Minimal Solutions to Generalized Three-View Relative Pose Problem

Yaqing Ding[1†], Chiang-Heng Chien[2†], Viktor Larsson[1], Karl Åström[1], Benjamin Kimia[2]
[1]Centre for Mathematical Sciences, Lund University
[2]School of Engineering, Brown University

Abstract

For a generalized (or non-central) camera model, the minimal problem for two views of six points has efficient solvers. However, minimal problems of three views with four points and three views of six lines have not yet been explored and solved, despite the efforts from the computer vision community. This paper develops the formulations of these two minimal problems and shows how state-of-the-art GPU implementations of Homotopy Continuation solver can be used effectively. The proposed methods are evaluated on both synthetic and real datasets, demonstrating that they are fast, accurate and that they improve on structure from motion estimations, when employed in an hypothesis and test setting.

1. Introduction

In both the central and the non-central camera models, an image point is in correspondence to a ray of a 3D point that projects to that image point. In the central camera model, each such ray goes through a common point, *i.e.*, the camera centre, whereas in the non-central camera model, two such rays do not always intersect. One example of a non-central camera is the catadioptric camera, where one or more lenses and/or mirrors are combined in an optical system, [17]. Another example is a rig of multiple cameras, [20]. A typical use-case is a car or drone equipped with several cameras. Here it is possible to use all of the cameras as one generalized sensor. Generalized cameras offer several advantages over the standard pinhole camera for relative pose estimation. As shown in previous works [20, 25, 10, 15], structure from motion estimation with a generalized camera also gives the scale of the translation as well as the scale of the 3D model. In addition, since the system consists of multiple cameras, it is possible to configure the location of each camera so that a maximal field-of-view can be achieved to avoid blind spots. This, in turn, enables more robust image feature detection and matching, which is a crucial procedure in relative pose estimation. Fig. 1 shows an example of a multi-camera system consisting of four pinhole cameras mounted on a car for relative pose estimation.

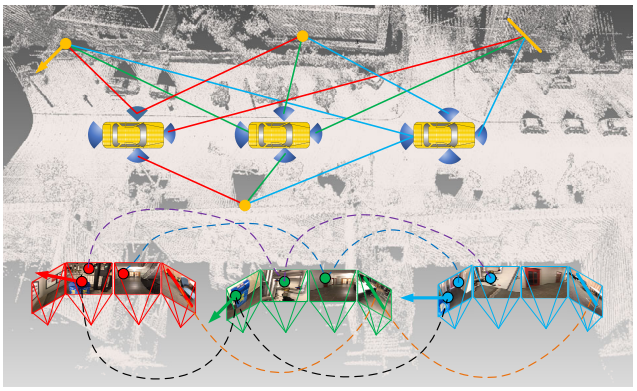


Figure 1. An example of a multi-camera system with four standard pinhole cameras mounted on a car.

Unlike the standard pinhole camera, the light rays from a generalized camera do not intersect at a single center of projection. This property makes it non-trivial to estimate the relative pose using a generalized camera. Pless [20] first formulated the light rays in a generalized camera as Plücker vectors and defined the generalized essential matrix. Based on the work of Pless, several solutions to the two-view generalized relative pose estimation have been proposed [26, 10, 25]. Among these solutions, the minimal solver with up to 64 solutions was proposed in [25]. This solver only needs 6 point correspondences which can significantly reduce the number of RANSAC [7] iterations than other solvers. Although the solvers to the two-view generalized relative pose problem are already mature, there are no solutions to the three-view case as the generalized three-view relative pose problem is extremely difficult to be formulated and solved. In general, three-view relative pose estimation needs fewer point correspondence and suffers from fewer degenerate cases than the two-view case. Moreover, three-view relative pose can be estimated only using the line correspondences. Hence, it has a great po-

† Equal contribution.

	Point •	Line ▬	Quiver •⟶	No. of Solutions
4-point	4	0	0	583
6-line	0	6	0	600
3-quiver	0	0	3	1008
2P3L	2	3	0	920
2P1L1Q	2	1	1	1472
4L1Q	0	4	1	1976
2L2Q	0	2	2	2240

Table 1. Possible cases and number of solutions for the generalized three-view relative pose problem. Each row shows the number of necessary features and the number of solutions. We mainly focus on the first two cases in this paper since they are more practical and relevant.

tential and can provide new ways to solve the relative pose problem.

The generalized three-view relative pose problem has 12 degrees of freedom (6 for the two rotations and 6 for the two translations). It is well known that the most commonly used features for relative pose estimation are points, lines and quivers, where a quiver consists of one point with tangent direction (or one point with one line passing through this point). According to the analyses in [5, 6], point, line and quiver in three views can provide 3,2 and 4 constraints, respectively. In Table 1, we show that there are 7 possible minimal cases for generalized three-view relative pose estimation using the combination of the three features. We only assume that there is one line passing through the point, there will be much more cases if there are several lines passing through one point.

In this paper, to estimate the generalized three-view relative pose, we focus on the first two cases in Table 1. We show detailed discussions on the formulations and the number of solutions for each case. The main contributions of this paper include:

- We solve the generalized three-view relative pose problem using 4 points or 6 lines. Specifically, detailed problem formulation for the two cases are given, which are then converted into solving a squared system of polynomials.

- This allows us to develop a solver using HC (homotopy continuation) based on the state-of-the-art GPU-HC solver [2, 3]. This significantly improves the efficiency over the existing modern HC software.

- In extensive synthetic and real experiments we show an improvement of accuracy over the state-of-the-art two-view 6-point solver. The proposed 4-point solver is able to deal with semi-pure translation, which is degenerated for the two-view case.

2. Related Work

Three-view Relative Pose. Estimating the calibrated three-view relative pose from standard pinhole cameras has been well studied in the literature. In general, this problem has 11 degrees of freedom (DOF), and can be solved using 4 points [19, 21] or 6 lines [9]. Note that, both 4 points and 6 lines are over-determined for standard pinhole cameras, which makes it difficult to derive stable solvers. The only available solver for the 4-point problem was shown in [11], where the authors added an extra parameter to solve a relaxed version of the problem. To the best of our knowledge, there are no publicly available solvers for the 6-line problem. It was shown in [9] that there might be up to 600 solutions to three-view relative pose with 6 lines using pinhole cameras. While the problem of estimating the three-view relative pose from standard pinhole cameras has received a lot of attention, this does not hold for the generalized three-view relative pose problem. There is only a discussion on the possible number of solutions using 4 points in three generalized views [25].

Solving Polynomials. Most of the minimal problems in computer vision can be transformed into solving a system of polynomial equations. A straightforward way of solving a polynomial system is to use the Gröbner basis method [4] and generate a specific solver using an automatic generator, *e.g.*, [12, 13, 14]. The Gröbner basis method and related automatic generators have been successfully used in many computer vision problems [14], showing promising results in terms of stability and efficiency. However, it's difficult to find a stable solution for large scale problems using the Gröbner basis technique, since we need to apply Gauss-Jordan elimination and eigenvalue decomposition to large size of matrices. Alternatively, homotopy continuation (HC) [24, 8], one of the promising numerical solver for polynomial equations, is able to handle systems that are out of the reach of the Gröbner basis methods. However, HC algorithms are known to be slow, impractical for many multiview geometry problems when a RANSAC loop is involved, *e.g.* trifocal relative pose estimation requires over 600 (ms) per correspondence in [6]. This issue has been tackled by GPU-HC [2, 3] where the design of the GPU implementation accelerate the speed in orders of magnitude, giving efficient and accurate estimations for many minimal problems. As there is no paper using GPU-HC to solve minimal problems in practice, this paper stands out as the first approach to leverage it as our solvers.

3. Problem Statement

As SO(3) is three-dimensional, and we set the first generalized camera to $[\mathbf{I} \mid \mathbf{0}]$, the parameter space of camera configurations for n-view case has dimension $6n-6$. Since one point correspondence in three views gives three constraints,

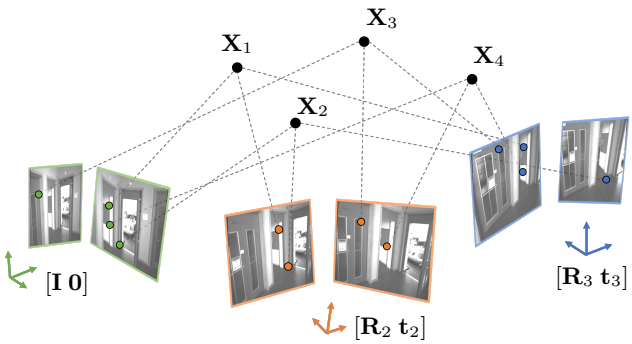


Figure 2. The minimal problem of 4 points seen in three generalized cameras. In this example the generalized cameras are visualized as multi-camera systems.

and one line correspondence in three views gives two constraints, we can find that 4 points and 6 lines in three views are two minimal cases.

3.1. The 4-point Case

Assuming that four points $\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4$ in the 3D space are observed by three generalized cameras (Figure 2). Note that, the four points $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4\}$ can't be viewed from a single pinhole camera, *i.e.*, the points need to be sampled from at least two cameras. Otherwise, the problem will be degenerated into the trifocal relative pose from a standard pinhole camera. The image points of the four 3D points in each view can be written as $\mathbf{x}_{1i}, \mathbf{x}_{2i}, \mathbf{x}_{3i}, \mathbf{x}_{4i}$, $i \in \{1, 2, 3\}$. On the other hand, the rotation matrices and translation vectors that align the image points with the world coordinate are defined as $\{\mathbf{A}_{ki}, \boldsymbol{\tau}_{ki}\}$. We formulate the depths of the image points in each generalized view as $\lambda_{1i}, \lambda_{2i}, \lambda_{3i}, \lambda_{4i}$, $i \in \{1, 2, 3\}$. Due to the rigid motion, the distance between any two points in each view should be the same. Hence, we have

$$\begin{aligned} &\|\lambda_{11}\mathbf{A}_{11}\mathbf{x}_{11}+\boldsymbol{\tau}_{11} - \lambda_{21}\mathbf{A}_{21}\mathbf{x}_{21} - \boldsymbol{\tau}_{21}\|^2 \\ &\quad = \|\lambda_{12}\mathbf{A}_{12}\mathbf{x}_{12} + \boldsymbol{\tau}_{12} - \lambda_{22}\mathbf{A}_{22}\mathbf{x}_{22} - \boldsymbol{\tau}_{22}\|^2 \\ &\quad \cdots \\ &\|\lambda_{31}\mathbf{A}_{31}\mathbf{x}_{31}+\boldsymbol{\tau}_{31} - \lambda_{41}\mathbf{A}_{41}\mathbf{x}_{41} - \boldsymbol{\tau}_{41}\|^2 \\ &\quad = \|\lambda_{43}\mathbf{A}_{43}\mathbf{x}_{43} + \boldsymbol{\tau}_{43} - \lambda_{44}\mathbf{A}_{44}\mathbf{x}_{44} - \boldsymbol{\tau}_{44}\|^2 \end{aligned} \tag{1}$$

In this case, we obtain 12 polynomials in 12 unknowns $\{\lambda_{1i}, \lambda_{2i}, \lambda_{3i}, \lambda_{4i}\}$. Based on [1], we find that there are up to 583 solutions. Once the depth parameters are computed, we let

$$\mathbf{X}_{ki} = \lambda_{ki}\mathbf{A}_{ki}\mathbf{x}_{ki} + \boldsymbol{\tau}_k, \tag{2}$$

where $k \in \{1, 2, 3, 4\}$ and $i \in \{1, 2, 3\}$. We have the following constraints

$$\mathbf{X}_{k2} = \mathbf{R}_2\mathbf{X}_{k1} + \mathbf{t}_2. \tag{3}$$

By eliminating the translation parameters, we have

$$\mathbf{X}_{k2} - \mathbf{X}_{12} = \mathbf{R}_2(\mathbf{X}_{k1} - \mathbf{X}_{11}),\ k \in \{2, 3, 4\}. \tag{4}$$

Hence, the rotation matrix $\mathbf{R}_2$ can be found by

$$\begin{aligned} \mathbf{R}_2 &= \mathbf{B}_2\mathbf{B}_1^{-1}, \\ \mathbf{B}_2 &= [\mathbf{X}_{22} - \mathbf{X}_{12}, \mathbf{X}_{32} - \mathbf{X}_{12}, \mathbf{X}_{42} - \mathbf{X}_{12}], \\ \mathbf{B}_1 &= [\mathbf{X}_{21} - \mathbf{X}_{11}, \mathbf{X}_{31} - \mathbf{X}_{11}, \mathbf{X}_{41} - \mathbf{X}_{11}]. \end{aligned} \tag{5}$$

Then the translation can be computed using (3). Similarly, we can obtain $\mathbf{R}_3, \mathbf{t}_3$.

3.2. The 6-line Case

Assuming that six lines $\mathbf{L}_1, \mathbf{L}_2, \cdots, \mathbf{L}_6$ in the 3D space are observed by three generalized cameras. The projections of the lines in each view can be written as $\boldsymbol{l}_{1i}, \boldsymbol{l}_{2i}, \cdots, \boldsymbol{l}_{6i}$, $i \in \{1, 2, 3\}$. The line and the camera center define a plane which passes through the origin of the local camera coordinate. In this case, the plane in each camera coordinate can be formulated as

$$\boldsymbol{\pi}_{ki} = [\boldsymbol{l}_{ki}; 0]. \tag{6}$$

In addition, we let the 4×4 transformation matrices that align the planes with the world coordinate as

$$\mathbf{C}_{ki} = \begin{bmatrix} \mathbf{A}_{ki} & \boldsymbol{\tau}_{ki} \\ \mathbf{0} & 1 \end{bmatrix}. \tag{7}$$

The planes in three views which are defined by one 3D line should intersect at this line, in this case, we can obtain the following constraints

$$\begin{aligned} \mathrm{rank}[\mathbf{P}_1^\top\mathbf{C}_k^\top\boldsymbol{\pi}_{k1} \mid \mathbf{P}_2^\top\mathbf{C}_k^\top\boldsymbol{\pi}_{k2} \mid \mathbf{P}_3^\top\mathbf{C}_k^\top\boldsymbol{\pi}_{k3}] \le 2, \\ k \in \{1, 2, 3, 4, 5, 6\}, \end{aligned} \tag{8}$$

where matrix $\mathbf{P}_i$ is defined as

$$\mathbf{P}_i = \begin{bmatrix} \mathbf{R}_i & \boldsymbol{t}_i \\ \mathbf{0} & 1 \end{bmatrix}, i \in \{1, 2, 3\}. \tag{9}$$

Based on (9), we can obtain $6 * \mathrm{C}_3^4 = 24$ polynomials in the rotation and translation parameters. If we consider the constraints that only contain the rotation components, we have 6 constraints in the rotation parameters

$$\begin{aligned} \det[\mathbf{A}_{k1}^\top\boldsymbol{l}_{k1} \mid \mathbf{R}_2^\top\mathbf{A}_{k2}^\top\boldsymbol{l}_{k2} \mid \mathbf{R}_3^\top\mathbf{A}_{k3}^\top\boldsymbol{l}_{k3}] = 0, \\ k \in \{1, 2, 3, 4, 5, 6\}. \end{aligned} \tag{10}$$

In order to find the motion parameters, we introduce two formulations for the rotation matrix.

Cayley parameterization. The major advantage lies in this formulation is that there remain only three variables for each rotation matrix:

$$\mathbf{R}_2 = \tfrac{1}{\kappa_2}\begin{bmatrix} 1+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_3) & 2(q_1q_3+q_2) \\ 2(q_1q_2+q_3) & 1-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_1) \\ 2(q_1q_3-q_2) & 2(q_2q_3+q_1) & 1-q_1^2-q_2^2+q_3^2 \end{bmatrix}$$

$$\mathbf{R}_3=\frac{1}{\kappa_3}\begin{bmatrix}1+q_4^2-q_5^2-q_6^2 & 2(q_4q_5-q_6) & 2(q_4q_6+q_5)\\ 2(q_4q_5+q_6) & 1-q_4^2+q_5^2-q_6^2 & 2(q_5q_6-q_4)\\ 2(q_4q_6-q_5) & 2(q_5q_6+q_4) & 1-q_4^2-q_5^2+q_6^2\end{bmatrix},$$

where $\kappa_2 = 1+q_1^2+q_2^2+q_3^2$, $\kappa_3 = 1+q_4^2+q_5^2+q_6^2$, note that this formulation is unstable in case of Cayley degenerate rotations ($\kappa_2, \kappa_3 = 0$). However, the probability for the degenerate case is quite low in practice, and this formulation has been successfully used in many computer vision problems [13]. Since the constraints (10) are homogenous in the rotation matrices, the scale factors κ_2, κ_3 can be omitted. In this case, we can obtain 6 quartic polynomials in 6 unknowns $\{q_1, q_2, ..., q_6\}$. The monodromy solver in [1] shows that there are up to 600 solutions. Once the rotation parameters are computed, we need to substitute them into (8) and obtain six linear equations in the translation parameters. The translation $\mathbf{t}_2, \mathbf{t}_3$ can be uniquely obtained.

Homogeneous rotation matrix. On the other hand, since constraints (8) are invariant to the scale of the rotation matrices, we can use 8 parameters to formulate the rotation up to a scale factor (assuming the first element to be 1)

$$\mathbf{R}_2 = \begin{bmatrix}1 & r_2 & r_3\\ r_4 & r_5 & r_6\\ r_7 & r_8 & r_9\end{bmatrix}, \ \mathbf{R}_3 = \begin{bmatrix}1 & r_{11} & r_{12}\\ r_{13} & r_{14} & r_{15}\\ r_{16} & r_{17} & r_{18}\end{bmatrix}.$$

The elements $\{r_2, r_2, ..., r_9\}$ should satisfy the following constraints

$$\begin{aligned} r_2r_8 + r_3r_9 + r_7 = 0,\ r_2r_5 + r_3r_6 + r_4 = 0,\\ r_4r_6 + r_7r_9 + r_3 = 0,\ r_4r_5 + r_7r_8 + r_2 = 0,\\ r_5^2 + r_6^2 - r_7^2 - 1 = 0. \end{aligned}$$

In this case, we can obtain 16 quadratic polynomials in 16 unknowns $\{r_2, r_3, ..., r_{18}\}$. Similar to the Cayley parameterization, this formulation is unstable in case of degenerate rotations ($r_1, r_{10} = 0$). Although such degenerate cases can be avoided by using unit quaternion formulation, it will result in more parameters than the Cayley parameterization and the system of polynomials has 2-fold symmetry. In order to derive more efficient solvers, we only discuss Cayley parameterization and homogeneous rotation matrix formulation in this paper.

Geometric Interpretation of (10). The line projection with the camera center define a plane, and this plane gives a line at infinity. Since the three lines in three views should correspond to the same 3D line, the three lines at infinity should intersect at the same point. In this case, we can obtain constraint (10).

3.3. Degenerate Case

A detailed analysis of the degeneracies for generalized two-view relative pose estimation has been shown in [10, 15]. The degenerated case is slightly different in generalized three-view relative pose problem. In our case, we found that if the multi-camera system undergoes pure translation ($\mathbf{R}_2 = \mathbf{R}_3 = \mathbf{I}$) with only intra-camera correspondences, our solvers can not recover the scale of the translation. However, if we only have $\mathbf{R}_2 = \mathbf{I}$ or $\mathbf{R}_3 = \mathbf{I}$, our solvers can still recover the translation with scale factor.

4. Polynomial System Solver

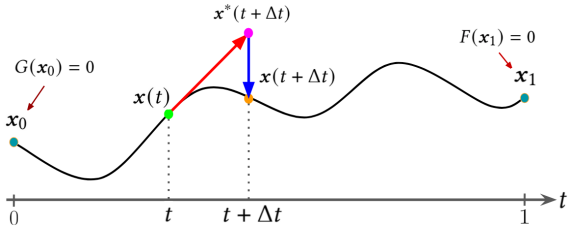


Figure 3. A track (curve) of a Homotopy Continuation algorithm showing $H(X,t)$ in black, along with one prediction (red) and one correction (blue).

Homotopy Continuation: The idea of Homotopy Continuation (HC) [18, 24] is to evolve the solutions of one polynomial system G, the "start system", to discover the solutions of another system F, the "target system". Let $X = (x_1, x_2, ..., x_M)$ represent M unknowns. Let $F(X)$ be a system of N polynomial equations $F = (f_1, f_2, ..., f_N)$. Let $G(x)$, $G = (g_1, g_2, ..., g_N)$ be the "start system" whose solutions are all known. The idea of HC is to construct a series of intermediate polynomial systems $H(X,t)$, $H = (H_1, H_2, ..., H_N)$; where $H(X,0) = G(X)$ and $H(X,1) = F(X)$, *e.g.*, via linear interpolation:

$$H(x,t) = (1-t)G(x) + tF(x), \qquad t \in [0,1]. \tag{11}$$

The basic idea is to find the solution of $H(X, t+\Delta t)$ from the solution of $H(X,t)$. Figure 3 illustrates the idea for one solution and one unknown. The black curve is the locus of the solution $X(t)$ of $H(X,t)$, the homotopy curve, where X_0 is the known solution of $G(X)$ and X_1 is the desired solution of $F(X)$. We track solution X_1 from X_0 in a number of small steps, each consisting of a prediction and a correction step. Prediction uses a first-order Taylor expansion to estitmate X at $t+\Delta t$ in the form of

$$X^*(t+\Delta t) = X(t) + \frac{dX}{dt}\Delta t, \tag{12}$$

where X^* is the first order estimation of $X(t+\Delta t)$. We obtain $\frac{dX}{dt}$ by differentiating $H(X(t),t)$, *i.e.*,

$$\frac{\partial H}{\partial X}\frac{dX}{dt} + \frac{\partial H}{\partial t} = 0 \longrightarrow \frac{dX}{dt} = -(\frac{\partial H}{\partial X})^{-1}\frac{\partial H}{\partial t}, \tag{13}$$

where $J = \frac{\partial H}{\partial x}$ is the $M \times N$ Jacobian of H wrt X, giving

$$X^*(t+\Delta t) = X(t) - (\frac{\partial H}{\partial X})^{-1}\frac{\partial H}{\partial t}\Delta t. \tag{14}$$

The first-order estimation of X^* from $X(t)$ is known as the *prediction* step (Figure 3) which can be further improved using a higher-order method such as a fourth order Runge-Kutta. Followed by a *correction* step which helps improve the prediction, Newton's update $X^*(t+\Delta t)$ to $\hat{X}(t+\Delta t)$ is used, *i.e.*,

$$H(X^*, t+\Delta t) + \frac{\partial H}{\partial X}(X^*, t+\Delta t)(\hat{X} - X^*) = 0, \quad (15)$$

giving the estimate $\hat{X}$ in the form of

$$\hat{X} = X^* - (\frac{\partial H}{\partial X})^{-1}(X^*, t+\Delta t)H(X^*, t+\Delta t). \quad (16)$$

The pairs of prediction and correction numerically evolve X_0 as the solution of $G(X)$ to X_1 as the solution of $F(X)$. HC algorithms find all the solutions (up to some approximation) with probability one given a good start system which can be provided easily by a monodromy solver [1]. In many computer vision applications, a good start system usually needs to be created once as multiple target systems correspond to different target parameters can be solved from the same start system under a RANSAC loop.

GPU-HC: The iterative nature of HC prevents it from being practically used. The GPU implementation in [2, 3] parallelizes *(i)* HC paths and *(ii)* the evaluations of $\frac{\partial H}{\partial X}$, $\frac{\partial H}{\partial t}$, and H, Equations (13) and (16) to enable HC efficiently solves problems whose complexity has far evaded a practical solution. We thus use GPU-HC as our minimal problem solver, giving promising results in our experiments.

GPU-HC Solution Refinement: Since GPU computation is memory-bound, extending numerical precision in GPU computation would significantly drag down efficiency. To obtain high numerical accuracy while avoiding slowdown, Levenberg–Marquardt algorithm using double precision is used, followed by GPU-HC using single precision for solution refinement. This refinement step can be done in CPU side efficiently as only partial GPU-HC solutions are necessary to be refined, *i.e.*, real solutions in the 6-line problem or real and positive solutions in the 4-point problem.

5. Experiments

To demonstrate that the proposed solvers work well in practice, we conduct extensive experiments on both synthetic and real-world data with multi-camera system configuration. Experiments are run on a 16-Core 3.6GHz Intel i7-11700K CPU and a NVIDIA Titan V GPU.

For both synthetic and real experiments, the solver's performance is evaluated by the following criteria:

$$\xi_{\mathbf{R}[\text{deg}]} = 2\arcsin\left(\frac{\|\mathbf{R}_{gt} - \mathbf{R}_e\|}{2\sqrt{2}}\right),$$
$$\xi_{\mathbf{t}[\text{deg}]} = 2\arcsin\left(\frac{1}{2}\left\|\frac{\mathbf{t}_e}{\|\mathbf{t}_e\|} - \frac{\mathbf{t}_g}{\|\mathbf{t}_g\|}\right\|\right),$$
$$\xi_{\text{Scale}} = \text{abs}(\|\mathbf{t}_e\|/\|\mathbf{t}_{gt}\| - 1),$$

where $\xi_{\mathbf{R}}$, $\xi_{\mathbf{t}}$, and ξ_{Scale} are the angle difference in rotation, direction difference in translation and relative scale error, respectively. $\mathbf{R}_{gt}, \mathbf{t}_{gt}$ are the ground truth rotation and translation, and $\mathbf{R}_e, \mathbf{t}_e$ are the estimated rotation and translation, respectively. All the norms are Frobenius norm. Note that, computing the rotation angle error and the translation direction error based on the $\arccos\left((\text{trace}(\mathbf{R}_{\text{gt}}^\top \mathbf{R}_{\text{e}}) - 1)/2\right)$ is simpler. However, it's not the best choice. For example, if the relative angle between the rotation matrices is small, $\left((\text{trace}(\mathbf{R}_{\text{gt}}^\top \mathbf{R}_{\text{e}}) - 1)/2\right)$ is close to unity in that case and the $\arccos$ is very sensitive to errors. Hence, it's better to compute angular errors using the $\arcsin$ formulation. Detailed comparisons between $\arcsin$ and $\arccos$ are shown in the supplementary material.

5.1. Numerical Accuracy of the Solvers

Although it has been reported in [2, 3] that GPU-HC provides promising and acceptable numerical accuracy in solving various minimal problems in computer vision, for the proposed problems to be usable the evaluation on polynomial residual is still necessary. Figure 4 top row shows the accuracy of (a) all GPU-HC converged solutions and (b) the converged solutions correspond to the ground truths, solving 6-line problem and 4-point problem using our synthetic dataset generating 10,000 real parameters for each problem. Both single precision (SP) and double precision (DP) are demonstrated, showing that the solver is able to provide less than 10^{-5} accurate results for the majority of the GPU-HC solutions. Nevertheless, if a refinement step is involved, significant shifts in the distributions of residuals can be observed, Figure 4 bottom row, where single precision solu-

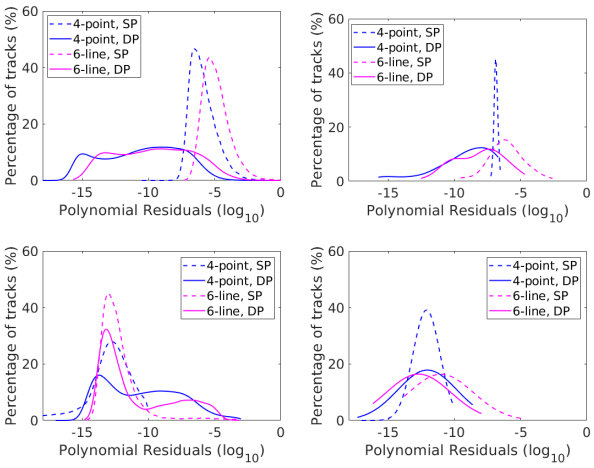


Figure 4. Numerical accuracy of GPU-HC from (a) all converged paths and (b) converged paths correspond to the true solution of the 4-points and 6-lines problems. **Top Row:** Original. **Bottom Row:** Refinement.

tions are nearly as accurate as the double precision.

5.2. Synthetic Evaluation

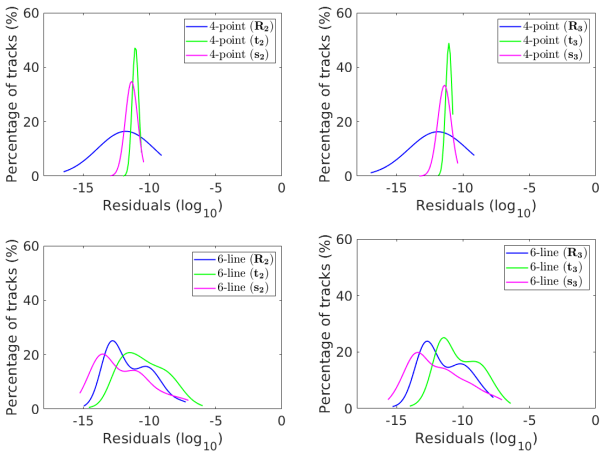


Figure 5. Relative errors with noise-free data under random points and movement. **Top Row:** The 4-point solver. **Bottom Row:** The 6-line solver.

In this section, we evaluate the numerical stability and the noise resilience of the proposed solvers. We simulated a multi-camera system with six cameras, and the focal length of each pinhole camera is set to 1000px. To generate the scene we uniformly sample 3D points from the box $[-2, 2] \times [-2, 2] \times [3, 6]$ in each camera's local coordinate system. Then based on two points we can generate lines in each camera. Finally these points and lines are transformed with a random but feasible rotation and translation.

Stability. We randomly generated 1000 noise-free instances with different transformations. Figure 5 top row shows the distribution of the $\log_{10}$ relative errors of the 4-point solver, and Figure 5 bottom row shows the distribution of the $\log_{10}$ relative errors of the 6-line solver, respectively. We can see that both solvers are quite stable.

Noise resilience. In this experiment, we focused on four practical motions: general motion, forward motion (along the z-axis), sideways motion (along the x-axis), and semi-pure translation ($\mathbf{R}_2 = \mathbf{I}$). Fig. 6 shows the performance of the proposed 4-point solver and 6-line solver under increasing image noise with different standard deviation. We can see that the 4-point solver has almost similar error distributions for the four motions. The only difference is that the translation direction error is slightly larger under sideways motion and forward motion. Overall, the performance of the 4-point solver is stable under different motions with increasing image noise. By contrast, the 6-line solver is more sensitive to image noise, especially the translation direction error. This is possibly due to the noise that we added to the end points. If the points are close to each other, very small noise in the end points may result in extremely large errors in the line direction. Since the 6-line solver is sensitivity to noise, we will focus on the performance of the 4-point solver in the real experiments.

Complexity analysis and running times. The success rate of finding the HC solutions correspond to the ground truths and the overall timings of our GPU-HC solver is shown in Table 3. Typically, using double precision would give a slowdown in speed compared to single precision, but there are times where double precision is faster than single precision, *e.g.*, the minimum GPU-HC time of 6-line problem. This is because the required number of HC steps is lesser when double precision is used which converges HC tracks faster. The speedup compared to Julia 8-core CPU HC solver in the table also demonstrate the benefit of using GPU-HC, which gives more than 150× speedup and 58× speedup for single precision of 6-line and 4-point problems, respectively, enabling real-time computation.

Success rate and accuracy under a RANSAC scheme. Table 2 shows the RANSAC experiment of the 4-point problem using the synthetic dataset with image size 1000×1000 pixels. Among the projections of the 1000 synthetic generated scene points, the inliers are corrupted by uniformly distributed noise within a 7×7 window, while the outliers are corrupted by shifting the point by at least half the image size. 80 iterations is used for every case running 1000 times, and a successful RANSAC estimation is decided if $\xi_{\mathbf{R}[\text{deg}]} < 2$ degrees. The table shows only $\mathbf{R}_2$, $\mathbf{t}_2$, and $\mathbf{s}_2$ since they share similar accuracy with $\mathbf{R}_3$, $\mathbf{t}_3$, and $\mathbf{s}_3$. The errors are the averaged accuracy of successful cases which show that below 70% outlier ratio the success rate has more than 80%, and as long as inlier points are picked, the estimation accuracy is very promising.

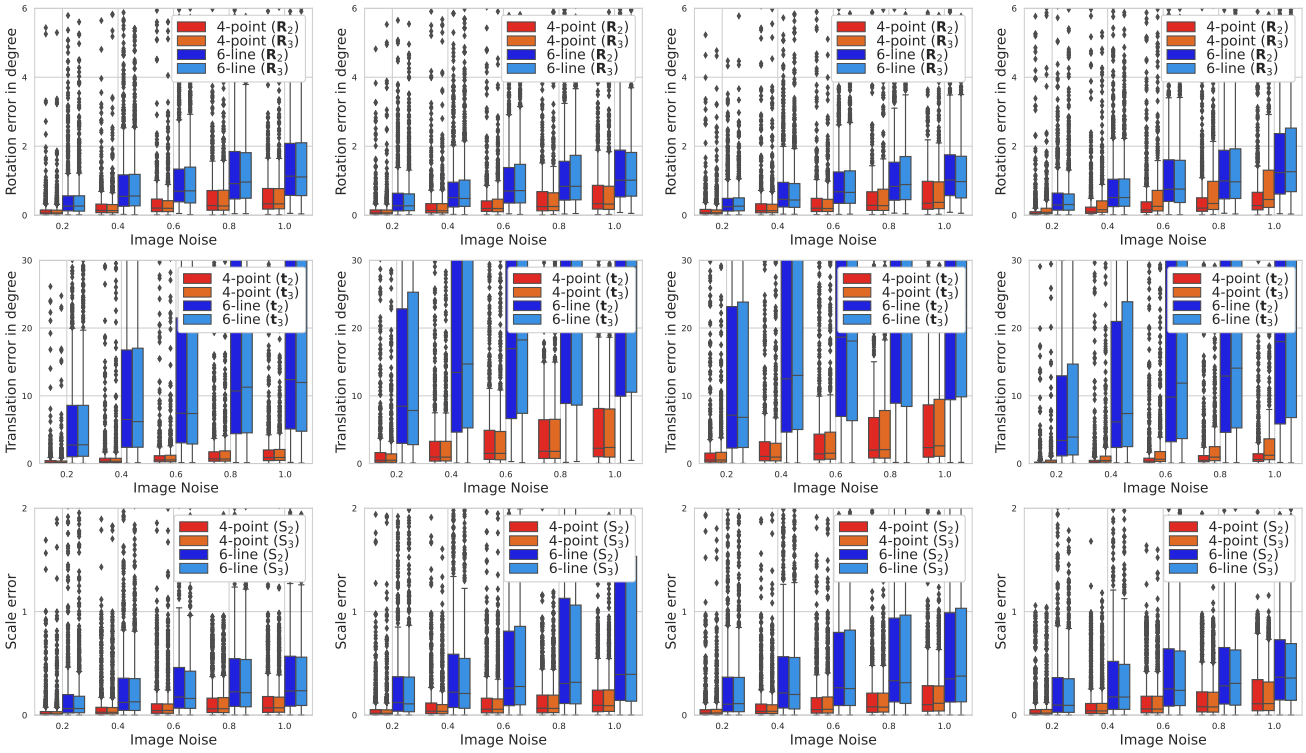
Outlier Ratio	Success Rate	$\xi_{\mathbf{R}[\text{deg}]}$	$\xi_{\mathbf{t}[\text{deg}]}$	ξ_{Scale}
10%	100%	0.193	0.018	0.009
20%	100%	0.199	0.133	0.076
30%	100%	0.574	0.188	0.175
40%	96.4%	0.997	0.159	0.122
50%	90.7%	1.344	0.442	0.169
60%	87.1%	1.128	0.134	0.137
70%	81.4%	0.895	0.143	0.060
80%	66.3%	1.495	0.328	0.329
90%	42.7%	1.749	0.134	0.139

Table 2. Performance of the 4-point problem under a RANSAC scheme given different outlier ratios using the synthetic dataset.

5.3. Real-world Experiments

In order to test the proposed technique on real-world data, we chose the `LaMAR` [22][1] dataset, which contains both indoor and outdoor images with illumination and semantic changes as well as dynamic objects. We used the image sequences captured by a multi-camera rig (HoloLens 2).

[1] https://lamar.ethz.ch/



General motion | Sideways motion | Forward motion | Semi-pure translation

Figure 6. Boxplot of the relative errors under increasing image noise with a maximum value of 1 pixel. **Top Row:** Rotation error in degree. **Middle Row:** Translation error in degree. **Bottom Row:** Translation scale error. From left column to right column are general motion, sideways motion, forward motion and semi-pure translation, respectively.

Solver	Success Rate (%)	# of real sols. (# of positive real sols.)				GPU-HC Timings (ms)				Mean Julia Timings (ms) (8-core CPU)	Julia / GPU-HC
		min	max	median	mean	min	max	median	mean		
4-point (DP)	99.78	34(4)	93(12)	68(8)	68.5(8.1)	9.6	32.8	16.8	17.4	428.26	24.6×
4-point (SP)	99.06	32(2)	89(10)	52(5)	53.2(5.1)	5.3	15.7	7.2	7.3		58.7×
6-line (DP)	98.67	52	139	87	87.7	2.9	72.4	35.4	34.1	1103	32.3×
6-line (SP)	97.63	48	120	84	82.4	5.2	15.4	7.1	7.2		153.2×

Table 3. Success rate of finding the true solutions and the timings of solving the problems.

Figure 7. An example of four images from the LaMAR CAB dataset captured by a multi-camera rig with four cameras.

It has a specialized large field-of-view (FOV) multi-camera tracking rig with four cameras. In general, this dataset contains three large scale sequences including: (i) HGE, the ground floor of a historic university building consists of numerous large halls and broad esplanades on either side with 18000 m^2. (ii) CAB, a multi-story office building that features a variety of small and large offices, a kitchen, storage rooms, and two courtyards with 12000 m^2. (iii) LIN, several blocks of an ancient town, characterized by shops, restaurants, and narrow passageways with 15000 m^2. To setup the experiments we extracted triplets of multi-camera images from the mapping set, where the ground-truth poses are available. From each trajectory in the mapping set we select consecutive triplets satisfying the following criteria; the distance between consecutive rigs is in $[0.2m,\ 5m]$ and and at least two image pairs have more than 50 matches. This yielded in total 1280 (CAB), 1537 (HGE) and 2213 (LIN) triplets for evaluation. Pairwise 2d-2d matches are established using the SIFT[16] implementation in [23].

Since there are no related solvers for generalized three-view relative pose problem, we only compare the proposed 4-point solver with the two-view 6-point solver [25].

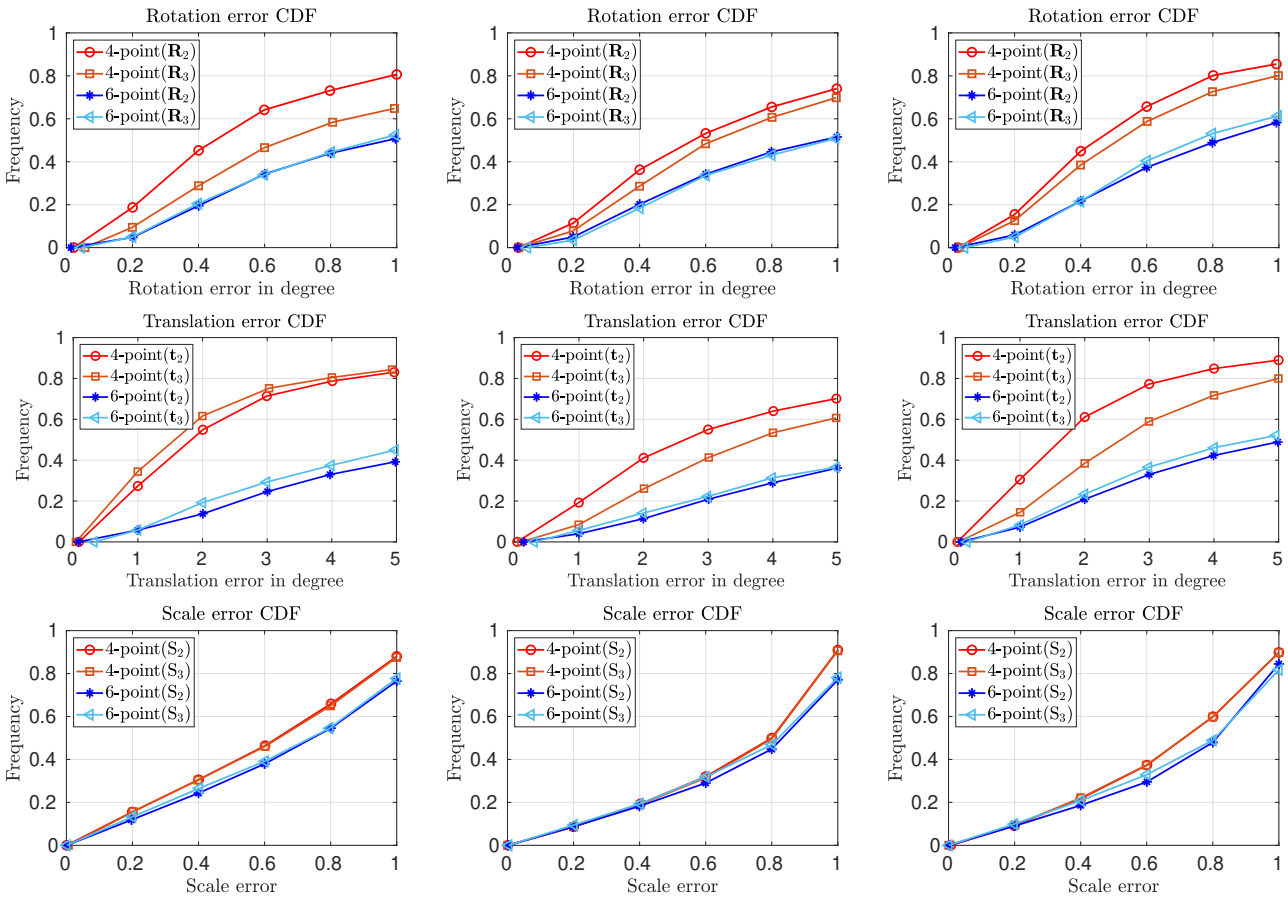


Figure 8. The cumulative distribution functions of the rotation, translation and scale errors. Being accurate is interpreted as a curve close to the top-left corner.

LaMAR	Solver	$\xi_{\mathbf{R}}$	$\xi_{\mathbf{t}}$	ξ_{S}	Inlier	Iter
CAB	4-point	0.54	1.65	0.63	329	105
	6-point	0.94	6.51	0.73	317	189
HGE	4-point	0.58	3.12	0.80	392	99
	6-point	0.97	8.25	0.85	387	193
LIN	4-point	0.46	2.01	0.72	573	100
	6-point	0.78	4.86	0.81	535	199

Table 4. The median errors, the number of inliers and the number of RANSAC iterations on the LaMAR dataset.

To make a fair comparison, we use the solvers in a simple RANSAC framework with a maximum number of RANSAC iterations set to 500. The inlier-outlier threshold is set to 2 pixels. In Fig. 8, we show the cumulative distribution functions (CDF) of the rotation (in degrees), translation (in degrees) and scale errors on the three tested datasets. Being accurate is interpreted as a curve close to the top-left corner. The median errors are reported in Table 4. The proposed solver leads to more accurate results, more inliers and fewer iterations than the compared two-view method. Due to the lack of space we only show the results using intra correspondences. Results for inter correspondences and mean errors are shown in the supplementary material.

Limitations. The main limitation is that the refinement step is currently done in CPU. We will consider to include the refinement in the GPU computation in the future.

6. Conclusion

We discuss minimal cases of generalized three-view relative pose estimation. In particular, we propose GPH-HC based solvers to the 4-point and 6-line cases. Experiments on thousands of image pairs from publicly available datasets show that the proposed three-view 4-point solver is superior to the state-of-the-art two-view 6-point solver. One goal of future research is to use the technique proposed in [11] to improve the efficiency. We believe that the proposed method is promising and will be a good complement to existing methods to improve the accuracy and stability of the structure from motion systems.

Acknowledgements. VL, KÅ received funding from ELLIIT. Kimia and Chien gratefully acknowledge the support of NSF award 1910530.

References

[1] Paul Breiding and Sascha Timme. HomotopyContinuation.jl: A Package for Homotopy Continuation in Julia. In *International Congress on Mathematical Software*, pages 458–465. Springer, 2018.

[2] Chiang-Heng Chien, Hongyi Fan, Ahmad Abdelfattah, Elias Tsigaridas, Stanimire Tomov, and Benjamin Kimia. GPU-based homotopy continuation for minimal problems in computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15765–15776, 2022.

[3] Chiang-Heng Chien, Hongyi Fan, Ahmad Abdelfattah, Elias Tsigaridas, Stanimire Tomov, and Benjamin Kimia. Parallel path tracking for homotopy continuation using GPU. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, 2022.

[4] David A. Cox, John Little, and Donal O'shea. *Using algebraic geometry*. Springer Science & Business Media, 2006.

[5] Timothy Duff, Kathlen Kohn, Anton Leykin, and Tomas Pajdla. PLMP - Point-Line Minimal Problems in Complete Multi-View Visibility. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[6] Ricardo Fabbri, Timothy Duff, Hongyi Fan, Margaret H. Regan, David da C. de Pinho, Elias Tsigaridas, Charles W. Wampler, Jonathan D. Hauenstein, Peter J. Giblin, Benjamin Kimia, Anton Leykin, and Tomas Pajdla. Trifocal Relative Pose From Lines at Points. *Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 2022.

[7] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981.

[8] Jonathan D Hauenstein and Margaret H Regan. Adaptive strategies for solving parameterized systems using homotopy continuation. *Applied Mathematics and Computation*, 332:19–34, 2018.

[9] Robert J. Holt and Arun N. Netravali. Motion and structure from line correspondences: Some further results. *International Journal of Imaging Systems and Technology*, 1994.

[10] Hongdong Li, Richard Hartley, and Jae-hak Kim. A linear approach to motion estimation using generalized camera models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[11] Petr Hruby, Timothy Duff, Anton Leykin, and Tomas Pajdla. Learning to Solve Hard Minimal Problems. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[12] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Automatic generator of minimal problem solvers. In *European Conference on Computer Vision (ECCV)*, 2008.

[13] Viktor Larsson, Kalle Åström, and Magnus Oskarsson. Efficient solvers for minimal problems by syzygy-based reduction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] Viktor Larsson, Magnus Oskarsson, Kalle Åström, Alge Wallis, Zuzana Kukelova, and Tomas Pajdla. Beyond gröbner bases: Basis selection for minimal solvers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[15] Gim Hee Lee, Marc Pollefeys, and Friedrich Fraundorfer. Relative Pose Estimation for a Multi-camera System with Known Vertical Direction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[16] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 2004.

[17] Branislav Micusik and Tomas Pajdla. Autocalibration & 3d reconstruction with non-central catadioptric cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[18] Alexander Morgan. *Solving polynomial systems using continuation for engineering and scientific problems*. SIAM, 2009.

[19] David Nistér and Frederik Schaffalitzky. Four Points in Two or Three Calibrated Views: Theory and Practice. *International Journal of Computer Vision*, 2006.

[20] Robert Pless. Using many cameras as one. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[21] Long Quan, Bill Triggs, and Bernard Mourrain. Some Results on Minimal Euclidean Reconstruction from Four Points. *Journal of Mathematical Imaging and Vision*, 2006.

[22] Paul-Edouard Sarlin, Mihai Dusmanu, Johannes L. Schönberger, Pablo Speciale, Lukas Gruber, Viktor Larsson, Ondrej Miksik, and Marc Pollefeys. LaMAR: Benchmarking Localization and Mapping for Augmented Reality. In *European Conference on Computer Vision (ECCV)*, 2022.

[23] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[24] Andrew J Sommese and Charles W Wampler. *The Numerical solution of systems of polynomials arising in engineering and science*. World Scientific, 2005.

[25] Henrik Stewenius, David Nistér, Magnus Oskarsson, and Kalle Åström. Solutions to minimal generalized relative pose problems. In *Workshop on omnidirectional vision*, 2005.

[26] Peter Sturm. Multi-View Geometry for General Camera Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.