.78	31.60	54.80	
garage	Iter.	–	5	5	4	1	4	1	4	
$n = 1661$	Cost	$8.36 \cdot 10^3$	$6.43 \cdot 10^{-1}$	$6.43 \cdot 10^{-1}$	$6.35 \cdot 10^{-1}$	$1.51 \cdot 10^0$	$6.35 \cdot 10^{-1}$	$1.12 \cdot 10^1$	$6.35 \cdot 10^{-1}$	}234
$m = 6275$	Time	–	0.32	0.33	0.43	0.32	0.48	1.23	1.42	
cubicle	Iter.	–	5	5	1	1	4	1	5	
$n = 5750$	Cost	$4.99 \cdot 10^6$	$5.16 \cdot 10^{21}$	$8.65 \cdot 10^{23}$	$5.91 \cdot 10^7$	$8.02 \cdot 10^4$	$1.62 \cdot 10^3$	$1.7 \cdot 10^6$	$1.62 \cdot 10^3$	}10000
$m = 16869$	Time	–	4.25	4.28	0.78	1.34	2.50	145.46	145.60	
rim	Iter.	–	5	5	1	1	5	1	1	
$n = 10195$	Cost	$4.5 \cdot 10^7$	$9.37 \cdot 10^{21}$	$1.82 \cdot 10^{25}$	$6.78 \cdot 10^8$	$1.31 \cdot 10^6$	$6.63 \cdot 10^4$	$4.86 \cdot 10^7$	$4.86 \cdot 10^7$	}10000
$m = 29743$	Time	–	7.52	7.83	1.42	2.59	5.06	293.18	289.08	

TABLE I

COST ATTAINED IN (2), CPU TIME, AND NUMBER OF GAUSS-NEWTON ITERATIONS FOR DIFFERENT BENCHMARKING PROBLEMS, COMPARING THE ODOMETRIC COST, THE COST ATTAINED BY g2o AND gtsam, AND THE PROPOSED INITIALIZATION chord+gtsam AND grad+gtsam.

B. *Initialization for Pose Graph Optimization*

In this section we show that the use of the chordal relaxation (Martinec and Pajdla [1]), as initialization for pose graph optimization, entails a performance boost in terms of speed and robustness. The gradient method (Tron and Vidal [31]) is less competitive, as it requires many iterations to converge, and sometimes is trapped in local minima.

Compared techniques. We use chord and grad techniques to initialize pose graph optimization. Here, computation speed is important, hence we implemented the two techniques in c++ and released the code in gtsam [32]. The initialization works as follows: we first solve for the rotations, and then use the rotation estimate as initial guess for a Gauss-Newton method (available in gtsam) that solves (2). The techniques using this initialization are called chord+gtsam and grad+gtsam, depending on the approach used for rotation estimation. We compare these techniques against state-of-the-art solvers that apply Gauss-Newton from the odometric guess: g2o [37] and gtsam [32]. We also compare against a technique that applies Gauss-Newton from a spanning tree initialization (label: g2oST); this technique is available in g2o.

Benchmarking scenarios. We consider 7 benchmarking problems: sphere, sphere-a, garage, torus, cube, cubicle, rim. The sphere dataset is a test problem released in gtsam [32]. The sphere-a dataset (Fig. 1) is a more challenging version with larger noise and is released in g2o [37]. The garage datasets is a real dataset from Vertigo [37]. Besides these standard benchmarks, we test the approaches on the torus and cube datasets ($\sigma_R = 0.1$rad), and on two real datasets (cubicle and rim) collected at the RIM center at Georgia Tech. In the cubicle and rim datasets, the relative pose constraints are obtained via ICP on the point clouds acquired from a 3D laser scanner.

Results. Table I reports the cost attained in (2) (using the angular distance) and the CPU time for the compared approaches. Values highlighted in red indicate that the technique was stuck in a local minima, while values in blue correspond to visually correct estimates. For a visual evaluation, we refer the reader to [35], which shows the trajectory estimates for each cell in Table I.

The sphere dataset is fairly easy and all techniques have good results. g2o stops after 5 iterations by default; gtsam uses a stopping criterion based on the cost and for this reason performs more iterations and attains a slightly smaller cost. For chord+gtsam and grad+gtsam we report the cost obtained doing a single Gauss-Newton iteration from the initialization, and the cost attained by letting gtsam perform multiple iterations. A single iteration in chord+gtsam already produces comparable results w.r.t. 5-7 iterations in g2o and gtsam. The optimal value in attained in 4 iterations. The proposed initialization reduces the CPU time from 1.19s (g2o) to 0.52s (chord+gtsam with 1 iteration). The results are less encouraging for grad+gtsam: the technique produces good results, but implies a large increase in CPU time.

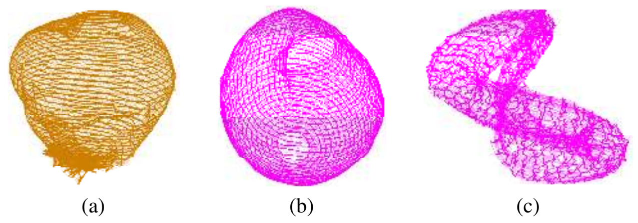


Fig. 4. (a) Estimate from g2oST in the scenario sphere-a. (b) Estimate from grad+gtsam in the scenario sphere-a. (c) Estimate from grad+gtsam in the scenario torus. Trajectories in (a)-(c) correspond to local minima.

While in easy scenarios (as sphere and garage) there is some advantage in using chord+gtsam, the initialization if *extremely* beneficial in difficult scenarios as sphere-a, torus, cube, cubicle and rim: in those scenarios the initial guess is inaccurate and the state-of-the-art techniques fail. gtsam exits after few iterations as it is not able to reduce the cost. g2o perseveres till 5 iterations and often gets worse costs compared with the initial odometric cost. The spanning tree initialization is more resilient but it still fails to produce good trajectories in sphere-a, cubicle, and rim, see Fig. 4(a) and [35].

In all scenarios, chord+gtsam produced very accurate results. The initialization in Fig. 1 is given by chord+gtsam, with a single Gauss-Newton iteration. The initialization is accurate enough to produce a globally consistent 3D reconstruction, as shown in Fig. 5. In all cases, performing multiple iterations in chord+gtsam resulted in the lowest observed cost and a visually correct trajectory. The grad+gtsam approach, besides being very expensive in practice, converged to a local minimum in the sphere and torus datasets, see Fig. 4(b)-(c).

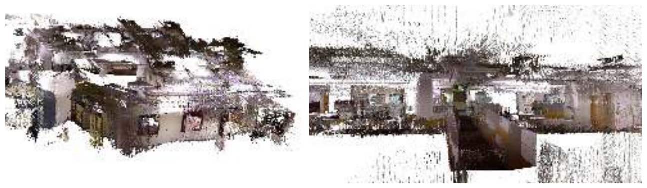


Fig. 5. cubicle: Reconstruction obtained by aligning the 3D laser scan in a global map, using the pose estimate from chord+gtsam (1 iteration).

V. Conclusion

We survey 3D rotation estimation techniques and we show how to use them to initialize pose graph optimization. Some of the surveyed techniques (in particular the one from Martinec and Pajdla [1]) have excellent performance in challenging benchmarking scenarios. On the easy datasets, a good initialization implies a computational advantage, as iterative techniques require less iterations to converge. On datasets with large noise, state-of-the-art approaches are doomed to fail, while the proposed initialization showed extreme resilience and global convergence capability. We released c++ implementations of the best performing techniques and we extended one of the techniques to include vertical prior measurements, as the ones obtained from an IMU sensing gravity. Extra results and visualizations are given in the supplementary material [35].

Acknowledgments

We wish to thank Siddharth Choudhary for providing the cubicle and the rim datasets, and for rendering the 3D point clouds. We also thank Nicu Ştiurcă and Hyun Soo Park for collecting the IMU data used in [35], and Gian Diego Tipaldi and Pratik Agarwal for the useful discussion on g2o.

References

[1] D. Martinec and T. Pajdla, "Robust rotation and translation estimation in multiview reconstruction," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8.

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

[3] G. Grisetti, C. Stachniss, and W. Burgard, "Non-linear constraint network optimization for efficient map learning," *Trans. on Intelligent Transportation systems*, vol. 10, no. 3, pp. 428–439, 2009.

[4] P. Agarwal, G. Grisetti, G. D. Tipaldi, L. Spinello, W. Burgard, and C. Stachniss, "Experimental analysis of Dynamic Covariance Scaling for robust map optimization under bad initial estimates," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014.

[5] L. Carlone, R. Aragues, J. Castellanos, and B. Bona, "A linear approximation for graph-based simultaneous localization and mapping," in *Robotics: Science and Systems (RSS)*, 2011.

[6] ——, "A fast and accurate approximation for planar pose graph optimization," *Intl. J. of Robotics Research*, 2014.

[7] L. Carlone and A. Censi, "From angular manifolds to the integer lattice: Guaranteed orientation estimation with application to pose graph optimization," *IEEE Trans. Robotics*, 2014.

[8] A. Martinelli, "Vision and IMU data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination," *IEEE Trans. Robotics*, vol. 28, no. 1, pp. 44–60, 2012.

[9] ——, "Closed-form solution of visual-inertial structure from motion," *Intl. J. of Computer Vision*, vol. 106, no. 2, pp. 138–152, 2014.

[10] L.Kneip and a. R. S.Weiss, "Deterministic initialization of metric state estimation filters for loosely-coupled monocular vision-inertial systems," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011, pp. 2235–2241.

[11] T. Dong-Si and A. I. Mourikis, "Estimator initialization in vision-aided inertial navigation with unknown camera-IMU calibration," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2012, pp. 1064–1071.

[12] V. M. Govindu, "Combining two-view constraints for motion estimation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001, pp. 218–225.

[13] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri, "Global motion estimation from point matches," in *3DIMPVT*, 2012.

[14] V. Govindu, "Lie-algebraic averaging for globally consistent motion estimation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[15] G. Sharp, S. Lee, and D. Wehe, "Multiview registration of 3D scenes by minimizing error between coordinate frames," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 26, no. 8, pp. 1037–1050, 2004.

[16] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *IJCV*, vol. 103, no. 3, pp. 267–305, 2013.

[17] J. Fredriksson and C. Olsson, "Simultaneous multiple rotation averaging using lagrangian duality," in *Asian Conf. on Computer Vision (ACCV)*, 2012.

[18] F. Bullo, J. Cortes, and S. Martinez, "Distributed control of robotic networks," *Applied Mathematics Series, Princeton University Press*, 2009.

[19] G. Piovan, I. Shames, B. Fidan, F. Bullo, and B. Anderson, "On frame and orientation localization for relative sensing networks," *Automatica*, vol. 49, no. 1, pp. 206–213, 2013.

[20] R. Tron, B. Afsari, and R. Vidal, "Intrinsic consensus on SO(3) with almost global convergence," in *IEEE Conference on Decision and Control*, 2012.

[21] T. Hatanaka, M. Fujita, and F. Bullo, "Vision-based cooperative estimation via multi-agent optimization," in *IEEE Conference on Decision and Control*, 2010.

[22] R. Olfati-Saber, "Swarms on sphere: A programmable swarm with synchronous behaviors like oscillator networks," in *IEEE Conference on Decision and Control*, 2006, pp. 5060–5066.

[23] A. Sarlette and R. Sepulchre, "Consensus optimization on manifolds," *SIAM J. Control and Optimization*, vol. 48, no. 1, pp. 56–76, 2009.

[24] R. Tron, B. Afsari, and R. Vidal, "Riemannian consensus for manifolds with bounded curvature," *IEEE Transactions on Automatic Control*, 2012.

[25] G. Dubbelman, P. Hansen, B. Browning, and M. Dias, "Orientation only loop-closing with closed-form trajectory bending," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2012.

[26] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher, "SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion," *IEEE Trans. Pattern Anal. Machine Intell.*, 2012.

[27] A. Chatterjee and V. M. Govindu, "Efficient and robust large-scale rotation averaging," in *Intl. Conf. on Computer Vision (ICCV)*, 2013, pp. 521–528.

[28] V. M. Govindu, "Robustness in motion averaging," in *Asian Conf. on Computer Vision (ACCV)*, 2006, pp. 457–466.

[29] O. Enqvist, F. Kahl, and C. Olsson, "Non-sequential structure from motion," in *Intl. Conf. on Computer Vision (ICCV)*, 2011, pp. 264–271.

[30] J. R. Peters, D. Borra, B. Paden, and F. Bullo, "Sensor network localization on the group of 3D displacements," *SIAM Journal on Control and Optimization, submitted*, 2014.

[31] R. Tron and R. Vidal, "Distributed 3-D localization of camera sensor networks from 2-D image measurements, accepted," *IEEE Transactions on Automatic Control*, 2014.

[32] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Tech. Rep. GT-RIM-CP&R-2012-002, 2012.

[33] G. Dubbelman, I. Esteban, and K. Schutte, "Efficient trajectory bending with applications to loop closure," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2010, pp. 1–7.

[34] G. Dubbelman and B. Browning, "Closed-form online pose-chain slam," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2013.

[35] 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. supplementary material." [Online]. Available: www.lucacarlone.com/index.php/resources/research/init3d

[36] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming." [Online]. Available: http://cvxr.com/cvx

[37] C. Stachniss, U. Frese, and G. Grisetti, "OpenSLAM." [Online]. Available: http://www.openslam.org/