

Rotation-Only Bundle Adjustment

Seong Hun Lee* Javier Civera
I3A, University of Zaragoza, Spain
{seonghunlee, jcivera}@unizar.es

Abstract

We propose a novel method for estimating the global rotations of the cameras independently of their positions and the scene structure. When two calibrated cameras observe five or more of the same points, their relative rotation can be recovered independently of the translation. We extend this idea to multiple views, thereby decoupling the rotation estimation from the translation and structure estimation. Our approach provides several benefits such as complete immunity to inaccurate translations and structure, and the accuracy improvement when used with rotation averaging. We perform extensive evaluations on both synthetic and real datasets, demonstrating consistent and significant gains in accuracy when used with the state-of-the-art rotation averaging method.

1. Introduction

Bundle adjustment is the problem of reconstructing the camera poses (*i.e.*, rotations and translations) and the 3D scene structure from the image measurements. It plays a crucial role in many areas of 3D vision, such as structure from motion [26], visual odometry [53], and simultaneous localization and mapping [8]. For this reason, significant research endeavors have been devoted to this problem, which led to tremendous progress over the past two decades.

Bundle adjustment aims to obtain jointly optimal structure and camera poses by minimizing the image reprojection errors [57]. Being a nonlinear optimization problem, it requires a good initialization to ensure the convergence to the statistically optimal solution [26]. A common strategy involves the following steps: (1) Estimate the pairwise motions. (2) Estimate the global rotations through rotation averaging (*e.g.*, [5, 10]). (3) Estimate the global translations (*e.g.*, [24, 59]). (4) Triangulate the points (*e.g.*, [32, 43]).

In such a pipeline, it is important to make an accurate initial guess of the rotations, as the subsequent steps directly depend on it. To this end, one could try to improve the rotation averaging method or its input (*i.e.*, the relative pairwise

motion estimates). Recent examples of the former include [5, 10, 16, 19, 51] and the latter include [6, 7, 23, 63].

These two types of approaches are certainly useful for initializing the rotations. However, relative pose estimation is limited to two views only, while rotation averaging does not directly leverage the image measurements. That is, it treats all relative rotations equally even if they were estimated from different numbers of points with different noise statistics and distributions. To our knowledge, no previous work has addressed this limitation for rotation estimation.

In this work, we present a novel method that, given the initial estimates of the rotations, performs rotation-only optimization using the image measurements as direct input. Our work is based on [34], where it was proposed to optimize the rotation between two views independently of the translation. We extend this idea to multiple views. We call our approach *rotation-only bundle adjustment* because it can be seen as the decoupling of the rotation estimation from the translation and structure estimation in bundle adjustment. This provides the following advantages:

- The rotations are estimated without requiring the knowledge of the translations and structure. This greatly simplifies the optimization problem.
- The rotations are immune to inaccurate estimation of the translations and structure.
- Both pure and non-pure rotations are treated in a unified manner, as we do not need to triangulate and discard the low-parallax points.
- It can be used after rotation averaging to improve the accuracy of the rotation estimates.

Table 1 summarizes the differences between our method and the related methods.

The paper is organized as follows: In the next two sections, we review the related work and the preliminaries. Section 4 reviews the two-view rotation-only method by Kneip and Lynen [34]. We describe our method in Section 5 and show the experimental results in Section 6. Finally, Section 7 and 8 present discussions and conclusions.

To download our code and the supplementary material, go to <https://seonghun-lee.github.io>.

*This work was partially supported by the Spanish govt. (PGC2018-096367-B-I00) and the Aragón regional govt. (DGA-T45.17R/FSE).

	Independent of the translations and the 3D scene structure?	Directly using the image measurements as input?	Applicable to n views?
Full bundle adjustment (e.g., [57])	✗	✓	✓
Rotation averaging (e.g., [10])	✓	✗	✓
Direct rotation optimization [34]	✓	✓	✗
Rotation-only bundle adjustment	✓	✓	✓

Table 1. Comparison between the related methods. To the best of our knowledge, we are the first to propose a multiview rotation-only optimization method using the image measurements as direct input. Our method can be generalized to both pure and non-pure rotations.

2. Related Work

Our work is related to several areas of study in 3D vision and robotics, namely structure from motion, simultaneous localization and mapping, bundle adjustment, rotation averaging, and relative pose estimation.

Structure from motion (SfM) is the problem of recovering the camera poses and the 3D scene from an unordered set of images. Large-scale systems may handle from hundreds of thousands [2] to millions of images [29, 64]. We refer to [50, 54] for excellent reviews of the SfM literature. The backbone of most SfM systems is bundle adjustment, the joint optimization of the camera poses and the 3D structure. To obtain the optimal solution, it requires good initial estimates of the poses and points [26]. In many works [4, 13, 14, 18, 24, 46, 47, 59, 66], the initialization consists of the following steps: (1) Estimate the relative poses between the camera pairs observing many points in common. (2) Perform multiple rotation averaging. (3) Estimate the camera locations (and the 3D points). In such a pipeline, one may use our method as Step 2.5 to refine the rotations.

Simultaneous localization and mapping (SLAM) is the problem of estimating the camera motion and the 3D scene in real time. Like SfM, the modern SLAM systems rely on bundle adjustment to jointly optimize the keyframe poses and the map points [9, 17, 40, 48]. Recently, in [11, 65], it was suggested that decoupling the rotation estimation through rotation averaging improves the efficiency and the handling of pure rotations.

Bundle adjustment is mainly classified into geometric and photometric methods. The former minimizes the reprojection errors [1, 26, 44, 57], and the latter minimizes the photometric errors [3, 15, 17, 60]. The bundle adjustment problems have been studied extensively for several decades, which led to diverse techniques for improving the scalability (e.g., [1, 36, 38, 44]) and the accuracy (e.g., [57, 61, 62]). In [30], an initialization-free approach was proposed. To our knowledge, however, no previous work has completely decoupled the rotation estimation in bundle adjustment.

Rotation averaging takes two forms: single rotation averaging that averages several estimates of a single rotation to obtain the best estimate [25, 42], and multiple rotation averaging that finds the multiple rotations \mathbf{R}_i given several noisy constraints on the relative rotations $\mathbf{R}_i \mathbf{R}_j^\top$

[4, 5, 10, 16, 19, 25, 46]. We refer to [28, 58] for an excellent tutorial and survey on the topic. As discussed earlier, multiple rotation averaging has wide application to SfM. This problem differs from bundle adjustment in that (1) only rotations are estimated, and (2) the input is the relative rotation estimates, not the image measurements. Since the states and the input are small compared to bundle adjustment, the computation process is faster and simpler. However, the downside is that it does not directly reflect the errors with respect to the image measurements. In contrast, our optimization method directly uses the image measurements as input, while maintaining only rotations in the state space. In this context, our method can be seen as the middle ground between rotation averaging and bundle adjustment.

Relative pose estimation Given a set of five or more point matches between two calibrated views, their relative pose can be obtained using a minimal method (e.g., [20, 31, 37, 49]) with RANSAC [6, 21, 52] or a non-minimal method (e.g., [7, 27, 34, 63]). In [34, 35], it was shown that the rotation can be estimated independently of the translation. In this work, we extend the idea of [34] to $n \geq 2$ views by aggregating multiple two-view costs and minimizing it through iterative nonlinear optimization.

3. Preliminaries and Notation

We use bold lowercase letters for vectors, bold uppercase letters for matrices, and light letters for scalars. We denote the Hadamard product, division and square root by $\mathbf{A} \circ \mathbf{B}$, $\mathbf{A} \oslash \mathbf{B}$ and $\mathbf{A}^{\circ 1/2}$, respectively. For a 3D vector \mathbf{v} , we define \mathbf{v}^\wedge as the corresponding 3×3 skew-symmetric matrix, and denote the inverse operator by $(\cdot)^\vee$, i.e., $(\mathbf{v}^\wedge)^\vee = \mathbf{v}$. The Euclidean norm of \mathbf{v} is denoted by $\|\mathbf{v}\|$, and its unit vector by $\hat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$. A rotation matrix $\mathbf{R} \in SO(3)$ can be represented by the corresponding rotation vector $\mathbf{u} = \theta \hat{\mathbf{u}}$, where θ and $\hat{\mathbf{u}}$ represent the angle and the unit axis of the rotation, respectively. The two representations are related by Rodrigues formula, and we denote the mapping between them by $\text{Exp}(\cdot)$ and $\text{Log}(\cdot)$ [22]:

$$\mathbf{R} = \text{Exp}(\mathbf{u}) := \mathbf{I} + \frac{\sin(\|\mathbf{u}\|)}{\|\mathbf{u}\|} \mathbf{u}^\wedge + \frac{1 - \cos(\|\mathbf{u}\|)}{\|\mathbf{u}\|^2} (\mathbf{u}^\wedge)^2, \quad (1)$$

$$\mathbf{u} = \text{Log}(\mathbf{R}) := \frac{\theta}{2 \sin(\theta)} (\mathbf{R} - \mathbf{R}^\top)^\vee \quad (2)$$

$$\text{with } \theta = \arccos((\text{tr}(\mathbf{R}) - 1)/2). \quad (3)$$

We denote the 3D position of a point with index i in a world reference frame w as $(\mathbf{x}_i)_w = [(x_i)_w, (y_i)_w, (z_i)_w]^\top$ and a perspective camera with index j as c_j . In the reference frame of c_j , the position of \mathbf{x}_i is given by $(\mathbf{x}_i)_j = [(x_i)_j, (y_i)_j, (z_i)_j]^\top = \mathbf{R}_j(\mathbf{x}_i)_w + \mathbf{t}_j$, where \mathbf{R}_j and \mathbf{t}_j are the rotation and translation that relate the local reference frame of c_j to the world. The projection of \mathbf{x}_i in the image plane of c_j has the pixel coordinates $[(u_i)_j, (v_i)_j]^\top = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{K}_j(\mathbf{f}_i)_j$, where \mathbf{K}_j is the camera calibration matrix of c_j and $(\mathbf{f}_i)_j = [(x_i)_j/(z_i)_j, (y_i)_j/(z_i)_j, 1]^\top$ is the normalized image coordinates of $(\mathbf{x}_i)_j$. Then, $(\mathbf{f}_i)_j$ can be obtained by $(\mathbf{f}_i)_j = \mathbf{K}_j^{-1}[(u_i)_j, (v_i)_j, 1]^\top$. We denote the rotation and the translation between camera j and k as \mathbf{R}_{jk} and \mathbf{t}_{jk} . The point \mathbf{x}_i in the reference frame of c_j and c_k is related by $(\mathbf{x}_i)_j = \mathbf{R}_{jk}(\mathbf{x}_i)_k + \mathbf{t}_{jk}$. This means that $\mathbf{R}_{jk} = \mathbf{R}_j\mathbf{R}_k^\top$ and $\mathbf{t}_{jk} = -\mathbf{R}_j\mathbf{R}_k^\top\mathbf{t}_k + \mathbf{t}_j$.

4. Review of Two-view Rotation-Only Method

In this section, we review the two-view rotation-only optimization method proposed by Kneip and Lynen in [34]. Consider two views with known internal calibration, c_j and c_k , observing $m \geq 5$ common points with index $i \in \{1, 2, \dots, m\}$. The normalized epipolar error [41] associated with each point i is defined as

$$(e_i)_{(j,k)} = \left| \hat{\mathbf{t}}_{jk} \cdot \left((\hat{\mathbf{f}}_i)_j \times \mathbf{R}_{jk}(\hat{\mathbf{f}}_i)_k \right) \right|, \quad (4)$$

where $(\hat{\mathbf{f}}_i)_j$ and $(\hat{\mathbf{f}}_i)_k$ are the unit bearing vectors corresponding to the i th point in c_j and c_k , respectively. The sum of squares of all these errors is then given by

$$\sum_{i=1}^m (e_i)_{(j,k)}^2 = \hat{\mathbf{t}}_{jk}^\top \mathbf{M}_{jk} \hat{\mathbf{t}}_{jk}, \quad (5)$$

where

$$\mathbf{M}_{jk} = \sum_{i=1}^m \left((\hat{\mathbf{f}}_i)_j \times \mathbf{R}_{jk}(\hat{\mathbf{f}}_i)_k \right) \left((\hat{\mathbf{f}}_i)_j \times \mathbf{R}_{jk}(\hat{\mathbf{f}}_i)_k \right)^\top. \quad (6)$$

In [34], it was shown that the 3×3 matrix \mathbf{M}_{jk} can also be computed as follows: denoting the entries of $(\hat{\mathbf{f}}_i)_j$ as $[(f_{xi})_j, (f_{yi})_j, (f_{zi})_j]^\top$, the following matrices are defined:

$$(\mathbf{F}_{xx})_{jk} = \sum_{i=1}^m (f_{xi})_j^2 (\hat{\mathbf{f}}_i)_k (\hat{\mathbf{f}}_i)_k^\top, \quad (7)$$

$$(\mathbf{F}_{xy})_{jk} = \sum_{i=1}^m (f_{xi})_j (f_{yi})_j (\hat{\mathbf{f}}_i)_k (\hat{\mathbf{f}}_i)_k^\top, \quad (8)$$

$$(\mathbf{F}_{xz})_{jk} = \sum_{i=1}^m (f_{xi})_j (f_{zi})_j (\hat{\mathbf{f}}_i)_k (\hat{\mathbf{f}}_i)_k^\top, \quad (9)$$

$$(\mathbf{F}_{yy})_{jk} = \sum_{i=1}^m (f_{yi})_j^2 (\hat{\mathbf{f}}_i)_k (\hat{\mathbf{f}}_i)_k^\top, \quad (10)$$

$$(\mathbf{F}_{yz})_{jk} = \sum_{i=1}^m (f_{yi})_j (f_{zi})_j (\hat{\mathbf{f}}_i)_k (\hat{\mathbf{f}}_i)_k^\top. \quad (11)$$

Let $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ be each row of \mathbf{R}_{jk} , and m_{ab} be the element of \mathbf{M}_{jk} at the a th row and b th column. (Notice that we

omitted the subscript jk here for simplicity). Then,

$$m_{11} = \mathbf{r}_3 \mathbf{F}_{yy} \mathbf{r}_3^\top - 2\mathbf{r}_3 \mathbf{F}_{yz} \mathbf{r}_2^\top + \mathbf{r}_2 \mathbf{F}_{zz} \mathbf{r}_2^\top, \quad (12)$$

$$m_{22} = \mathbf{r}_1 \mathbf{F}_{zz} \mathbf{r}_1^\top - 2\mathbf{r}_1 \mathbf{F}_{xz} \mathbf{r}_3^\top + \mathbf{r}_3 \mathbf{F}_{xx} \mathbf{r}_3^\top, \quad (13)$$

$$m_{33} = \mathbf{r}_2 \mathbf{F}_{xx} \mathbf{r}_2^\top - 2\mathbf{r}_1 \mathbf{F}_{xy} \mathbf{r}_2^\top + \mathbf{r}_1 \mathbf{F}_{yy} \mathbf{r}_1^\top, \quad (14)$$

$$m_{12} = \mathbf{r}_1 \mathbf{F}_{yz} \mathbf{r}_3^\top - \mathbf{r}_1 \mathbf{F}_{zz} \mathbf{r}_2^\top - \mathbf{r}_3 \mathbf{F}_{xy} \mathbf{r}_3^\top + \mathbf{r}_3 \mathbf{F}_{xz} \mathbf{r}_2^\top, \quad (15)$$

$$m_{13} = \mathbf{r}_2 \mathbf{F}_{xy} \mathbf{r}_3^\top - \mathbf{r}_2 \mathbf{F}_{xz} \mathbf{r}_2^\top - \mathbf{r}_1 \mathbf{F}_{yy} \mathbf{r}_3^\top + \mathbf{r}_1 \mathbf{F}_{yz} \mathbf{r}_2^\top, \quad (16)$$

$$m_{23} = \mathbf{r}_1 \mathbf{F}_{xz} \mathbf{r}_2^\top - \mathbf{r}_1 \mathbf{F}_{yz} \mathbf{r}_1^\top - \mathbf{r}_3 \mathbf{F}_{xx} \mathbf{r}_2^\top + \mathbf{r}_3 \mathbf{F}_{xy} \mathbf{r}_1^\top, \quad (17)$$

and $m_{21} = m_{12}$, $m_{31} = m_{13}$, $m_{32} = m_{23}$. This is a more efficient computation of \mathbf{M}_{jk} than (6), as (7)–(11) can be precomputed regardless of \mathbf{R}_{jk} , reducing the number of operations during the rotation optimization [34].

Given the set of corresponding unit bearing vectors, one can jointly optimize the relative rotation and translation by minimizing (5) with respect to \mathbf{R}_{jk} and $\hat{\mathbf{t}}_{jk}$. In [34], it was shown that this problem can be transformed into a rotation-only form:

$$\mathbf{R}_{jk}^* = \underset{\mathbf{R}_{jk}}{\operatorname{argmin}} \lambda_{\mathbf{M}}(\mathbf{R}_{jk}), \quad (18)$$

where $\lambda_{\mathbf{M}}(\mathbf{R}_{jk})$ is the smallest eigenvalue of \mathbf{M}_{jk} (which is a function of \mathbf{R}_{jk}). This eigenvalue can be obtained in closed form [34]:

$$b_1 = -m_{11} - m_{22} - m_{33}, \quad (19)$$

$$b_2 = -m_{13}^2 - m_{23}^2 - m_{12}^2 + m_{11}m_{22} + m_{11}m_{33} + m_{22}m_{33}, \quad (20)$$

$$b_3 = m_{22}m_{13}^2 + m_{11}m_{23}^2 + m_{33}m_{12}^2 - m_{11}m_{22}m_{33} - 2m_{12}m_{23}m_{13}, \quad (21)$$

$$s = 2b_1^3 - 9b_1b_2 + 27b_3, \quad (22)$$

$$t = 4(b_1^2 - 3b_2)^3, \quad (23)$$

$$k = \left(\sqrt{t}/2 \right)^{1/3} \cos \left(\arccos \left(s/\sqrt{t} \right) / 3 \right), \quad (24)$$

$$\lambda_{\mathbf{M}}(\mathbf{R}_{jk}) = (-b_1 - 2k) / 3. \quad (25)$$

To summarize, the rotation part of the optimal solution $(\mathbf{R}_{jk}^*, \hat{\mathbf{t}}_{jk}^*)$ that minimizes (5) is obtained by solving (18).

5. Rotation-Only Bundle Adjustment

5.1. Cost function

We extend the idea of [34] for n views. Let \mathcal{E} be the set of all edges, *i.e.*, camera pairs (j, k) observing a sufficient number of points in common (> 10 in our implementation). Then, we formulate the optimization problem as follows:

$$\{\mathbf{R}_1^*, \dots, \mathbf{R}_n^*\} = \underset{\mathbf{R}_1, \dots, \mathbf{R}_n}{\operatorname{argmin}} C(\mathbf{R}_1, \dots, \mathbf{R}_n) \quad (26)$$

with

$$C(\mathbf{R}_1, \dots, \mathbf{R}_n) = \sum_{(j,k) \in \mathcal{E}} \underbrace{\sqrt{\lambda_{\mathbf{M}}(\mathbf{R}_{jk})}}_{c_{jk}}, \quad (27)$$

where $\lambda_{\mathbf{M}}(\mathbf{R}_{jk})$ is the same cost function used in (18) for the two-view case and $c_{jk} = \sqrt{\lambda_{\mathbf{M}}(\mathbf{R}_{jk})}$ is our edge cost. We empirically found that this square rooting improves the convergence rate (see Table 5), which we presume is due to the downweighted influence of outliers. Alg. 1 summarizes the steps for computing the edge cost c_{jk} .

5.2. Optimization

To solve (26) iteratively, we use Adam [33], a first-order gradient-based optimization algorithm for stochastic objective functions. Adam has been widely used in deep learning, and we found that it also works well for our geometric optimization problem. Given the initial estimates of $\mathbf{R}_1, \dots, \mathbf{R}_n$, let \mathbf{s}_0 be the initial state vector formed by stacking $\text{Log}(\mathbf{R}_1), \dots, \text{Log}(\mathbf{R}_n)$ in one column. Let $\mathbf{m}_0 = \mathbf{0}_{3n \times 1}$, $\mathbf{v}_0 = \mathbf{0}_{3n \times 1}$, $t = 0$ and $\epsilon = (10^{-8})\mathbf{1}_{3n \times 1}$. Then, using Adam, we repeat the following steps at each iteration t of our optimization:

$$t \leftarrow t + 1, \quad (28)$$

$$\mathbf{g}_t \leftarrow \nabla_{\mathbf{s}} C(\mathbf{R}_1, \dots, \mathbf{R}_n), \quad (29)$$

$$\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{g}_t, \quad (30)$$

$$\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) (\mathbf{g}_t \circ \mathbf{g}_t), \quad (31)$$

$$\mathbf{m}'_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t), \quad (32)$$

$$\mathbf{v}'_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t), \quad (33)$$

$$\mathbf{s}_t \leftarrow \mathbf{s}_{t-1} - \alpha \mathbf{m}'_t \odot (\mathbf{v}'_t \circ \mathbf{1}^{1/2} + \epsilon), \quad (34)$$

$$\mathbf{u}_i \leftarrow [(\mathbf{s}_t)_{3i-2}, (\mathbf{s}_t)_{3i-1}, (\mathbf{s}_t)_{3i}]^T \text{ for } i = 1, \dots, n, \quad (35)$$

$$\mathbf{R}_i \leftarrow \text{Exp}(\mathbf{u}_i) \text{ for } i = 1, \dots, n. \quad (36)$$

We detail the computation of the gradient (*i.e.*, (29)) in the next section. For the hyper-parameters β_1 and β_2 , we use the default values given in [33]: $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For the step size α , we use $\alpha = 0.01$ at the beginning and switch to $\alpha = 0.001$ permanently once the cost increases in five successive iterations. We empirically found that this switching sometimes helps the convergence (see Table 5). Alg. 2 summarizes our method.

5.3. Gradient computation

We compute the gradient \mathbf{g}_t in (29) numerically¹. This can be done efficiently by slightly perturbing each rotation parameter in \mathbf{s}_t and summing the resulting changes of all the edge costs c_{jk} (27) as we traverse the edge set \mathcal{E} . Since we need to run Alg. 1 seven times for each edge (*i.e.*, 1 from the unperturbed state, 3×2 from perturbing \mathbf{R}_j and \mathbf{R}_k), if there are $n_{\mathcal{E}}$ edges, this method will require $7n_{\mathcal{E}}$ computations of edge costs. To reduce the computation time, we

¹It is possible to compute it analytically. However, the closed-form expressions involve more operations than the numerical method (see the supplementary material of [34]). We empirically found that this takes approximately 1.8 times longer, while the numerical difference is negligible.

Algorithm 1: Computation of edge cost c_{jk}

Input: Relative rotation \mathbf{R}_{jk} , Precomputed matrices $(\mathbf{F}_{xx})_{jk}, (\mathbf{F}_{xy})_{jk}, (\mathbf{F}_{xz})_{jk}, (\mathbf{F}_{yy})_{jk}, (\mathbf{F}_{yz})_{jk}$.

Output: Edge cost c_{jk} .

- 1 $\mathbf{r}_i \leftarrow (\mathbf{R}_{jk})_{i\text{th row}}$ for $i = 1, 2, 3$;
 - 2 compute m_{11}, \dots, m_{23} using (12)–(17);
 - 3 compute $\lambda_{\mathbf{M}}(\mathbf{R}_{jk})$ using (19)–(25);
 - 4 $c_{jk} \leftarrow \sqrt{\lambda_{\mathbf{M}}(\mathbf{R}_{jk})}$;
 - 5 **return** c_{jk} ;
-

Algorithm 2: Rotation-Only Bundle Adjustment

Input: Initial rotations $\mathbf{R}_1, \dots, \mathbf{R}_n$, Edges \mathcal{E} , Matched unit bearing vectors $((\hat{\mathbf{f}}_i)_j, (\hat{\mathbf{f}}_i)_k)$ for all $i \in 1, \dots, m_{(j,k)}$ for all $(j, k) \in \mathcal{E}$, Number of iterations n_{it} .

Output: Final rotations $\mathbf{R}_1, \dots, \mathbf{R}_n$.

- ```

/* Initialization: */
1 compute $(\mathbf{F}_{xx})_{jk}, \dots, (\mathbf{F}_{yz})_{jk}$ for all $(j, k) \in \mathcal{E}$
 using (7)–(11);
2 compute C using (27) and Alg. 1;
3 obtain \mathbf{s}_0 by stacking $\text{Log}(\mathbf{R}_1), \dots, \text{Log}(\mathbf{R}_n)$ in
 one column;
4 $\beta_1 \leftarrow 0.9$; $\beta_2 \leftarrow 0.999$; $\alpha \leftarrow 0.01$; $\epsilon \leftarrow (10^{-8})\mathbf{1}_{3n \times 1}$;
5 $\mathbf{m}_0 \leftarrow \mathbf{0}_{3n \times 1}$; $\mathbf{v}_0 \leftarrow \mathbf{0}_{3n \times 1}$; $t \leftarrow 0$;
/* Optimization: */
6 while $t < n_{it}$ do
7 $t \leftarrow t + 1$;
8 compute C and \mathbf{g}_t using Alg. 3;
9 Perform (30)–(36);
10 if C increased in five successive iterations then
11 $\alpha \leftarrow 0.001$;
12 return $\mathbf{R}_1, \dots, \mathbf{R}_n$;

```
- 

make the following approximation:

$$c_{jk} (\mathbf{R}_j(\mathbf{R}_k)_{x+\Delta x}^\top - c_{jk}(\mathbf{R}_j \mathbf{R}_k^\top)) \approx c_{jk}(\mathbf{R}_j \mathbf{R}_k^\top) - c_{jk}((\mathbf{R}_j)_{x+\Delta x} \mathbf{R}_k^\top), \quad (37)$$

where  $(\mathbf{R}_j)_{x+\Delta x}$  and  $(\mathbf{R}_k)_{x+\Delta x}$  respectively denote  $\mathbf{R}_j$  and  $\mathbf{R}_k$  after being perturbed (by the same magnitude) in the  $x$  component of the rotation vector. That is, we assume that  $\Delta c_{jk}$  due to  $(\mathbf{R}_k)_{x+\Delta x}$  is approximately equal to the negative of  $\Delta c_{jk}$  due to  $(\mathbf{R}_j)_{x+\Delta x}$ . We make analogous approximations for the perturbations in the  $y$  and  $z$  component of the rotation vector. By approximating the gradient of  $\mathbf{R}_k$  using that of  $\mathbf{R}_j$ , we reduce the number of edge cost computations from  $7n_{\mathcal{E}}$  to  $4n_{\mathcal{E}}$ . Empirically, we found that this improves the efficiency significantly at a relatively small loss of accuracy (see Table 6). Alg. 3 summarizes the procedure for computing the gradient and the total cost.

---

**Algorithm 3: Cost and gradient computation**

---

**Input:** Current rotations  $\mathbf{R}_1, \dots, \mathbf{R}_n$ , Edges  $\mathcal{E}$ ,  
 $(\mathbf{F}_{xx})_{jk}, (\mathbf{F}_{xy})_{jk}, (\mathbf{F}_{xz})_{jk}, (\mathbf{F}_{yy})_{jk}, (\mathbf{F}_{yz})_{jk}$   
for all  $(j, k) \in \mathcal{E}$

**Output:** Cost  $C$ , Gradient  $\mathbf{g}$ .

```
1 $C \leftarrow 0$; $\mathbf{g} \leftarrow \mathbf{0}_{3n \times 1}$; $\delta \leftarrow 10^{-4}$;
2 $\delta_x \leftarrow [\delta, 0, 0]^\top$; $\delta_y \leftarrow [0, \delta, 0]^\top$; $\delta_z \leftarrow [0, 0, \delta]^\top$;
3 $\mathbf{u}_i \leftarrow \text{Log}(\mathbf{R}_i)$ for all $i \in 1, \dots, n$;
 /* Perturb the rotations: */
4 $(\mathbf{R}_i)_{x+\Delta x} \leftarrow \text{Exp}(\mathbf{u}_i + \delta_x)$ for all $i \in 1, \dots, n$;
5 $(\mathbf{R}_i)_{y+\Delta y} \leftarrow \text{Exp}(\mathbf{u}_i + \delta_y)$ for all $i \in 1, \dots, n$;
6 $(\mathbf{R}_i)_{z+\Delta z} \leftarrow \text{Exp}(\mathbf{u}_i + \delta_z)$ for all $i \in 1, \dots, n$;
 /* Sum the resulting changes of each c_{jk} : */
7 for $(j, k) \in \mathcal{E}$ do
8 $\mathbf{R}_{jk} \leftarrow \mathbf{R}_j \mathbf{R}_k^\top$;
9 $(\mathbf{R}_{jk})_{x+\Delta x} \leftarrow (\mathbf{R}_j)_{x+\Delta x} \mathbf{R}_k^\top$;
10 $(\mathbf{R}_{jk})_{y+\Delta y} \leftarrow (\mathbf{R}_j)_{y+\Delta y} \mathbf{R}_k^\top$;
11 $(\mathbf{R}_{jk})_{z+\Delta z} \leftarrow (\mathbf{R}_j)_{z+\Delta z} \mathbf{R}_k^\top$;
12 obtain c_{jk} using Alg. 1 with \mathbf{R}_{jk} .
13 obtain $(c_{jk})_{x+\Delta x}$ using Alg. 1 with $(\mathbf{R}_{jk})_{x+\Delta x}$.
14 obtain $(c_{jk})_{y+\Delta y}$ using Alg. 1 with $(\mathbf{R}_{jk})_{y+\Delta y}$.
15 obtain $(c_{jk})_{z+\Delta z}$ using Alg. 1 with $(\mathbf{R}_{jk})_{z+\Delta z}$.
16 $\Delta(c_{jk})_x \leftarrow (c_{jk})_{x+\Delta x} - c_{jk}$;
17 $\Delta(c_{jk})_y \leftarrow (c_{jk})_{y+\Delta y} - c_{jk}$;
18 $\Delta(c_{jk})_z \leftarrow (c_{jk})_{z+\Delta z} - c_{jk}$;
19 $\mathbf{g}_{3j-2} \leftarrow \mathbf{g}_{3j-2} + \Delta(c_{jk})_x$;
20 $\mathbf{g}_{3j-1} \leftarrow \mathbf{g}_{3j-1} + \Delta(c_{jk})_y$;
21 $\mathbf{g}_{3j} \leftarrow \mathbf{g}_{3j} + \Delta(c_{jk})_z$;
22 $\mathbf{g}_{3k-2} \leftarrow \mathbf{g}_{3k-2} - \Delta(c_{jk})_x$;
23 $\mathbf{g}_{3k-1} \leftarrow \mathbf{g}_{3k-1} - \Delta(c_{jk})_y$;
24 $\mathbf{g}_{3k} \leftarrow \mathbf{g}_{3k} - \Delta(c_{jk})_z$;
25 $C \leftarrow C + c_{jk}$;
26 $\mathbf{g} \leftarrow \mathbf{g}/\delta$;
27 return C and \mathbf{g} ;
```

---

## 6. Results

### 6.1. Evaluation method

We compare our method (henceforth ROBA) against the state-of-the-art rotation averaging method by Chatterjee and Govindu [10] (henceforth RA). Both methods are implemented in MATLAB and run on a laptop CPU (Intel i7-4710MQ, 2.8GHz). For RA, we use the code publicly shared by the authors of [10]<sup>2</sup> with the  $L_{\frac{1}{2}}$  loss function, as recommended in [10]. We use the output of RA as input to ROBA, so that we can compare RA versus RA + ROBA.

In Alg. 2, the bottleneck is the gradient computation (line 8), where the predominant part is the computation of the edge costs using Alg. 1. To speed up this part, we im-

<sup>2</sup><http://www.ee.iisc.ac.in/labs/cvl/research/rotaveraging/>

plement Alg. 1 in a C++ MEX function. We set the number of iterations ( $n_{it}$ ) to 100 in Alg. 2. Note that, in practice, it would be sensible to adopt some stopping criteria to detect the convergence (e.g., based on the relative change of the total cost or the angular change in the rotations). In our experiment, however, we aim to investigate the convergence behavior of ROBA, so we are agnostic about this heuristics.

Finally, we draw attention to the error metrics for evaluating the rotation estimates  $(\mathbf{R}_1, \dots, \mathbf{R}_n)$ . Since they do not share the same reference frame as the ground-truth rotations  $(\mathbf{R}_1^{\text{gt}}, \dots, \mathbf{R}_n^{\text{gt}})$ , we must first align them with the ground truth to evaluate the accuracy. Commonly, this is done by rotating them with one of the following rotations:

$$\mathbf{R}_{L1} = \underset{\mathbf{R}_{L1}}{\text{argmin}} \sum_{j=1}^n d(\mathbf{R}_{L1}, \mathbf{R}_j^\top \mathbf{R}_j^{\text{gt}}), \quad (38)$$

$$\mathbf{R}_{L2} = \underset{\mathbf{R}_{L2}}{\text{argmin}} \sum_{j=1}^n d(\mathbf{R}_{L2}, \mathbf{R}_j^\top \mathbf{R}_j^{\text{gt}})^2, \quad (39)$$

where  $d(\cdot, \cdot)$  denotes the geodesic distance between the two rotations, i.e.,  $d(\mathbf{R}_1, \mathbf{R}_2) = \arccos((\text{tr}(\mathbf{R}_1 \mathbf{R}_2^\top) - 1)/2)$ . Note that (38) and (39) are single rotation averaging problems, and they can be solved using iterative algorithms [25, 28]. Afterwards, we rotate the estimates as follows<sup>3</sup>:

$$\mathbf{R}_j \leftarrow \mathbf{R}_j \mathbf{R}_{L1}, \quad \text{or} \quad \mathbf{R}_j \leftarrow \mathbf{R}_j \mathbf{R}_{L2}. \quad (40)$$

Since  $\mathbf{R}_{L1}$  minimizes the sum of absolute distances and  $\mathbf{R}_{L2}$  minimizes the sum of squares, we call the first method  $L_1$  alignment and the second  $L_2$  alignment. In our evaluation, we report the mean and median angular errors using these two alignment methods.

### 6.2. Synthetic data

To study how different factors affect our method, we run Monte Carlo simulations in controlled settings: we uniformly distribute  $n$  cameras on a circle on the xy-plane such that the neighbors are 1 unit apart. After aligning their optical axes with the z-axis, we perturb the rotations by random angles  $\theta \sim \mathcal{U}(0, 20^\circ)$ . We set the image size to  $640 \times 480$  pixels and the focal length to 525 pixels, the same as those in [56]. We create 3D points at random distances  $d \sim \mathcal{U}(d_{\min}, d_{\max})$  from the xy-plane, ensuring that every neighboring view observes at least  $n_{\text{cov}}$  points in common. We perturb the image coordinates of the points by  $\mathcal{N}(0, \sigma^2)$ . For every pair of views observing  $n_{\text{cov}}$  or more points in common, we estimate the relative pose and the inlying points. If there are at least 10 inliers, we add the pair as an edge in  $\mathcal{E}$ . Table 2 specifies the configuration parameters we set for our simulations. For each parameter setting, we generate 100 different datasets, each with randomly sampled camera rotations, 3D points and 2D measurements.

<sup>3</sup>We must use right multiplication here in order to make sure that  $(\mathbf{R}_{12})_{\text{after}} = (\mathbf{R}_1 \mathbf{R}_{L1})(\mathbf{R}_2 \mathbf{R}_{L1})^\top = \mathbf{R}_1 \mathbf{R}_2^\top = (\mathbf{R}_{12})_{\text{before}}$ .

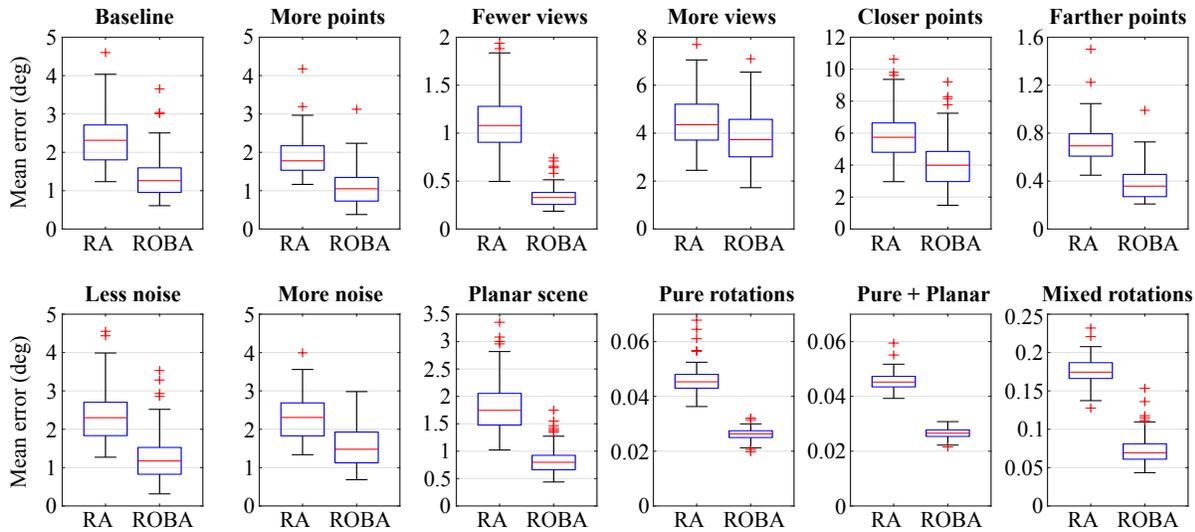


Figure 1. Results on the synthetic data (see Table 2 for the settings). We compare RA [10] and ROBA (initialized by RA) in terms of the mean angular error after the  $L_1$  alignment (see the supplementary material for the other error metrics). It shows that ROBA improves the results of RA in all scenarios considered. In particular, the relative error reduction is large for fewer views, farther points and pure rotations, all of which lead to a denser view-graph (see Table 3).

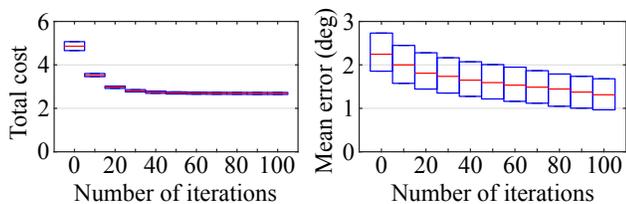


Figure 2. Evolution of the cost function (left) and the mean angular errors after the  $L_1$  alignment (right) in the baseline setting. Here, we only show the interquartile range. Notice that while our cost (27) seems to plateau after 30–40 iterations, the actual rotation errors continue to decrease. This is discussed in Section 7.2.

To obtain the relative rotation estimates, we use the following method: first, we obtain 100 pose samples around the ground-truth relative pose. We do this by perturbing the rotation and the translation by two arbitrary angles ( $< 20$  deg). Then, we evaluate the  $L_1$ -optimal angular reprojection errors [39]<sup>4</sup> for each pose and choose the one that yields the most inliers. This method is similar to the standard method of using a minimal solver (e.g., five-point algorithm [49]) in RANSAC [21], except that our samples are simulated, not estimated. We use this method because our focus is on the optimization of the rotations and we are agnostic about the relative pose estimation method.

Fig. 1 and Table 3 present the results in each setting. Notice that ROBA improves the results of RA in all scenarios considered. In Fig. 2, we show the evolution of our cost function (27) and the rotation error in the baseline setting.

<sup>4</sup>This can be computed using (4) and their relation derived in [41]. We do not consider chirality [26], because otherwise we would end up discarding many inlying low-parallax points that appear in pure rotations.

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Baseline setting | $n = 100, n_{\text{cov}} = 50, \sigma = 1\text{px}, d_{\text{min}} = 2, d_{\text{max}} = 5,$<br>Views are uniformly spaced by 1 unit on a circle.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Variations       | <b>More points</b> ( $n_{\text{cov}} = 100$ ), <b>Fewer views</b> ( $n = 30$ ),<br><b>More views</b> ( $n = 300$ ), <b>Closer points</b> ( $d_{\text{max}} = 3$ ),<br><b>Farther points</b> ( $d_{\text{max}} = 10$ ), <b>Less noise</b> ( $\sigma = 0.5\text{px}$ ),<br><b>More noise</b> ( $\sigma = 2\text{px}$ ), <b>Planar scene</b> ( $d_{\text{min}} = 5$ ).<br><b>Pure rotations</b> (all views are placed at the origin),<br><b>Pure rotations + Planar scene</b> ( $d_{\text{min}} = 5$ ),<br><b>Mixed rotations</b> (20 groups of views are uniformly spaced by 1 unit on a circle. A group consists of five views at the same location). |

Table 2. Simulation settings.

| Settings        | $\%_{\mathcal{E}}$ | $\bar{\epsilon}_{\mathcal{E}}$ | $\tilde{\epsilon}_{\mathcal{E}}$ | RA [10] | RA [10]<br>+ ROBA | $\%_{\text{better}}$ |
|-----------------|--------------------|--------------------------------|----------------------------------|---------|-------------------|----------------------|
| Baseline        | 6.0%               | 2.21                           | 1.39                             | 2.31    | <b>1.26</b>       | 100%                 |
| More points     | 7.2%               | 2.51                           | 1.52                             | 1.78    | <b>1.05</b>       | 100%                 |
| Fewer views     | 21%                | 2.23                           | 1.38                             | 1.08    | <b>0.33</b>       | 100%                 |
| More views      | 2.0%               | 2.24                           | 1.40                             | 4.35    | <b>3.73</b>       | 100%                 |
| Closer points   | 4.0%               | 2.57                           | 1.65                             | 5.74    | <b>3.99</b>       | 100%                 |
| Farther points  | 10%                | 1.97                           | 1.22                             | 0.69    | <b>0.36</b>       | 100%                 |
| Less noise      | 6.0%               | 2.25                           | 1.40                             | 2.30    | <b>1.17</b>       | 100%                 |
| More noise      | 5.9%               | 2.23                           | 1.38                             | 2.31    | <b>1.48</b>       | 100%                 |
| Planar scene    | 7.4%               | 2.61                           | 1.56                             | 1.74    | <b>0.80</b>       | 100%                 |
| Pure rotations  | 100%               | 0.89                           | 0.74                             | 0.045   | <b>0.026</b>      | 100%                 |
| Pure + Planar   | 100%               | 0.89                           | 0.74                             | 0.045   | <b>0.027</b>      | 100%                 |
| Mixed rotations | 37%                | 3.13                           | 1.66                             | 0.17    | <b>0.069</b>      | 100%                 |

$\%_{\mathcal{E}}, \%_{\text{better}}$ : proportion of existing edges and improved results,  
 $\bar{\epsilon}_{\mathcal{E}}, \tilde{\epsilon}_{\mathcal{E}}$ : mean and median angular errors (in deg) of the relative rotations from all edges.

Table 3. Median results of the 100 simulations in each setting. The 5th and 6th columns give the median errors shown in Fig. 1. Overall, the denser the view-graph, the more accurate both methods are. ROBA improves upon RA in 1200 out of 1200 simulations.

| Name | Datasets<br>( $n, n_{\mathcal{E}}, \%\mathcal{E}$ ) | RA [10] |      |      |      | RA [10] + ROBA (100 iter) |             |             |             | Computation time (s) |      |      |       |
|------|-----------------------------------------------------|---------|------|------|------|---------------------------|-------------|-------------|-------------|----------------------|------|------|-------|
|      |                                                     | mn1     | md1  | mn2  | md2  | mn1                       | md1         | mn2         | md2         | RA [10]              | Init | Opti | Total |
| ALM  | (577, 96653, 58%)                                   | 4.08    | 1.11 | 4.74 | 2.17 | <b>2.35</b>               | <b>0.42</b> | <b>3.20</b> | <b>1.47</b> | 16                   | 35   | 216  | 267   |
| ELS  | (227, 18709, 73%)                                   | 2.10    | 0.50 | 2.37 | 0.93 | <b>1.06</b>               | <b>0.10</b> | <b>1.48</b> | <b>0.57</b> | 1                    | 5    | 42   | 48    |
| GDM  | (677, 33662, 15%)                                   | 6.05    | 2.78 | 6.14 | 3.12 | <b>2.43</b>               | <b>1.13</b> | <b>2.44</b> | <b>1.15</b> | 3                    | 8    | 76   | 87    |
| MDR  | (341, 23228, 40%)                                   | 6.20    | 1.27 | 7.23 | 2.88 | <b>4.42</b>               | <b>0.61</b> | <b>5.71</b> | <b>2.50</b> | 2                    | 7    | 52   | 61    |
| MND  | (480, 51172, 45%)                                   | 1.46    | 0.51 | 1.59 | 0.72 | <b>0.82</b>               | <b>0.26</b> | <b>1.01</b> | <b>0.50</b> | 4                    | 24   | 114  | 142   |
| NTD  | (553, 96672, 63%)                                   | 2.08    | 0.64 | 2.31 | 0.88 | <b>1.27</b>               | <b>0.30</b> | <b>1.47</b> | <b>0.57</b> | 14                   | 60   | 217  | 291   |
| NYC  | (332, 18787, 34%)                                   | 2.87    | 1.32 | 2.99 | 1.40 | <b>1.03</b>               | <b>0.20</b> | <b>1.20</b> | <b>0.45</b> | 1                    | 6    | 42   | 49    |
| PDP  | (338, 24121, 42%)                                   | 3.86    | 0.91 | 4.67 | 2.48 | <b>2.18</b>               | <b>0.33</b> | <b>2.92</b> | <b>1.64</b> | 1                    | 6    | 54   | 61    |
| PIC  | (2151, 275895, 12%)                                 | 4.14    | 2.28 | 4.18 | 2.38 | <b>1.58</b>               | <b>0.29</b> | <b>1.75</b> | <b>0.49</b> | 220                  | 62   | 617  | 899   |
| ROF  | (1083, 68379, 12%)                                  | 2.94    | 1.52 | 2.99 | 1.56 | <b>2.18</b>               | <b>0.27</b> | <b>2.28</b> | <b>0.61</b> | 6                    | 26   | 154  | 186   |
| TOL  | (472, 23379, 21%)                                   | 3.83    | 2.33 | 3.85 | 2.38 | <b>1.15</b>               | <b>0.16</b> | <b>1.20</b> | <b>0.24</b> | 1                    | 10   | 53   | 64    |
| TFG  | (5057, 663755, 5%)                                  | 3.40    | 2.34 | 3.42 | 2.28 | <b>2.76</b>               | <b>2.09</b> | <b>2.78</b> | <b>1.79</b> | 553                  | 153  | 1488 | 2194  |
| USQ  | (787, 23639, 8%)                                    | 5.59    | 4.03 | 5.60 | 4.06 | <b>3.26</b>               | <b>0.90</b> | <b>3.46</b> | <b>1.26</b> | 1                    | 7    | 54   | 62    |
| VNC  | (836, 98999, 28%)                                   | 6.12    | 1.33 | 7.96 | 4.06 | <b>4.96</b>               | <b>0.25</b> | <b>7.31</b> | <b>3.47</b> | 15                   | 53   | 221  | 289   |
| YKM  | (437, 27039, 28%)                                   | 3.67    | 1.60 | 3.72 | 1.61 | <b>1.66</b>               | <b>0.19</b> | <b>1.74</b> | <b>0.28</b> | 1                    | 12   | 61   | 74    |

$n$ : # connected views with known ground truth,  $n_{\mathcal{E}}$ : # edges with at least 10 covisible 3D points,  $\%\mathcal{E} = n_{\mathcal{E}}/nC2$  in %, mn/md/1/2: mean/median angular error (in deg) after the  $L_1/L_2$  alignment, Init: Initialization (line 1–5 of Alg. 2), Opti: Optimization (line 6–11 of Alg. 2).

Table 4. Results on the real data [59]. For all datasets, ROBA improves the results of RA [10]. This is shown across all error criteria, and often, the relative error reduction is significant. See the supplementary material for the evolution of the total cost (27) and the errors.

### 6.3. Real data

We also perform an evaluation on the real-world datasets publicly shared by the authors of [59]<sup>5</sup>, which include:

- internal camera calibration and radial distortion data,
- SIFT [45] feature tracks and their image coordinates,
- estimated relative rotations,
- reconstruction made with Bundler [55], consisting of the camera poses and a sparse set of 3D points.<sup>6</sup>

As in [42], we use the provided reconstruction model as the ground truth in our experiment. We undistort all image measurements when we process the input. Additionally, some preprocessing is required for these datasets because (1) they do not provide the SIFT IDs of the reconstructed 3D points, and (2) some edges (*i.e.*, camera pairs for which the estimated relative rotations are given) lack covisible 3D points because some of the tracks were disregarded during the reconstruction process (*e.g.*, outliers and low-parallax points).

We address the first issue by projecting each 3D point in its associated images and finding the track that yields the smallest mean reprojection error. Since most reconstructed points have small reprojection errors (<1–2 pixels), this can associate the points quite accurately, which we verified qualitatively. We address the second issue by removing all edges with less than 10 covisible 3D points. As a result, some cameras get disconnected from the main view-graph. In our experiment, we disregard all such cameras, as well as those without the available ground truth.

<sup>5</sup><http://www.cs.cornell.edu/projects/ldsfm/>

<sup>6</sup>The pose estimates are not available for some of the cameras. Also, some of the SIFT features are not associated with the available 3D points.

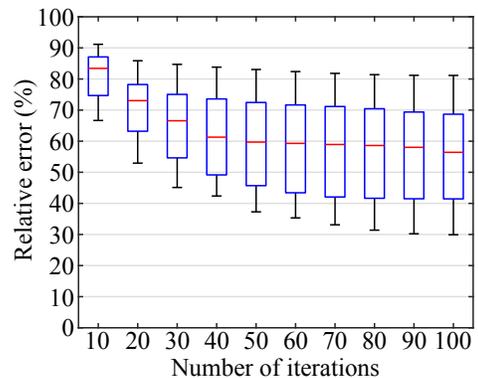


Figure 3. Relative errors with respect to the initial values for the real-world datasets [59]. Here, we use the mean angular errors after the  $L_1$  alignment (see the supplementary material for the other metrics). For more than half of the datasets, the error decreases to less than 60% of the initial value after 50 iterations. After 100 iterations, we observe error reductions up to 70%.

Table 4 presents the results. It shows that ROBA offers a consistent and significant gain in accuracy. In Fig. 3, we plot the evolution of the relative errors aggregated from all datasets. Table 5 and 6 present the ablation study results.

## 7. Discussions

### 7.1. On error metrics

As shown in Table 4, the different alignment methods in (40) result in different error values. For example, the mean errors are always lower with the  $L_1$  alignment because it yields the theoretically minimal mean error among all pos-

| Datasets | Baseline    |             | Without $\sqrt{\cdot}$ in (27) |      | Without switching $\alpha$ |             |
|----------|-------------|-------------|--------------------------------|------|----------------------------|-------------|
|          | mn1         | mn2         | mn1                            | mn2  | mn1                        | mn2         |
| ALM      | <b>2.35</b> | <b>3.20</b> | 2.48                           | 3.35 | <b>2.35</b>                | <b>3.20</b> |
| ELS      | 1.06        | 1.48        | 0.95                           | 1.32 | <b>0.94</b>                | <b>1.27</b> |
| GDM      | <b>2.43</b> | <b>2.44</b> | 2.73                           | 2.77 | 2.45                       | 2.47        |
| MDR      | <b>4.42</b> | <b>5.71</b> | 4.82                           | 6.02 | <b>4.42</b>                | <b>5.71</b> |
| MND      | <b>0.82</b> | <b>1.01</b> | 0.94                           | 1.11 | <b>0.82</b>                | <b>1.01</b> |
| NTD      | <b>1.27</b> | 1.47        | 1.47                           | 1.75 | 1.28                       | <b>1.41</b> |
| NYC      | <b>1.03</b> | <b>1.20</b> | 1.23                           | 1.44 | <b>1.03</b>                | <b>1.20</b> |
| PDP      | <b>2.18</b> | <b>2.92</b> | 2.57                           | 3.38 | <b>2.18</b>                | <b>2.92</b> |
| PIC      | <b>1.58</b> | 1.75        | 2.43                           | 2.73 | 1.62                       | <b>1.72</b> |
| ROF      | <b>2.18</b> | <b>2.28</b> | 2.92                           | 3.09 | 3.53                       | 3.83        |
| TOL      | 1.15        | 1.20        | 1.98                           | 2.04 | <b>1.10</b>                | <b>1.14</b> |
| TFG      | <b>2.76</b> | <b>2.78</b> | 3.07                           | 3.11 | 3.53                       | 3.59        |
| USQ      | 3.26        | 3.46        | 4.32                           | 4.41 | <b>3.08</b>                | <b>3.21</b> |
| VNC      | <b>4.96</b> | <b>7.31</b> | 5.70                           | 7.93 | 5.09                       | 7.48        |
| YKM      | 1.66        | 1.74        | 1.85                           | 1.93 | <b>1.59</b>                | <b>1.66</b> |

Baseline: RA [10] + ROBA (100 iterations),  
mn1/2: mean angular error (deg) after  $L_1/L_2$  alignment

Table 5. Ablation study I: Undoing the square-rooting in (27) (line 4 of Alg. 1) worsens the accuracy for all datasets except ELS. Also, disabling the change of the step size  $\alpha$  (line 11 of Alg. 2) significantly worsens the accuracy for ROF and TFG.

sible alignments. Also, the  $L_1$  alignment often gives very small median errors because using both median and the  $L_1$  alignment significantly diminishes the influence of large errors. This was also reported in [10]. In Section 6, we used the mean error after the  $L_1$  alignment as our primary metric due to its moderate sensitivity to large errors.

## 7.2. On convergence

Most of the time, ROBA converges after 30–40 iterations, but it is subject to local minima. Generally, the better the initial rotations, the better the final result. Since rotation averaging depends entirely on the relative rotation estimates, these input rotations must be accurate enough to achieve good results. Empirically, we found that it is much better to have noisier input with fewer outliers than the other way around. We also observed that sometimes a relatively small change in the total cost (27) induces a non-negligible change in the rotational accuracy (see Fig. 3 and the supplementary material). This is somewhat in line with the observation in [7] for the two-view case.

## 7.3. On robustness to outliers

In this work, we did not consider outliers in the input, *i.e.*, the image measurements and the initial rotation estimates. We assumed that the outliers in the former have already been dealt with by a robust pose estimator and the latter by robust rotation averaging. Taking into account also the outliers in our optimization is left for future work.

| Datasets | Baseline |             |             | Without approximating $\mathbf{g}$ |             |      |
|----------|----------|-------------|-------------|------------------------------------|-------------|------|
|          | mn1      | mn2         | Opti        | mn1                                | mn2         | Opti |
| ALM      | 2.35     | 3.20        | <b>216</b>  | <b>2.23</b>                        | <b>3.09</b> | 389  |
| ELS      | 1.06     | 1.48        | <b>42</b>   | <b>0.97</b>                        | <b>1.32</b> | 76   |
| GDM      | 2.43     | 2.44        | <b>76</b>   | <b>2.36</b>                        | <b>2.40</b> | 135  |
| MDR      | 4.42     | 5.71        | <b>52</b>   | <b>4.21</b>                        | <b>5.41</b> | 93   |
| MND      | 0.82     | 1.01        | <b>114</b>  | <b>0.76</b>                        | <b>0.97</b> | 207  |
| NTD      | 1.27     | 1.47        | <b>217</b>  | <b>1.14</b>                        | <b>1.33</b> | 384  |
| NYC      | 1.03     | <b>1.20</b> | <b>42</b>   | <b>1.00</b>                        | 1.21        | 75   |
| PDP      | 2.18     | 2.92        | <b>54</b>   | <b>1.99</b>                        | <b>2.67</b> | 96   |
| PIC      | 1.58     | 1.75        | <b>617</b>  | <b>1.43</b>                        | <b>1.61</b> | 1103 |
| ROF      | 2.18     | 2.28        | <b>154</b>  | <b>1.17</b>                        | <b>1.29</b> | 274  |
| TOL      | 1.15     | 1.20        | <b>53</b>   | <b>1.10</b>                        | <b>1.13</b> | 94   |
| TFG      | 2.76     | 2.78        | <b>1488</b> | <b>2.54</b>                        | <b>2.55</b> | 2664 |
| USQ      | 3.26     | 3.46        | <b>54</b>   | <b>2.68</b>                        | <b>2.95</b> | 95   |
| VNC      | 4.96     | 7.31        | <b>221</b>  | <b>4.39</b>                        | <b>7.03</b> | 395  |
| YKM      | 1.66     | 1.74        | <b>61</b>   | <b>1.56</b>                        | <b>1.64</b> | 109  |

Baseline: RA [10] + ROBA (100 iterations),  
mn1/2: mean angular error (deg) after  $L_1/L_2$  alignment  
Opti: Optimization time (in s) (line 6–11 of Alg. 2).

Table 6. Ablation study II: In Alg. 3, computing  $\mathbf{g}$  without using the approximation (37) improves the accuracy slightly. Exceptions are ROF and USQ where we observe large gains. In all cases, however, this significantly increases the optimization time.

## 7.4. On speed and scalability

As discussed in Section 6.1, we fixed the number of iterations at 100. In some of the datasets, we observe that ROBA converges in much fewer iterations (see the supplementary material), so implementing the stopping criteria would reduce the runtime. We also note that our code was not highly optimized. Possibly, one could increase the speed by vectorizing line 2 in Al. 1 using SIMD instructions.

As can be seen from Table 4, the complexity of ROBA is linear in the number of edges. To enhance the scalability, one could partition the view-graph and perform the local and the global optimization in parallel, as in [12, 64]. This is left for future work.

## 8. Conclusion

In this work, we presented rotation-only bundle adjustment, a novel method for estimating the global rotations of multiple views independently of the translations and the scene structure. We formulate the optimization problem by extending the two-view rotation-only method of [34] and solve it using the Adam optimizer [33]. As we decouple the rotation estimation from the translation and structure estimation, it is completely immune to their inaccuracies. Our evaluation shows that (1) our method is robust to challenging configurations such as pure rotations and planar scenes, and (2) it consistently and significantly improves the accuracy when used after rotation averaging.

## References

- [1] Sameer Agarwal, Noah Snavely, Steven M. Seitz, and Richard Szeliski. Bundle adjustment in the large. In *Eur. Conf. Comput. Vis.*, pages 29–42, 2010. 2
- [2] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building Rome in a day. In *IEEE Int. Conf. Comput. Vis.*, pages 72–79, 2009. 2
- [3] Hatem Alismail, Brett Browning, and Simon Lucey. Photometric bundle adjustment for vision-based slam. In *Asian Conf. Comput. Vis.*, pages 324–341, 2017. 2
- [4] Mica Arie-Nachimson, Shahar Z. Kovalsky, Ira Kemelmacher-Shlizerman, Amit Singer, and Ronen Basri. Global motion estimation from point matches. In *Int. Conf. 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 81–88, 2012. 2
- [5] Federica Arrigoni, Beatrice Rossi, Pasqualina Fragneto, and Andrea Fusiello. Robust synchronization in  $SO(3)$  and  $SE(3)$  via low-rank and sparse matrix decomposition. *Comput. Vis. and Image Underst.*, 174:95 – 113, 2018. 1, 2
- [6] Eric Brachmann and Carsten Rother. Neural-guided RANSAC: Learning where to sample model hypotheses. In *Int. Conf. Comput. Vis.*, pages 4322–4331, 2019. 1, 2
- [7] Jesus Briales, Laurent Kneip, and Javier Gonzalez-Jimenez. A certifiably globally optimal solution to the non-minimal relative pose problem. In *IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 145–154, 2018. 1, 2, 8
- [8] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robotics*, 32(6):1309–1332, 2016. 1
- [9] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *CoRR*, abs/2007.11898, 2020. 2
- [10] Avishek Chatterjee and Venu Madhav Govindu. Robust relative rotation averaging. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):958–972, 2018. 1, 2, 5, 6, 7, 8
- [11] Chee-Kheng Chng, Alvaro Parra, Tat-Jun Chin, and Yasir Latif. Monocular rotational odometry with incremental rotation averaging and loop closure. *CoRR*, abs/2010.01872, 2020. 2
- [12] Hainan Cui, Xiang Gao, Shuhan Shen, and Zhanyi Hu. HSfM: Hybrid Structure-from-Motion. In *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017. 8
- [13] Zhaopeng Cui, Nianjuan Jiang, Chengzhou Tang, and Ping Tan. Linear global translation estimation with feature tracks. In *Brit. Mach. Vis. Conf.*, pages 46.1–46.13, 2015. 2
- [14] Zhaopeng Cui and Ping Tan. Global structure-from-motion by similarity averaging. In *IEEE Int. Conf. Comput. Vis.*, pages 864–872, 2015. 2
- [15] Amaël Delaunoy and Marc Pollefeys. Photometric bundle adjustment for dense multi-view 3D modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1486–1493, 2014. 2
- [16] Frank Dellaert, David M. Rosen, Jing Wu, Robert Mahony, and Luca Carlone. Shonan rotation averaging: Global optimality by surfing  $SO(p)^n$ . *CoRR*, abs/2008.02737, 2020. 1, 2
- [17] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(3):611–625, 2018. 2
- [18] Olof Enqvist, Fredrik Kahl, and Carl Olsson. Non-sequential structure from motion. In *IEEE Int. Conf. Comput. Vis. Workshops*, pages 264–271, 2011. 2
- [19] Anders Eriksson, Carl Olsson, Fredrik Kahl, and Tat-Jun Chin. Rotation averaging with the chordal distance: Global minimizers and strong duality. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1–1, 2019. 1, 2
- [20] Kaveh Fathian, J. P. Ramirez-Paredes, Emily A. Doucette, J. W. Curtis, and Nicholas R. Gans. QuEst: A quaternion-based approach for camera motion estimation from minimal feature points. *IEEE Robotics and Automation Letters*, 3(2):857–864, 2018. 2
- [21] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381—395, 1981. 2, 6
- [22] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Trans. Robot.*, 33(1):1–21, 2017. 2
- [23] Mercedes Garcia-Salguero, Jesus Briales, and Javier Gonzalez-Jimenez. Certifiable relative pose estimation. *CoRR*, abs/2003.13732, 2020. 1
- [24] Venu Madhav Govindu. Combining two-view constraints for motion estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, volume 2, pages 1–8, 2001. 1, 2
- [25] Richard Hartley, Khurram Aftab, and Jochen Trunpf. L1 rotation averaging using the weiszfeld algorithm. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3041–3048, 2011. 2, 5
- [26] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2003. 1, 2, 6
- [27] Richard I. Hartley. In defense of the eight-point algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(6):580–593, 1997. 2
- [28] Richard I. Hartley, Jochen Trunpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *Int. J. Comput. Vis.*, 103(3):267–305, 2013. 2, 5
- [29] Jared Heinly, Johannes L. Schönberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world\* in six days. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3287–3295, 2015. 2
- [30] Je Hyeong Hong and Christopher Zach. pPOSE: Pseudo object space error for initialization-free bundle adjustment. In *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018. 2
- [31] Hongdong Li and Richard Hartley. Five-point motion estimation made easy. In *IEEE Conf. Comput. Vis. Pattern Recog.*, volume 1, pages 630–633, 2006. 2
- [32] Lai Kang, Lingda Wu, and Yee-Hong Yang. Robust multi-view L2 triangulation via optimal inlier selection and 3D structure refinement. *Pattern Recognition*, 47(9):2974 – 2992, 2014. 1

- [33] Diederik P. Kingma and Jimmy Lei Ba. ADAM: A method for stochastic optimization. In *Int. Conf. Learn. Represent.*, 2015. 4, 8
- [34] Laurent Kneip and Simon Lynen. Direct optimization of frame-to-frame rotation. In *IEEE Int. Conf. Coput. Vis.*, pages 2352–2359, 2013. 1, 2, 3, 4, 8
- [35] Laurent Kneip, Roland Siegwart, and Marc Pollefeys. Finding the exact rotation between two images independently of the translation. In *Eur. Conf. Comput. Vis.*, pages 696–709, 2012. 2
- [36] Kurt Konolige. Sparse sparse bundle adjustment. In *Brit. Mach. Vis. Conf.*, pages 102.1–102.11, 2010. 2
- [37] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems. In *Brit. Mach. Vis. Conf.*, pages 56.1–56.10, 2008. 2
- [38] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *IEEE Int. Conf. on Robotics and Automation*, pages 3607–3613, 2011. 2
- [39] Seong Hun Lee and Javier Civera. Closed-form optimal two-view triangulation based on angular errors. In *Int. Conf. Comput. Vis.*, 2019. 6
- [40] Seong Hun Lee and Javier Civera. Loosely-coupled semi-direct monocular SLAM. *IEEE Robot. and Autom. Lett.*, 4(2):399–406, 2019. 2
- [41] Seong Hun Lee and Javier Civera. Geometric interpretations of the normalized epipolar error. *CoRR*, abs/2008.01254, 2020. 3, 6
- [42] Seong Hun Lee and Javier Civera. Robust single rotation averaging. *CoRR*, abs/2004.00732, 2020. 2, 7
- [43] Seong Hun Lee and Javier Civera. Robust uncertainty-aware multiview triangulation. *CoRR*, abs/2008.01258, 2020. 1
- [44] Manolis I. A. Lourakis and Antonis A. Argyros. SBA: A software package for generic sparse bundle adjustment. *ACM Trans. Math. Softw.*, 36(1), 2009. 2
- [45] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, 2004. 7
- [46] Daniel Martinec and Tomáš. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1–8, 2007. 2
- [47] Pierre Moulon, Pascal Monasse, and Renaud Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *IEEE Int. Conf. Coput. Vis.*, pages 3248–3255, 2013. 2
- [48] Raúl Mur-Artal, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robotics*, 31(5):1147–1163, 2015. 2
- [49] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6):756–777, 2004. 2, 6
- [50] Onur Özyesil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey on structure from motion. *CoRR*, abs/1701.08493, 2017. 2
- [51] Pulak Purkait, Tat-Jun Chin, and Ian Reid. NeuRoRA: Neural robust rotation averaging. *CoRR*, abs/1912.04485, 2020. 1
- [52] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. USAC: A universal framework for random sample consensus. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):2022–2038, 2013. 2
- [53] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry. *IEEE robotics & automation magazine*, 18(4):80–92, 2011. 1
- [54] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4104–4113, 2016. 2
- [55] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo Tourism: Exploring photo collections in 3D. *ACM Trans. Graph.*, 25(3):835–846, 2006. 7
- [56] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ Int. Conf. Intell. Robots. Syst.*, pages 573–580, 2012. 5
- [57] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – a modern synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–375, 2000. 1, 2
- [58] Roberto Tron, Xiaowei Zhou, and Kostas Daniilidis. A survey on rotation optimization in structure from motion. In *IEEE Conf. Comput. Vis. Pattern Recog. Workshops*, pages 1032–1040, 2016. 2
- [59] Kyle Wilson and Noah Snavely. Robust global translations with 1DSfM. In *Eur. Conf. Comput. Vis.*, pages 61–75, 2014. 1, 2, 7
- [60] Oliver J. Woodford and Edward Rosten. Large scale photometric bundle adjustment. In *Brit. Mach. Vis. Conf.*, pages 1–12, 2020. 2
- [61] Christopher Zach. Robust bundle adjustment revisited. In *Eur. Conf. Comput. Vis.*, pages 772–787, 2014. 2
- [62] Christopher Zach and Guillaume Bourmaud. Descending, lifting or smoothing: Secrets of robust cost optimization. In *Eur. Conf. Comput. Vis.*, 2018. 2
- [63] Ji Zhao. An efficient solution to non-minimal case essential matrix estimation. *CoRR*, abs/1903.09067, 2019. 1, 2
- [64] Siyu Zhu, Runze Zhang, Lei Zhou, Tianwei Shen, Tian Fang, Ping Tan, and Long Quan. Very large-scale global SfM by distributed motion averaging. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2, 8
- [65] Álvaro Parra Bustos, Tat-Jun Chin, Anders Eriksson, and Ian Reid. Visual slam: Why bundle adjust? In *IEEE Int. Conf. on Robotics and Automation*, pages 2385–2391, 2019. 2
- [66] Onur Özyesil and Amit Singer. Robust camera location estimation by convex programming. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2674–2683, 2015. 2