Minimal Solvers for Mini-Loop Closures in 3D Multi-Scan Alignment

Pedro Miraldo
KTH Royal Institute of Technology
miraldo@kth.se

Surojit Saha and Srikumar Ramalingam
School of Computing, The University of Utah
{surojit,srikumar}@cs.utah.edu

Abstract

3D scan registration is a classical, yet a highly useful problem in the context of 3D sensors such as Kinect and Velodyne. While there are several existing methods, the techniques are usually incremental where adjacent scans are registered first to obtain the initial poses, followed by motion averaging and bundle-adjustment refinement. In this paper, we take a different approach and develop minimal solvers for jointly computing the initial poses of cameras in small loops such as 3-, 4-, and 5-cycles[1]*. Note that the classical registration of 2 scans can be done using a minimum of 3 point matches to compute 6 degrees of relative motion. On the other hand, to jointly compute the 3D registrations in n-cycles, we take 2 point matches between the first $n-1$ consecutive pairs (i.e., Scan 1 & Scan 2, . . . , and Scan $n-1$ & Scan n) and 1 or 2 point matches between Scan 1 and Scan n. Overall, we use 5, 7, and 10 point matches for 3-, 4-, and 5-cycles, and recover 12, 18, and 24 degrees of transformation variables, respectively. Using simulations and real-data we show that the 3D registration using mini n-cycles are computationally efficient, and can provide alternate and better initial poses compared to standard pairwise methods.*

1. Introduction

Many geometers working on algebraic minimal solvers have attempted to solve the notorious and classical 3-view 4-point relative pose estimation. Given 4 triplets of point matches, the goal is to jointly find the poses of the 3 cameras. There have been some great progress on this problem using one-dimensional search [38] and semi-definite programming [27], but we still miss the simple and direct minimal algebraic solver that we usually derive for most geometric vision problems. If one manages to solve this problem for RGB cameras, what would be the next big challenge? Do we look at the 4-view 3-point relative pose problem? While there has been a great deal of effort to solve the higher-order pose estimation in the case of RGB sensors, the equivalent problem with RGB-D cameras has received no attention. In the case of RGB-D sensors, the number of correspondences for the n-camera relative pose problem is less notorious for $n \leq 5$, and even practically deployable. At this point, when the price point for commercial RGB-D sensors is decreasing due to the progress in robotics and self-driving industry, it would be a good time to fully equip the arsenal with algebraic minimal solvers for depth sensors.

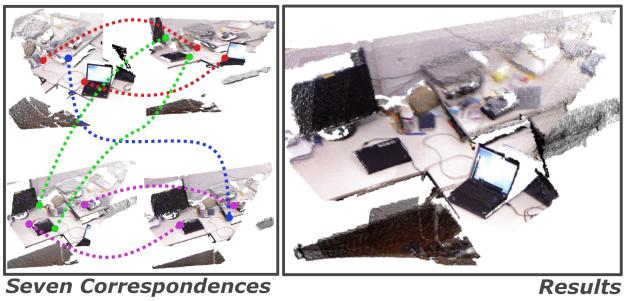


Figure 1: *At the left we show four scans and seven 3D point matches (2 from Scans 1 & Scan 2; 2 from Scans 2 & Scan 3; 2 from Scans 3 & Scan 4; and 1 from Scans 1 & Scan 4). At the right, we show the registered scan using our minimal solver for 4-cycle, that jointly computes the pose parameters for all the four cameras.*

In Fig. 1 we show four different scans collected using a Kinect sensor. We jointly compute the 3D registration for all four scans using a minimal solver that uses a total of seven point matches. We are able to compute 18 degrees of transformation variables, and the points from all the four scans are registered as shown in Fig. 1. Previous methods for RGB-D registration typically employ pair-wise registration where the initial poses are computed between pairs of cameras, and a final refinement is done using a non-linear refinement technique. The pairwise methods (see Orthogonal Procrustes problem [45] that uses a minimum of three point matches) typically accumulate drift even in the case of three cameras. Our formulation naturally eliminates the

[1] A cycle graph C_n, also referred to as n-cycles, is a subgraph with n nodes and edge set $\{(1,2),\ldots,(n-1,n),(n,1)\}$.

drift in these mini $n-$cycles, and thereby provides better pose parameters. This paper systematically studies the possibility of joint 3D registration for mini $n-$cycles, and derives algebraic minimal solvers, which are typically embedded in a Random Sample Consensus (RANSAC) [13] framework for robust estimation of pose parameters. It has been well established that minimal solvers and RANSAC tend to perform robustly in the presence of outliers.

1.1. Related Work

We carefully survey some of the classical and modern registration algorithms that employ 3D sensors.

3D scan alignment: The classic approach to solve the 3D scan alignment problem is the Iterated Closest Point (ICP) algorithm, proposed in [3]. Over the years, several efficient and robust solutions have been proposed in the literature to solve the 3D multi-scan alignment using 3D points, such as [46, 35, 36, 56, 15, 29, 40]. A method for fast and efficient 3D rotation search is proposed in [5].

Besides the classic approaches presented above, alternate methods have been proposed that utilize the properties of the observed 3D scene. In [12], a beam-based environment measurement model was introduced to achieve frame-to-frame registration. In [42, 32, 4, 30] we use 3D planes to improve the SLAM using 3D cameras. In [31] we extract and use 3D straight lines for 3D SLAM, while [9] focuses on edge detection. In [16], a more general method is proposed to detect and enforce constraints (geometric relationships and feature correspondences). Surveys on the evaluation of 3D SLAM methods were presented in [11, 49]. There have also been some solvers for the non-rigid 3D registration problems (see for example [59, 2, 47, 33]). A survey on rigid and non-rigid registration of 3D point clouds is presented in [51].

In addition to finding the 3D transformations that align 3D scans, there have been some developments on doing both the 3D registration and semantic segmentation using RGB-D images. Several works were proposed such as [52, 44, 57, 58].

Recently, some deep learning techniques techniques were used in order to obtain 3D registration. In [10], local 3D geometric structures are extracted using a deep neural network auto-encoder. Compact geometric features are learned in [18]. Automatic reconstruction of floorplans is achieved using a deep learning algorithm in [30].

Minimal solvers: We review some of the minimal solutions that are relevant to pose estimation using RGB cameras. Several solutions were proposed for the absolute pose for central perspective cameras (three 3D point correspondences between the world and image), see for example [19, 17, 55, 41]. The pose estimation has also been studied for the pose of multi-perspective systems, such as [54, 24, 7, 34].

When considering the relative pose estimation, several approaches have also been proposed for solving the minimal relative pose problem. See for example [37, 28] for calibrated cameras. There are other solutions such as [25] which studies the relative pose estimation with a known relative rotation angle, [14, 43] for the relative pose with known directions, [26] for the relative pose with unknown focal length, solutions invariant to translation [20], and solutions to the generalized relative pose problem [48, 53]. In [6], a hybrid minimal solver that combines relative with absolute poses is proposed.

Cycle #Cameras	#Correspondences	Total	#Solutions
Two	$\#\mathbf{3}(\mathcal{S}_1, \mathcal{S}_2)$	3	2
Three	$\#\mathbf{2}(\mathcal{S}_1, \mathcal{S}_2)$; $\#\mathbf{2}(\mathcal{S}_2, \mathcal{S}_3)$; $\#\mathbf{1}(\mathcal{S}_1, \mathcal{S}_3)$;	5	4
Four	$\#\mathbf{2}(\mathcal{S}_1, \mathcal{S}_2)$; $\#\mathbf{2}(\mathcal{S}_2, \mathcal{S}_3)$; $\#\mathbf{2}(\mathcal{S}_3, \mathcal{S}_4)$; $\#\mathbf{1}(\mathcal{S}_1, \mathcal{S}_4)$	7	16
Five	$\#\mathbf{2}(\mathcal{S}_1, \mathcal{S}_2)$; $\#\mathbf{2}(\mathcal{S}_2, \mathcal{S}_3)$; $\#\mathbf{2}(\mathcal{S}_3, \mathcal{S}_4)$; $\#\mathbf{2}(\mathcal{S}_4, \mathcal{S}_5)$ $\#\mathbf{2}(\mathcal{S}_1, \mathcal{S}_5)$	10	32

Table 1: *This table summarizes the minimal number of correspondences required to compute the 3D point registration. In the table, $\#\boldsymbol{i}(\mathcal{S}_j, \mathcal{S}_k)$ means i point correspondences within the sequence of point clouds $\mathcal{S}_j$ and $\mathcal{S}_k$.*

1.2. Notation and Problem Definition

For simplicity, we use $\mathcal{S}_n$ to denote Scan n. The i^{th} 3D point in $\mathcal{S}_n$ is denoted as $\mathbf{p}_i^{\mathcal{S}_n} \in \mathbb{R}^4$, which is represented in homogeneous coordinates. Rotation matrices and translation vectors are denoted as $\mathbf{R}^{\mathcal{S}_n,\mathcal{S}_m} \in \mathcal{SO}(3)$ & $\mathbf{t}^{\mathcal{S}_n,\mathcal{S}_m} \in \mathbb{R}^3$, for transformations from $\mathcal{S}_n$ to $\mathcal{S}_m$. We use the n-cycle to denote the sequences of n 3D scans with loop closure (first and last point clouds in the sequence have 3D point correspondences).

The goal is to find the transformation matrices $\mathbf{T}^{\mathcal{S}_n,\mathcal{S}_m} \in \mathcal{SE}(3)$ that transform 3D points from coordinate system $\mathcal{S}_n$ to $\mathcal{S}_m$ such that

$$\mathbf{p}_i^{\mathcal{S}_m} \simeq \underbrace{\begin{bmatrix} \mathbf{R}^{\mathcal{S}_n,\mathcal{S}_m} & \mathbf{t}^{\mathcal{S}_n,\mathcal{S}_m} \\ \mathbf{0}_{1,3} & 1 \end{bmatrix}}_{\mathbf{T}^{\mathcal{S}_n,\mathcal{S}_m}} \mathbf{p}_i^{\mathcal{S}_n}. \tag{1}$$

We are given sets of 3D point matches $(\mathbf{p}_i^{\mathcal{S}_n}, \mathbf{p}_i^{\mathcal{S}_m})$. Symbol $\simeq$ denotes that the terms are equal up to a scale factor.

Contributions: We propose novel minimal solvers for the mini $n-$cycles in 3D point cloud registration. We propose three solvers for 3-, 4-, and 5-cycles for the general six degrees of freedom and planar motions. The Tab. 1 highlights the different $n-$cycles, required point correspondences, and the number of solutions. To the best of our knowledge, we are the first to propose and solve these cases.

2. Minimal Solvers

In this section, we formulate the minimal solution for jointly estimating the poses of $n-$cameras that occur in an $n-$cycle. In all the $n-$cycles, when $n > 2$ we use a simple geometric idea. Let us assume that we would like to find the registration between two different camera scans $\mathcal{S}_1$ and $\mathcal{S}_2$. As shown in Fig. 2(a), the basic idea is to first use two point correspondences to construct a virtual axis passing through these two points. Now we align the coordinate frames of $\mathcal{S}_1$ and $\mathcal{S}_2$ in such a manner that the $z-$axis of both these frames are aligned along this virtual axis. The triplets $\{\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z\}$ and $\{\boldsymbol{f}_x, \boldsymbol{f}_y, \boldsymbol{f}_z\}$ denote the coordinate frames for both these cameras after the alignment. Next, the problem of estimating the transformations between these coordinate frames can be seen as just estimating the rotation angle around the $z-$axis. This idea of using simple predefined transformations before the actual registration allows us to simplify the constraint equations. Once we obtain the final registration, we can always find the relative poses between the original coordinate frames, by just using the inverses of the predefined transformation matrices.

Next, we show the details of the predefined transformations that we use on the original scans, so that the actual minimal solvers become easier to derive (see Tab. 1).

2.1. Setting the Stage for Minimal Solvers

Let us consider two point matches $(\mathbf{p}_1^{\mathcal{S}_1}, \mathbf{p}_2^{\mathcal{S}_1})$ and $(\mathbf{p}_1^{\mathcal{S}_2}, \mathbf{p}_2^{\mathcal{S}_2})$ in $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively. We consider the predefined transformations to align the scans such that the new coordinates frames of $\mathcal{S}_1$ and $\mathcal{S}_2$ satisfy the following conditions:

- Centered in $\mathbf{p}_1^{\mathcal{S}_1}$ and $\mathbf{p}_1^{\mathcal{S}_2}$, respectively;
- $z-$axis of both frames are aligned with directions $(\mathbf{p}_2^{\mathcal{S}_1}$ - $\mathbf{p}_1^{\mathcal{S}_1})$ and $(\mathbf{p}_2^{\mathcal{S}_2}$ - $\mathbf{p}_1^{\mathcal{S}_2})$, respectively.

A depiction of these predefined transformations is shown in Fig. 2(a). To get these, we define transformation matrices $\mathbf{H}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1}, \mathbf{G}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2} \in \mathcal{SE}(3)$ such that

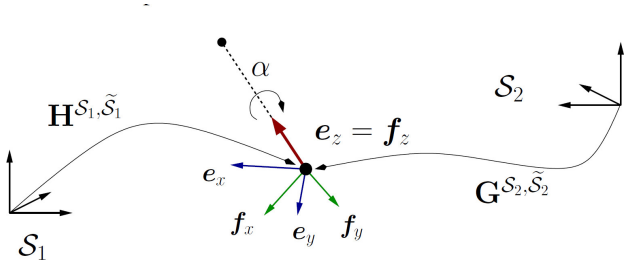
$$\mathbf{p}_{1,2}^{\widetilde{\mathcal{S}}_1} \simeq \mathbf{H}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1}\mathbf{p}_{1,2}^{\mathcal{S}_1} \quad \text{and} \quad \mathbf{p}_{1,2}^{\widetilde{\mathcal{S}}_2} \simeq \mathbf{G}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2}\mathbf{p}_{1,2}^{\mathcal{S}_2}, \tag{2}$$

where $\widetilde{\mathcal{S}}_n$ denotes the transformed point clouds and

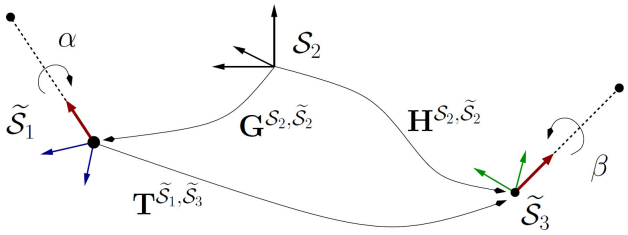
$$\mathbf{H}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1} = \begin{bmatrix} \mathbf{U}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{q}_1^{\mathcal{S}_1} \\ \mathbf{0} & 1 \end{bmatrix} \text{ and} \tag{3}$$

$$\mathbf{G}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2} = \begin{bmatrix} \mathbf{V}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{q}_1^{\mathcal{S}_2} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{4}$$

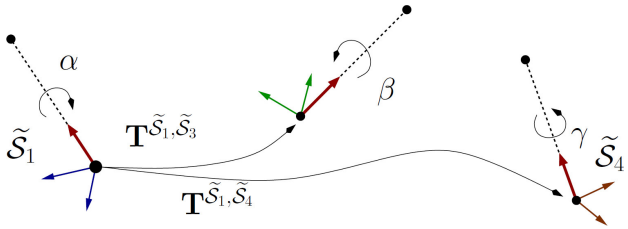
in which $\mathbf{U}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1}, \mathbf{V}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2} \in \mathcal{SO}(3)$ are any rotation matrices that align the $z-$axis of $\mathcal{S}_1$ and $\mathcal{S}_2$ (respectively) with the direction from $\mathbf{p}_1$ to $\mathbf{p}_2$, and $\mathbf{q}_1 \in \mathbb{R}^3$ represents the regular coordinates of $\mathbf{p}_1$.



(a) *Predefined transformations for two point clouds and two correspondences. The remaining degree of freedom is α. Note that $\boldsymbol{e}$ and $\boldsymbol{f}$ are the transformed coordinate frames of $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively.*



(b) *Three point clouds, two correspondences between $\mathcal{S}_1$ & $\mathcal{S}_2$ and $\mathcal{S}_2$ & $\mathcal{S}_3$. α and β are the remaining degrees of freedom.*



(c) *Four point clouds, two correspondences between $\mathcal{S}_1$ & $\mathcal{S}_2$, $\mathcal{S}_2$ & $\mathcal{S}_3$, and $\mathcal{S}_3$ & $\mathcal{S}_4$. α, β, and γ are the remaining degrees of freedom.*

Figure 2: *Representation of the predefined transformations (a) and the resulting degrees of freedom for the three camera 3D registrations, with two point correspondences between point clouds one & two and two & three (b). (c) shows the remaining degrees of freedom for four point clouds and two 3D point correspondences between one & two, two & three, and three & four.*

The transformation matrix from $\mathcal{S}_1$ to $\mathcal{S}_2$, after applying predefined transformations, is as follows:

$$\mathbf{T}^{\mathcal{S}_1,\mathcal{S}_2} = \mathbf{G}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2\,-1} \underbrace{\begin{bmatrix} c\alpha & -s\alpha & 0 & 0 \\ s\alpha & c\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{L}(\alpha)} \mathbf{H}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1}, \tag{5}$$

where $\mathbf{L}(\alpha)$ is a single degree of freedom transformation matrix representing a rotation around the $z-$axis. We use $c\alpha$ and $s\alpha$ to denote $\cos(\alpha)$ and $\sin(\alpha)$, respectively.

Once we align the coordinate frames using the predefined transformations, all we have to compute is one rota-

tion angle for every pair of 3D scans (see Fig. 2). So, for the case of having two scans, we just focus on getting the one unknown rotation from Scan 1 to Scan 2. In the next few sections, we show the minimal solutions for $n-$cycles. Note that this idea of using virtual axis to register scans is straightforward in the case of two cameras, but a little intriguing when we start using multiple axes. For different pairs of cameras in the case of $n-$cycles, when $n > 2$, the underlying idea is still the same. We use only 2 point correspondences between different pairs of 3D scans to realize the predefined transformations (refer to Fig. 2(a)). Following this, we just need to find the corresponding rotation angles.

2.2. Pairwise Registration

We show the two camera registration for illustrating the idea. By considering the predefined transformations defined in the previous subsection, this can be easily achieved by considering a third point correspondence between $\widetilde{\mathcal{S}}_1$ and $\widetilde{\mathcal{S}}_2$ (see (2)), and checking for α that satisfies

$$\mathbf{p}_3^{\widetilde{\mathcal{S}}_2} \simeq \mathbf{L}(\alpha)\mathbf{p}_3^{\widetilde{\mathcal{S}}_1}. \tag{6}$$

Notice that (6) has two linear equations as a function of $c\alpha$ and $s\alpha$, meaning that we can compute a single solution for both variables, and therefore a single solution for α. However, when using noisy data, solutions for $c\alpha$ and $s\alpha$ will not satisfy the trigonometric constraints $c\alpha^2+s\alpha^2 = 1$. To avoid this, we consider a single constraint of (6), which we solve as a function of $c\alpha$ and replace it in $c\alpha^2+s\alpha^2 = 1$, which gives up to two solution to the problem. Although this approach gives more than one solution, they ensure $\mathbf{L}(\alpha)$ is a rotation matrix and therefore $\mathbf{T}^{\mathcal{S}_1,\mathcal{S}_2}$ is a transformation matrix. In addition, one can remove one of the solutions by back-substituting them in (6). As in Procrustes's solver, this can be computed in closed-form.

In the following sections, we show the registration for $n-$cycles for $n > 2$. Note that we establish constraints between different pairs of cameras, but the 3D registration for all the cameras is computed by jointly solving all the equations. In other words, the registration is a higher-order one and not solving different pairwise registrations independently.

2.3. 3-Cycle Registration

Now, let us consider three point clouds $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, and two correspondences between $\mathcal{S}_1$ and $\mathcal{S}_2$, and two correspondences between $\mathcal{S}_2$ and $\mathcal{S}_3$. We start by considering some predefined transformations to the point clouds, to ensure that the respective 3D points satisfy the assumptions of Sec. 2.1. We aim at finding $\widetilde{\mathcal{S}}_1$, $\widetilde{\mathcal{S}}_2$, and $\widetilde{\mathcal{S}}_3$ that allow us to write constraints similar to (6). For this purpose, one has to find $\mathbf{H}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1}$, $\mathbf{G}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2}$, $\mathbf{H}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2}$, and $\mathbf{G}^{\mathcal{S}_3,\widetilde{\mathcal{S}}_3}$ similar to the ones in (3) and (4), such that

$$\mathbf{p}_i^{\widetilde{\mathcal{S}}_1} \simeq \mathbf{H}^{\mathcal{S}_1,\widetilde{\mathcal{S}}_1}\mathbf{p}_i^{\mathcal{S}_1} \text{ and } \mathbf{p}_j^{\widetilde{\mathcal{S}}_3} \simeq \mathbf{G}^{\mathcal{S}_3,\widetilde{\mathcal{S}}_3}\mathbf{p}_j^{\mathcal{S}_3}. \tag{7}$$

Using these predefined transformations, we define the transformation from $\widetilde{\mathcal{S}}_1$ to $\widetilde{\mathcal{S}}_3$ as

$$\mathbf{T}^{\widetilde{\mathcal{S}}_1,\widetilde{\mathcal{S}}_3} = \mathbf{L}(\beta)\underbrace{\mathbf{H}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2}\,\mathbf{G}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2\,-1}}_{\mathbf{K}_2\in\mathcal{SE}(3)}\mathbf{L}(\alpha). \tag{8}$$

By doing this, we reduce the problem of estimating the transformation between three 3D scans to two degrees of freedom (in this case angles α and β). A graphical representation of this problem is shown in Fig. 2(b).

Now, to compute the transformations we have to use addition information. Let us consider that we have a correspondence between $\mathcal{S}_1$ and $\mathcal{S}_3$, i.e. a correspondence to close the cycle between the first and third cameras (notice that additional correspondences between $\mathcal{S}_1$ & $\mathcal{S}_2$ and $\mathcal{S}_2$ & $\mathcal{S}_3$ can be solved by the method proposed in Sec. 2.3). Let us denote the correspondence point between $\mathcal{S}_1$ and $\mathcal{S}_3$ as $\mathbf{p}_5^{\mathcal{S}_1}$ and $\mathbf{p}_5^{\mathcal{S}_3}$, respectively. By applying the predefined transformation to the data as shown in (7), and using (8), we get three constraints of the form

$$\mathbf{p}_5^{\widetilde{\mathcal{S}}_3} \simeq \mathbf{L}(\beta)\mathbf{K}_2\mathbf{L}(\alpha)\mathbf{p}_5^{\widetilde{\mathcal{S}}_1}. \tag{9}$$

Notice that we have two unknowns and three constraints in (9). Therefore, in general, it is possible to find α and β with only one point correspondence.

To solve this problem, we use the fact that the third constraint in (9) (i.e. its third row) only depends on the unknown parameter α:

$$a_1c\alpha + a_2s\alpha + a_3 = 0, \tag{10}$$

where a_1, a_2, a_3 are known coefficients. On the other hand, if we consider the inverse transformation $\mathbf{T}^{\widetilde{\mathcal{S}}_3,\widetilde{\mathcal{S}}_1}$:

$$\mathbf{p}_5^{\widetilde{\mathcal{S}}_1} \simeq \mathbf{L}(\alpha)^T\mathbf{K}_2^{-1}\mathbf{L}(\beta)^T\mathbf{p}_5^{\widetilde{\mathcal{S}}_3} \tag{11}$$

and use, again, the third row of (11), we get a constraint that only depends on β:

$$a_4c\beta + a_5s\beta + a_6 = 0. \tag{12}$$

Now, to solve the problem we just have to solve (10) & (12), using the trigonometric constraints $c\alpha^2 + s\alpha^2 = 1$ & $c\beta^2 + s\beta^2 = 1$. Note that the unknowns are decoupled, meaning that we can compute them separately. This can be done as follows: 1) we solve (10) as a function of $c\alpha$; 2) substitute the solution in $c\alpha^2 + s\alpha^2 = 1$ (which gives a two degree polynomial equation in $c\alpha$); and 3) compute the roots of the resulting equation giving up to two solutions to $c\alpha$. The value for $s\alpha$ is given by choosing one in

$\{\pm\sqrt{1-c\alpha^2}\}$ that satisfy (10). This procedure is repeated for the $s\beta$ and $c\beta$, giving two additional solutions for these two unknowns. Since the pairs of solutions for α and β are decoupled, we will have up to four valid solutions for our problem (as reported in Tab. 1). Next, we study the four 3D scans case.

2.4. 4-Cycle Registration

Let us consider 4 point clouds. Again, assume that we have two correspondences between $\mathcal{S}_1$ & $\mathcal{S}_2$, $\mathcal{S}_2$ & $\mathcal{S}_3$, and $\mathcal{S}_3$ & $\mathcal{S}_4$ (see Fig. 2(c)). By following the same assumptions of previous subsections, we get $\mathbf{T}^{\mathcal{S}_1,\mathcal{S}_2}$ as in (5),

$$\mathbf{T}^{\mathcal{S}_2,\mathcal{S}_3} = \left(\mathbf{G}^{\mathcal{S}_3,\widetilde{\mathcal{S}}_3}\right)^{-1}\mathbf{L}(\beta)\mathbf{H}^{\mathcal{S}_2,\widetilde{\mathcal{S}}_2}, \text{ and} \tag{13}$$

$$\mathbf{T}^{\mathcal{S}_3,\mathcal{S}_4} = \left(\mathbf{G}^{\mathcal{S}_4,\widetilde{\mathcal{S}}_4}\right)^{-1}\mathbf{L}(\gamma)\mathbf{H}^{\mathcal{S}_3,\widetilde{\mathcal{S}}_3}. \tag{14}$$

The matrices $\mathbf{G}$ and $\mathbf{H}$ are given by applying the method in Sec. 2.1. Therefore, we have only three degrees of freedom remaining to get the relative poses between all the four 3D scans. More specifically, angles α, β, and γ. A trivial solution to this problem would be to consider additional correspondences between $\mathcal{S}_1$ & $\mathcal{S}_2$, $\mathcal{S}_2$ & $\mathcal{S}_3$, or $\mathcal{S}_3$ & $\mathcal{S}_4$. One could use a combination of the methods presented in the previous subsections to solve the relative positions between the cameras. However, here we are interested in the 4-cycles, i.e. only one correspondence between $\mathcal{S}_1$ and $\mathcal{S}_4$ in addition to the pairwise correspondences.

By premultiplying the transformations defined in (5), (13), and (13), we can define

$$\mathbf{T}^{\widetilde{\mathcal{S}}_1,\widetilde{\mathcal{S}}_4} = \mathbf{L}(\gamma)\mathbf{K}_3\mathbf{L}(\beta)\mathbf{K}_2\mathbf{L}(\alpha), \tag{15}$$

where $\mathbf{K}_i \in \mathcal{SE}(3) = \mathbf{H}^{\mathcal{S}_i,\widetilde{\mathcal{S}}_i}\ {\mathbf{G}^{\mathcal{S}_i,\widetilde{\mathcal{S}}_i}}^{-1}$ (similar to (8)).

Now, if we have an additional correspondence between $\mathcal{S}_1$ and $\mathcal{S}_4$ (let's say $\mathbf{p}_7$), we write

$$\mathbf{p}_7^{\widetilde{\mathcal{S}}_4} \simeq \mathbf{L}(\gamma)\mathbf{K}_3\mathbf{L}(\beta)\mathbf{K}_2\mathbf{L}(\alpha)\mathbf{p}_7^{\widetilde{\mathcal{S}}_1}. \tag{16}$$

Notice that we have three equation and three unknowns, meaning that in general one can get a solution for the relative poses using a single point correspondence.

To solve the problem, we take the three constraints in (16), together with $c\alpha^2 + s\alpha^2 = 1$, $c\beta^2 + s\beta^2 = 1$, and $c\gamma^2 + s\gamma^2 = 1$. Since in this case we have many unknowns and high degree polynomial equations, we aim at using automatic solvers (e.g. [22, 23]). In this paper we use the automatic *Grobner Basis* generator provided in [21]. As inputs for the automatic generator, we give the unknowns $c\alpha$, $c\beta$, $c\gamma$, $s\alpha$, $s\beta$, & $s\gamma$ and the three constraints of (16) plus the three trigonometric constraints. The solver gives up to 16 solutions, as indicated in the Tab. 1.

2.5. 5-Cycle Registration

We start by trying a general method for $n-$cycles, and show that is feasible only till $n = 5$. Similar to the cases defined in the previous subsections, we consider two point correspondences between the sequences of 3D scans (without closing any cycle). Using this data and considering the previously defined predefined transformations (Sec. 2.1), we get matrices $\mathbf{K}_i$ as shown in (8) and (15). Using this information and applying the predefined transformations to the first and last point-clouds (similar to (7)), for an $n-$cycle loop we define the transformation from $\widetilde{\mathcal{S}}_1$ to $\widetilde{\mathcal{S}}_n$ as

$$\mathbf{T}^{\widetilde{\mathcal{S}}_1,\widetilde{\mathcal{S}}_n} = \mathbf{L}(\theta_{n-1})\mathbf{K}_{n-1}\mathbf{L}(\theta_{n-2})\cdots\mathbf{L}(\theta_2)\mathbf{K}_2\mathbf{L}(\theta_1), \tag{17}$$

where θ_i are the unknown degrees of freedom.

Now, for any $n = \{5, 6, 7\}$, we will have between four to six degrees of remaining unknowns. Since each point correspondence between the first and the last 3D scans generates three constraints, we will need two point correspondences to close the loop between $\widetilde{\mathcal{S}}_1$ and $\widetilde{\mathcal{S}}_n$:

$$\mathbf{p}_{l+1}^{\widetilde{\mathcal{S}}_n} \simeq \mathbf{T}^{\widetilde{\mathcal{S}}_1,\widetilde{\mathcal{S}}_n}\mathbf{p}_{l+1}^{\widetilde{\mathcal{S}}_1} \text{ and } \mathbf{p}_{l+2}^{\widetilde{\mathcal{S}}_n} \simeq \mathbf{T}^{\widetilde{\mathcal{S}}_1,\widetilde{\mathcal{S}}_n}\mathbf{p}_{l+2}^{\widetilde{\mathcal{S}}_1}, \tag{18}$$

where $l = 2(n-1)$.

Similar to what we did in the previous subsection, we use the standard *Grobner Basis* generator [21]. Specifically, we provide the generator $c\theta_i$ and $s\theta_i$ (a total of $2(n-1)$ variables) as the unknowns, and choose $n-1$ constraints within the set of equations in (18). The remaining $n-1$ constraints are given by the trigonometric relations $c\theta_i^2 + s\theta_i^2 = 1$. The number of solutions for the solver with $n = 5$ is 32 (as shown in Tab. 1). As we can observe, this line of research may become computationally infeasible when $n > 5$ [6, 50]. For example, in the case of $n = 6$, we may have up to 288 solutions and there is no easy way to build the solver.

3. Planar Motion Case

We consider the problem of solving the 3D registration between scans when there is only planar motion between the point-clouds (3 degrees of freedom – 2 translation and 1 rotation).

We note that, in Sec. 2.1, while $\mathbf{p}_1$ is used to set the point cloud's coordinate system (see (2)), the $\mathbf{p}_2$ is only used to set the direction of the $z-$axis. Now, one of the features of the planar motion is that the rotation matrices between the sequences of 3D scans will have associated a single rotation angle. Without loss of generality, the respective rotation axis can be freely chosen, and in this case we choose the $z-$axis. Using this choice, one can conclude that the second point correspondence in the method presented in Sec. 2.1 is not needed. Therefore, for the computation of the predefined transformations defined in Sec. 2.1, only one 3D point

Loop Cycle #Cameras	#Correspondences	Total	#Solutions
Two	$\#\mathbf{2}(\mathcal{S}_1, \mathcal{S}_2)$	3	2
Three	$\#\mathbf{1}(\mathcal{S}_1, \mathcal{S}_2)$; $\#\mathbf{1}(\mathcal{S}_2, \mathcal{S}_3)$; $\#\mathbf{1}(\mathcal{S}_1, \mathcal{S}_3)$	3	4
Four	$\#\mathbf{1}(\mathcal{S}_1, \mathcal{S}_2)$; $\#\mathbf{1}(\mathcal{S}_2, \mathcal{S}_3)$; $\#\mathbf{1}(\mathcal{S}_3, \mathcal{S}_4)$; $\#\mathbf{1}(\mathcal{S}_1, \mathcal{S}_4)$	4	16

Table 2: *This table summarizes the minimal number of correspondences required to compute the poses in $n-$cycles while considering planar motions. In the table, $\#\boldsymbol{i}(\mathcal{S}_j, \mathcal{S}_k)$ means i point correspondences within the sequence of point clouds $\mathcal{S}_j, \mathcal{S}_k$.*

correspondence is required for each pair of 3D scans. The rest of the solvers follow the steps derived in Secs. 2.2, 2.3, 2.4, and 2.5.

A summary of the number of the correspondences needed for these problems, as well as the number of solutions that the solvers give is shown in Tab. 2. Notice that, in this case, the minimal solution for the two point-cloud registration is two 3D points, meaning that we are looking for cycles that consider less than two point correspondences between point-clouds. For that reason, we are only interested in mini-loop cycles up to four 3D scans.

4. Motion Averaging

In this section, we show a method to use our $n-$cycle solvers to generate initial relative poses for a large collection of 3D scans. First, we construct a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ to denote the pose relationship between the cameras. The vertices $\mathcal{V}$ of this graph denote the poses of the cameras, and the edges $\mathcal{E}$ exist if two cameras have any scene overlap. We use SURF feature correspondences on the RGB components of the data to identify the edges for all pairs of cameras in the pose graph. We consider an edge between two cameras if we find at least T feature correspondences between them.

Edge-disjoint pose graph decomposition: In this method, we decompose the pose graph into edge-disjoint mini-loops. To achieve this we use a simple depth first search (DFS) traversal of the graph to identify $n-$cycles and remove the corresponding edges, so that they do not reappear in the next iteration. We first identify all the edge-disjoint 5-cycles from the graph, and then move on 4-cycles. Once we identify all the cycles with $n = 3, 4, 5$, the remaining edges are handled using the pairwise method. We initialize the relative poses between pairs of cameras using the associated $n-$cycle solvers, or the simple pairwise solver if an edge is not a member of an $n-$cycle.

Rotation averaging using Lie group: We obtain the relative poses between different pairs of cameras using $n-$cycle minimal solvers. Due to the redundancy in the edges (i.e., we only need a set of edges in a spanning tree to uniquely compute the pose of each camera), we will have to perform some kind of averaging of the pose parameters. We use the rotation averaging framework developed by Chatterjee and Govindu [8]. Their approach is to first consider the Lie group structure of 3D rotations and solve the rotation averaging using the L_1 method. Using the results from L_1 optimizers as initialization, they use an iteratively reweighted least squares (IRLS) approach to derive solutions that are robust to outliers. Once the rotation parameters are computed, the remaining problem is just linear in the translation and standard least squares minimization can be used.

Method	`Pairwise`	`3-cycle`	`4-cycle`	`5-cycle`
Mean [ms]	0.0392	0.1192	3.3422	24.954

Table 3: *Computation timings for $n-$cycle solvers in milliseconds (ms). Note that the implementation is in Matlab, a C++ implementation would speedup the computation time.*

5. Experimental Results

We conducted two sets of experiments: (1) 3D registration on small $n-$cycle graphs to illustrate the advantages over pairwise methods, (2) 3D registration on a large dataset by first decomposing the pose graph into smaller edge-disjoint $n-$cycles, solving the registration using minimum $n-$cycle solvers, and finally evaluating the error with respect to the ground truth.

5.1. Synthetic Data

We consider 400 randomly generated 3D points and five 3D cameras in the environment, within a cube of 400 units of side length. We consider point correspondences between different camera pairs. We select a subset of 20 to 70% random correspondences for testing our algorithms.

Computational time and the number of solutions: From the data as defined above, we select the minimal number of correspondences for each of the methods in Tab. 1, and compute the 3D registration as defined in Sec. 2. We consider the cases: Pairwise, 3-, 4-, and 5-cycles. We repeat this procedure 10^5 times with randomly generated data in each test. In Fig. 3, we show the distribution of the number of solutions[2]. The computation time for the solvers is given in Tab. 3. Note that the pairwise and 3-cycle cases can be computed using closed-form operations, while the 4- and 5-cycle cases require iterative techniques, this is reflected in the experimental results.

Evaluation of the proposed solvers: We use Gaussian noise with a standard deviation that depends on the distance

[2]This graphic is limited in both the number of solutions (the number of solutions for more than 16 is very small) and the number of occurrences.

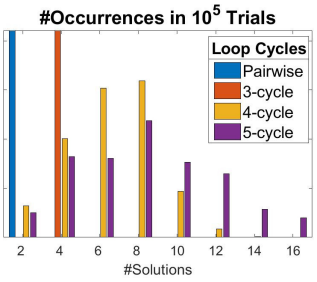


Figure 3: *Number of solutions obtained from the $n-$cycle solvers proposed in Sec. 2. 10^5 randomly generated trials were considered.*

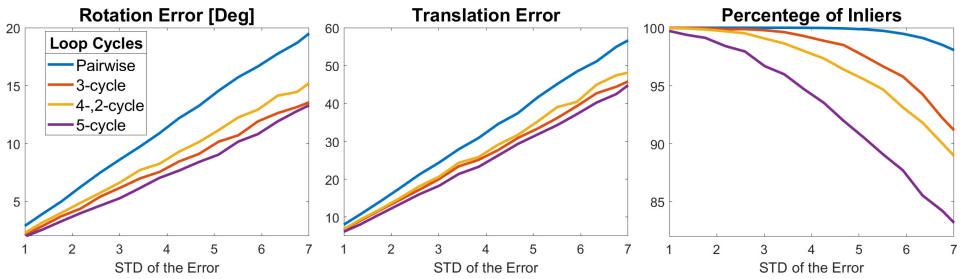


Figure 4: *We aim at finding the transformations between five 3D scans. We consider four different approaches: 1)* `Pairwise` *which uses only the technique of Sec. 2.2; 2)* `3-cycle` *that uses only the method in Sec. 2.3 (compute $\mathcal{S}_1$ to $\mathcal{S}_3$ and $\mathcal{S}_3$ to $\mathcal{S}_5$); 3)* `4-,2-cycle` *that uses the method in Sec. 2.4 from $\mathcal{S}_1$ to $\mathcal{S}_4$ and the one in Sec. 2.2 from $\mathcal{S}_4$ to $\mathcal{S}_5$; and, finally, 4)* `5-cycle` *that uses only the method in Sec. 2.5.*

of the points from the camera center, to simulate a real 3D sensor, and the following methods:

- `Pairwise`: in which we use the method of Sec. 2.2 to compute individual 3D registrations from $\mathcal{S}_1$ to $\mathcal{S}_2$, $\mathcal{S}_2$ to $\mathcal{S}_3$, $\mathcal{S}_3$ to $\mathcal{S}_4$, and $\mathcal{S}_4$ to $\mathcal{S}_5$.
- `3-cycle`: method in Sec. 2.3 to compute transformations between $\mathcal{S}_1$, $\mathcal{S}_2$, & $\mathcal{S}_3$ and $\mathcal{S}_3$, $\mathcal{S}_4$, & $\mathcal{S}_5$;
- `4-,2-cycle`: method in Sec. 2.4 to compute the 3D registrations from $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, & $\mathcal{S}_4$, and the method in Sec. 2.2 to compute the 3D registrations from $\mathcal{S}_4$ to $\mathcal{S}_5$; and
- `5-cycle`: method in Sec. 2.5 to compute all the transformation from $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, $\mathcal{S}_4$, and $\mathcal{S}_5$.

The minimal solvers were used in the RANSAC framework. A fixed number of 1000 RANSAC iterations was used, with no adaptive stopping criterion. A point distance of 50 units was used for the inlier counting. The registration from $\mathcal{S}_1$ to $\mathcal{S}_5$ is computed by multiplying each of the individual transformations from $\mathcal{S}_1$ to $\mathcal{S}_5$.

We show the angular rotation & translation errors and the percentage inliers in Fig. 4. For each level of noise, 10^3 randomly generated trials were used. These results show that the $n-$cycle solvers reduce the overall error in the estimation of the rotation and translation parameters. While the $5-$cycle gives the lowest rotation and translation error, it also achieves the lowest number of inliers.

5.2. Real Experiments

For real experiments, we use three sequences from the TUM dataset [39] that come with the ground-truth positions of the cameras (`freiburg1_room`, `freiburg1_xyz`, and `freiburg2_desk` sequences). We extract and match features using SURF [1] on the RGB images, and get the associated 3D points from the correspondent points in the Depth image. We start by analyzing the performance of the individual solvers separately and, then, we show their application in a large sequence using the pose graph and motion averaging discussed in Sec. 4.

Performance of the minimal solvers: From the dataset, we get sequences of 5 scans with loop cycles (i.e. sets of scans with enough correspondences between $\mathcal{S}_i$ & $\mathcal{S}_j$, to compute the poses using the respective minimal solvers). For each set of 5 scans, we compute the 3D registrations of all the 5 scans using the `Pairwise`, `3-cycle`, `4-,2-cycle`, and `5-cycle` methods in a RANSAC framework, similar to what was done in the evaluation of the proposed solvers in the previous subsection. A fixed number of 2000 RANSAC iterations was used for all the methods, with no adaptive stopping criterion. A point distance of 10[cm] was used as the threshold for the inlier counting. After getting the solutions, the inliers from all the different four alternatives are injected in a non-minimal pairwise 3D registration refinement method [45], to compute the cameras' relative position from $\mathcal{S}_1$ to $\mathcal{S}_2$, $\mathcal{S}_2$ to $\mathcal{S}_3$, $\mathcal{S}_3$ to $\mathcal{S}_4$, and $\mathcal{S}_4$ to $\mathcal{S}_5$.

The rotation and translation errors in the transformation from $\mathcal{S}_1$ to $\mathcal{S}_5$ (given by multiplying each of the pairwise transformations matrices from $\mathcal{S}_1$ to $\mathcal{S}_5$) are shown in Tab. 4. The n-cycle methods generally outperforms the pairwise technique. The `3-cycle` performs slightly better than the `4-,2-cycle`. The `5-cycle` solver produces better results in terms of rotation errors. In addition, in Tab. 4 we also show the number of times that the $n-$cycles outperforms the Pairwise technique.

TUM sequences: We get 100 3D scans from the three sequences, and define a graph according to Sec. 4. The total number of edges in the pose graph for the sequences `freiburg1_room`, `freiburg1_xyz`, and `freiburg2_desk` are 435, 1751, and 687, respectively. The number of n-cycle loops generated from each pose

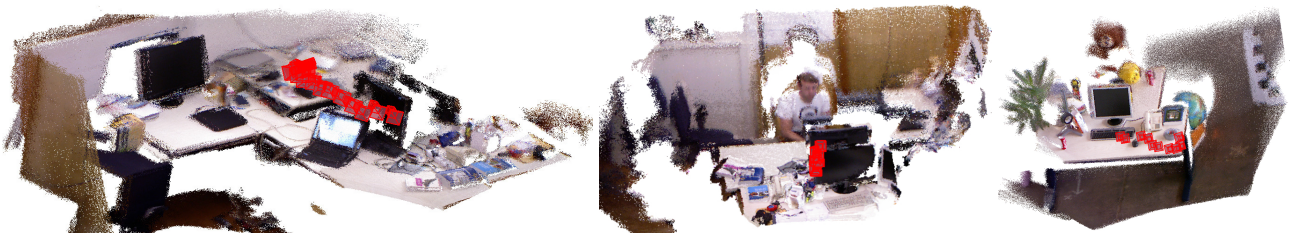


Figure 5: *Results for the 3D point-cloud registration, using the TUM RGB-D data-set [39]. We use three different sequences of 3D scans,* `freiburg1_room` *(at the left),* `freiburg1_xyz` *(at the center), and* `freiburg2_desk` *(at the right) and the method described in the paper to compute the relative transformations between the cameras. In these figures we show the registration of 100 RGB-D scans.*

	Errors		$n-$**cycles better than Pairwise**		$n-$**cycles equal to the Pairwise**	
Method	**Rot.**	**Tran.**	**Rot.**	**Tran.**	**Rot.**	**Tran.**
`Pairwise`	0.90	2.53	—	—	—	—
`3-cycle`	0.80	2.44	53%	48%	30%	32%
`4-,2-cycle`	0.80	2.47	46%	36%	36%	39%
`5-cycle`	0.77	2.60	63%	46%	8%	9%

Table 4: *Mean errors for the rotation (in degrees), translation (centimeters), and the number of times that the $n-$cycles outperforms the Pairwise technique*[3]*, using mini sequences of 5 3D scans in the TUM dataset.*

Data-Set	**Pairwise**	**3-Cycle**	**4-Cycle**	**5-Cycle**
`freiburg1_room`	58	1	1	74
`freiburg1_xyz`	57	0	1	338
`freiburg2_desk`	53	2	2	124

(a) *#n-cycle loops in the 100 3D scans.*

Data-Set	**Rotation** [deg]	**Translation** [cm]
`freiburg1_room`	1.96	4.52
`freiburg1_xyz`	0.740	2.44
`freiburg2_desk`	1.33	2.26

(b) *Errors in the estimation of the transformation parameters.*

Table 5: *Results obtained for the data-sets tested in this paper, i.e. 3D registrations shown in Fig. 5. (a) shows the the number of edges covered by each of the solvers, and (b) presents the average of the rotation and translation errors.*

graph is shown in Tab. 5(a).

After getting the poses on the pose graph using the proposed solvers in the RANSAC framework, we use the rotation averaging framework [8] to compute the final rotation matrices for all the cameras. After getting the rotations from the sequences, we get the corresponding translation parameters that satisfy the 3D point correspondences, using a standard least squares minimization method. The errors in the relative poses w.r.t. the ground-truth are shown in Tab. 5(b). The final registered scans are shown in Fig. 5.

6. Discussion

The main contribution of this paper is to show that one can jointly compute the pose of the cameras in $n-$cycles using the minimal number of point correspondences. In contrast to pairwise methods, the proposed approach uses only a fewer point correspondences. For example, computing the poses of 4 cameras in 4-cycles would only require 7 point correspondences, while the pairwise methods would require a minimum of 9 correspondences (3 between every camera pair). This may come as a surprise to many of us, since we assume that we need a minimum of 3 point correspondences for registering two scans. Actually, the 3-point relative pose solver for 3D cameras is not a minimal solution. It is only a near-minimal solution. To be precise, we actually need only $2\frac{1}{3}$ point correspondences to register two scans if we count the number of pose variables and number of constraints from point correspondences. Thus we can see that for obtaining 4 camera poses (assuming one of the cameras as the reference frame), our method only requires $3 \times 2\frac{1}{3} = 7$ point matches. This implies that our $n-$cycle solvers are exactly minimal, and not near minimal ones.

The proposed solvers provide alternate ways to obtain relative poses for pairs of cameras, in addition to standard pairwise methods, and this can be very beneficial in pose graph refinement or any motion averaging framework [8]. We observed that it is not practically feasible to solve the $n-$cycle solver when $n > 5$.

[3]Equal in the table means that the differences in the errors computed by the $n-$cycles and pairwise are less than 10^{-4}[deg] and 10^{-3}[mm].

Acknowledgements

This work was partially supported by the Portuguese Foundation for Science and Technology (FCT), project UID/EEA/50009/2019, National Science Foundation (NSF) grant IIS 1764071, and by the Swedish Foundation for Strategic Research (SSF), project COIN. We thank the reviewers and ACs for valuable feedback.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In *European Conf. Computer Vision (ECCV)*, pages 404–417, 2006.

[2] Florian Bernard, Frank R. Schmidt, Johan Thunberg, and Daniel Cremers. A combinatorial solution to non-rigid 3D shape-to-image matching. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1436–1445, 2017.

[3] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 14(2):239–256, 1992.

[4] Uttaran Bhattacharya, Sumit Veerawal, and Venu Madhav Govindu. Fast multiview 3D scan registration using planar structures. In *Int'l Conf. 3D Vision (3DV)*, pages 548–556, 2017.

[5] Alvaro Parra Bustos, Tat-Jun Chin, Anders Eriksson, Hongdong Li, and David Suter. Fast rotation search with stereographic projections for 3D registration. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 38(11):2227–2240, 2016.

[6] Federico Camposeco, Andrea Cohen, Marc Pollefeys, and Torsten Sattler. Hybrid camera pose estimation. In *IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 136–144, 2018.

[7] Federico Camposeco, Torsten Sattler, and Marc Pollefeys. Minimal solvers for generalized pose and scale estimation from two rays and one point. In *European Conf. Computer Vision (ECCV)*, pages 202–218, 2016.

[8] Avishek Chatterjee and Venu Madhav Govindu. Efficient and robust large-scale rotation averaging. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 521–528, 2013.

[9] Changhyun Choi, Alexander J. B. Trevor, and Henrik I. Christensen. RGB-D edge detection and edge-based registration. In *IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, pages 1568–1575, 2013.

[10] Gil Elbaz, Tamar Avraham, and Anath Fischer. 3D point cloud registration for localization using a deep neural network auto-encoder. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2472–2481, 2017.

[11] Felix Endres, Jurgen Hess, Nikolas Engelhard, Jurgen Sturm, Daniel Cremers, and Wolfram Burgard. An evaluation of the RGB-D SLAM system. In *IEEE Int'l Conf. Robotics and Automation (ICRA)*, pages 1691–1696, 2012.

[12] Felix Endres, Jurgen Hess, Jurgen Sturm, Daniel Cremers, and Wolfram Burgard. 3-D mapping with an RGB-D camera. *IEEE Trans. Robotics (T-RO)*, 30(1):177–187, 2014.

[13] 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.

[14] Friedrich Fraundorfer, Petri Tanskanen, and Marc Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In *European Conf. Computer Vision (ECCV)*, pages 269–282, 2010.

[15] Venu Madhav Govindu and Pooja A. On averaging multiview relations for 3D scan registration. *IEEE Trans. Image Processing (T-IP)*, 23(3):1289–1302, 2014.

[16] Maciej Halber and Thomas Funkhouser. Fine-to-coarse global registration of RGB-D scans. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 6660–6669, 2017.

[17] Tong Ke and Stergios I. Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 4618–4626, 2017.

[18] Marc Khoury, Qian-Yi Zhou, and Vladlen Koltun. Learning compact geometric features. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 153–161, 2017.

[19] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2969–2976, 2011.

[20] Laurent Kneip, Roland Siegwart, and Marc Pollefeys. Finding the exact rotation between two images independently of the translation. In *European Conf. Computer Vision (ECCV)*, pages 696–709, 2012.

[21] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Automatic generator of minimal problem solvers. In *European Conf. Computer Vision (ECCV)*, pages 302–315, 2008.

[22] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Polynomial eigenvalue solutions to minimal problems in computer vision. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 7(34):1381–1393, 2012.

[23] Viktor Larsson, Kalle Astrom, and Magnus Oskarsson. Polynomial solvers for saturated ideals. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 2307–2316, 2017.

[24] Gim Hee Lee. A minimal solution for non-perspective pose estimation from line correspondences. In *European Conf. Computer Vision (ECCV)*, pages 170–185, 2016.

[25] Bo Li, Lionel Heng, Gim Hee Lee, and Marc Pollefeys. A 4-point algorithm for relative pose estimation of a calibrated camera with a known relative rotation angle. In *IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, pages 1595–1601, 2013.

[26] Hongdong Li. A simple solution to the six-point two-view focal-length problem. In *European Conf. Computer Vision (ECCV)*, pages 200–213, 2006.

[27] Hongdong Li. Multi-view structure computation without explicitly estimating motion. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2777–2784, 2010.

[28] Hongdong Li and Richard Hartley. Five-point motion estimation made easy. In *Int'l Conf. Pattern Recognition (ICPR)*, volume 1, pages 630–633, 2006.

[29] Hongdong Li and Richard Hartley. The 3D-3D registration problem revisited. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 1–8, 2007.

[30] Chen Liu, Jiaye Wu, and Yasutaka Furukawa. FloorNet: A unified framework for floorplan reconstruction from 3D scans. In *European Conf. Computer Vision (ECCV)*, pages 203–219, 2018.

[31] Yan Lu and Dezhen Song. Robust RGB-D odometry using point and line features. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 3934–3942, 2015.

[32] Lingni Ma, Christian Kerl, Jorg Stuckler, and Daniel Cremers. CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM. In *IEEE Int'l Conf. Robotics and Automation (ICRA)*, pages 1285–1291, 2016.

[33] Lingni Ma, Jurg Stuckler, Christian Kerl, and Daniel Cremers. Multi-view deep learning for consistent semantic mapping with rgb-d cameras. In *IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, pages 598–605, 2017.

[34] Pedro Miraldo, Tiago J. Dias, and Srikumar Ramalingam. A minimal closed-form solution for multi-perspective pose estimation using points and lines. In *European Conf. Computer Vision (ECCV)*, pages 490–507, 2018.

[35] Andriy Myronenko and Xubo Song. Point set registration: Coherent point drift. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 32(12):2262–2275, 2010.

[36] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE Int'l Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[37] David Nister. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI)*, 26(6):756–770, 2004.

[38] David Nister and Frederik Schaffalitzky. Four points in two or three calibrated views: Theory and practice. *Int'l J. Computer Vision (IJCV)*, 67(2):211–231, 2006.

[39] Technical University of Munich (TUM). RGB-D SLAM dataset and benchmark. `https://vision.in.tum.de/data/datasets/rgbd-dataset`. Accessed: 2018-11-13.

[40] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 143–152, 2017.

[41] Mikael Persson and Klas Nordberg. Lambda twist: An accurate fast robust perspective three point (P3P) solver. In *European Conf. Computer Vision (ECCV)*, pages 334–349, 2018.

[42] Srikumar Ramalingam and Yuichi Taguchi. A theory of minimal 3D point to 3D plane registration and its generalization. *Int'l J. Computer Vision (IJCV)*, 102(1-3):73–90, 2013.

[43] Olivier Saurer, Pascal Vasseur, Cedric Demonceaux, and Friedrich Fraundorfer. A homography formulation to the 3pt plus a common direction relative pose problem. In *Asian Conf. Computer Vision (ACCV)*, pages 288–301, 2015.

[44] Johannes L. Schonberger, Marc Pollefeys, Andreas Geiger, and Torsten Sattler. Semantic visual localization. In *IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 6896–6906, 2018.

[45] Peter H. Schonemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[46] Aleksandr V. Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-ICP. In *Robotics: Science and Systems (RSS)*, 2009.

[47] Miroslava Slavcheva, Maximilian Baust, Daniel Cremers, and Slobodan Ilic. KillingFusion: Non-rigid 3D reconstruction without correspondences. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 5474–5483, 2017.

[48] Henrik Stewenius, David Nister, Magnus Oskarsson, and Kalle Astrom. Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision (OMNIVIS)*, 2005.

[49] Jurgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, pages 573–580, 2012.

[50] Chris Sweeney, Laurent Kneip, Tobias Höllerer, and Matthew Turk. Computing similarity transformations from only image correspondences. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 3305–3313, 2015.

[51] Gary K. L. Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C. Langbein, Yonghuai Liu, David Marshall, Ralph R. Martin, Xian-Fang Sun, and Paul L. Rosin. Registration of 3D point clouds and meshes: A survey from rigid to nonrigid. *IEEE Trans. Visualization and Computer Graphics (T-VCG)*, 19(7):1199–1217, 2013.

[52] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3D. In *IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 3887–3896, 2018.

[53] Jonathan Ventura, Clemens Arth, and Vincent Lepetit. An efficient minimal solution for multi-camera motion. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 747–755, 2015.

[54] Jonathan Ventura, Clemens Arth, Gerhard Reitmayr, and Dieter Schmalstieg. A minimal solution to the generalized pose-and-scale problem. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 422–429, 2014.

[55] Ping Wang, Guili Xu, Zhengsheng Wang, and Yuehua Cheng. An efficient solution to the perspective-three-point pose problem. *Computer Vision and Image Understanding (CVIU)*, 166:81–87, 2018.

[56] Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In *IEEE Int'l Conf. Computer Vision (ICCV)*, pages 1457–1464, 2013.

[57] Anestis Zaganidis, Li Sun, Tom Duckett, and Grzegorz Cielniak. Integrating deep semantic segmentation into 3-D point cloud registration. *IEEE Robotis and Automation Letters (RA-L)*, 3(4):2018, 2942-2949.

[58] Huayi Zeng, Jiaye Wu, and Yasutaka Furukawa. Neural procedural reconstruction for residential buildings. In *European Conf. Computer Vision (ECCV)*, pages 759–775, 2018.

[59] Michael Zollhofer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, and Marc Stamminger. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Trans. Graph.*, 33(4):156:1–156:12, 2014.