A Direct Least-Squares Solution to Multi-View Absolute and Relative Pose from 2D-3D Perspective Line Pairs*

Hichem Abdellali[a], Robert Frohlich[a]
[a]University of Szeged, Szeged, Hungary
hichem,frohlich@inf.u-szeged.hu

Zoltan Kato[a,b]
[b]J. Selye University, Komarno, Slovakia
kato@inf.u-szeged.hu

Abstract

We propose a new algorithm for estimating the absolute and relative pose of a multi-view camera system. We derive a direct least squares solver using Grobner basis which works both for the minimal case (set of 3 line pairs for each camera) and the general case using all inlier 2D-3D line pairs for a multi-view camera system. The algorithm has been validated on a large synthetic dataset as well as real data. Experimental results confirm the stable and real-time performance under realistic outlier ratio and noise on the line parameters. Comparative tests show that our method compares favorably to the latest state of the art algorithms.

1. Introduction

Absolute pose estimation consists in determining the position and orientation of a camera with respect to a 3D world coordinate frame, while relative pose estimation aims to compute the same parameters with respect to a reference camera. Relative pose estimation is needed when we have a system of two or more cameras. Absolute and relative poses are fundamental in various computer vision applications, such as visual odometry, simultaneous localization and mapping (SLAM), image-based localization and navigation, augmented reality. The problem has been extensively studied yielding various formulations and solutions. Most of the approaches focus on a single camera pose estimation using point correspondences. However, modern applications, especially in vision-based localization and navigation for robotics and autonomous vehicles, it is often desirable to use multi-camera systems which covers large field of views and provides direct 3D measurements. The absolute pose estimation of a perspective camera from n 2D–3D point correspondences is known in the literature as the *Perspective-n-Point* (PnP) problem, which has been widely studied in the last few decades [7, 19, 20, 12, 10]. Various solutions have been developed for both large n as well as for the $n = 3$ minimal case (see [12] for a recent overview). The use of line correspondences, known as the *Perspective-n-Line* (PnL) problem, has also been investigated in the last two decades, yielding robust and efficient solutions (see [36] for a detailed overview). The minimal case of $n = 3$ line correspondences is particularly important as its solution is the basis for dealing with the general PnL problem. It has been shown in [5], that P3L leads to an 8^{th} order polynomial, which is higher than the 4^{th} order polynomial of a P3P problem. While the use of point and line correspondences are widespread, there are pose estimation methods relying on other type of correspondences, *e.g.* set of regions [28, 27] or silhouettes. However, such approaches are typically computationally more expensive hence they cannot be used as real-time solvers.

Recently, due to increasing popularity of multi-camera systems in *e.g.* autonomous driving [17] and UAVs, the problem of multi-view pose estimation has been addressed. Solutions to the PnP or PnL problem cover only single-view perspective cameras, hence new methods are needed to efficiently deal with the generalized PnP (gPnP) or non-perspective PnP (NPnP) [4, 12, 17, 18]. As for relative pose, lines in two images do not provide any constraints on the camera pose [8]. However, if information is available about some special 3D geometric configuration of the lines, then relative pose can also be estimated [6]. In [33], relative pose estimation is extended with a 3D line direction estimation step applied for monocular visual odometry, while [21] used a line-based space resection approach for UAV navigation. More recently, in [38], lines of building structures

*This work was partially supported by the NKFI-6 fund through project K120366; "Integrated program for training new generation of scientists in the fields of computer science", EFOP-3.6.3-VEKOP-16-2017-0002; the Research & Development Operational Programme for the project "Modernization and Improvement of Technical Infrastructure for Research and Development of J. Selye University in the Fields of Nanotechnology and Intelligent Space", ITMS 26210120042, co-funded by the European Regional Development Fund; Research & Innovation Operational Programme for the Project: "Support of research and development activities of J. Selye University in the field of Digital Slovakia and creative industry", ITMS code: NFP313010T504, co-funded by the European Regional Development Fund.

were used for trajectory estimation in a SLAM approach, while in [26], a hybrid point and line based SLAM technique was proposed.

Several point-based [24, 10, 12] as well as mixed point and line based methods [4, 22] exist, but little work has been done on using line correspondences only. For absolute pose of a single camera, Mirzaei and Roumeliotis proposed the first globally optimal non-iterative solution (AlgLS) [23] which formulates the problem as a multivariate polynomial system with rotation parametrized by the Cayley-Gibbs-Rodriguez (CGR) representation. Zhang *et al*. proposed RPnL [37] which was further modified into the Accurate Subset-based PnL (ASPnL) method [36], which is one of the most accurate non-iterative method. Another recent work from Wang *et al*. deals with the P3L [34] as well as with the PnL [35] problem. In [35], a fast and robust solution is proposed (called SRPnL) and its superior performance is confirmed by a comprehensive experimental comparison with many state of the art PnL solvers, like AlgLS [23], ASPnL [36]. Therefore in this paper, we will validate our method through various comparative experiments with AlgLS and SRPnL as they perfomed the best in [35]. For multi-view camera systems, one notable work is the minimal NP3L solver of Lee [16], which deals with the 6 pose parameter estimation for a fully calibrated multiview camera system. In [9], the same problem is addressed with known vertical direction which leads to two fast and robust solvers.

While robust minimal solutions based on line correspondences for absolute pose [23, 37, 36, 16, 9, 35] or absolute and relative pose with known vertical direction [1] exists, none of these methods estimate full absolute and relative poses simultaneously in a multiview system. In this paper, we propose a direct least squares solution to this problem. First, a direct solver using Grobner bases is proposed which works for any $n \geq 3$ number of lines suitable for hypothesis testing like in RANSAC [7]. The poses can be further refined through a few iterations of an iterative least squares solver, which runs efficiently due to the optimal initialization provided by our direct solver. The performance and robustness of the proposed method have been evaluated for $n \geq 3$ lines as well as for $M \geq 1$ camera systems on large synthetic datasets as well as on real data.

2. Perspective Lines and Camera Pose

Given a calibrated perspective camera, its camera matrix $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$ consists of the internal calibration matrix $\mathbf{K}$ and the camera pose $[\mathbf{R}|\mathbf{t}]$ w.r.t. the world coordinate frame $\mathcal{W}$. A homogeneous 3D point $\mathbf{X}$ is mapped by $\mathbf{P}$ into a homogeneous 2D image point $\mathbf{x}'$ as [8]

$$\mathbf{x}' \cong \mathbf{P}\mathbf{X} = \mathbf{K}[\mathbf{R}|\mathbf{t}]\mathbf{X}, \tag{1}$$

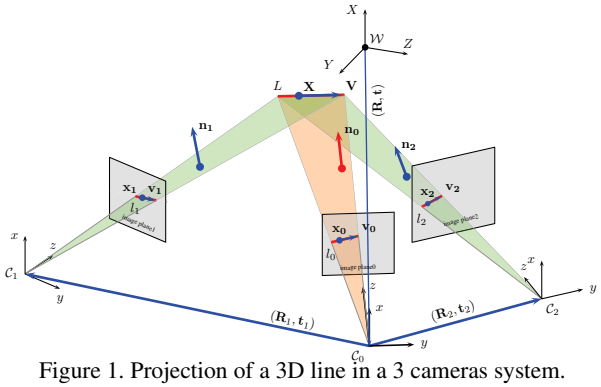

Figure 1. Projection of a 3D line in a 3 cameras system.

where '$\cong$' denotes the equivalence of homogeneous coordinates, *i.e*. equality up to a non-zero scale factor. Since we assume a calibrated camera, we can multiply both sides of (1) by $\mathbf{K}^{-1}$ and work with the equivalent normalized image

$$\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}' \cong \mathbf{K}^{-1}\mathbf{P}\mathbf{X} = [\mathbf{R}|\mathbf{t}]\mathbf{X}. \tag{2}$$

The above equation is the starting point of perspective pose estimation [12, 20, 19, 14]: given a set of 3D–2D point correspondences $(\mathbf{x}_i \leftrightarrow \mathbf{X}_i)$, one can recover the 3D rigid body transformation $(\mathbf{R}, \mathbf{t}) : \mathcal{W} \rightarrow \mathcal{C}$ acting between the world coordinate frame $\mathcal{W}$ and the camera coordinate frame $\mathcal{C}$.

3D lines may have various representations in the projective space [25, 3]. In this paper, 3D lines will be represented as $L = (\mathbf{V}, \mathbf{X})$, where $\mathbf{V}$ is the unit direction vector of the line and $\mathbf{X}$ is a point on the line [29, 9, 1].

The projection of L in a perspective camera is a 2D line l which can also be represented as $l = (\mathbf{v}, \mathbf{x})$. Note that the point $\mathbf{x}$ is *not* necessarily the image of the 3D point $\mathbf{X}$! Both the 3D line L and its perspective image l lie on the projection plane π passing through the camera projection center $\mathbf{C}$ (see Fig. 1). The unit normal to the plane π in the camera coordinate system $\mathcal{C}$ is denoted by $\mathbf{n}$, which can be computed from the image line l as

$$\mathbf{n} = \frac{(\mathbf{v} \times \mathbf{x})}{\|\mathbf{v} \times \mathbf{x}\|}. \tag{3}$$

Since L lies also on π, its direction vector $\mathbf{V}$ is perpendicular to $\mathbf{n}$. Hence we get the following equation which involves only the absolute pose $(\mathbf{R}, \mathbf{t})$ [9, 1]

$$\mathbf{n}^\top \mathbf{R}\mathbf{V} = \mathbf{n}^\top \mathbf{V}^{\mathcal{C}} = 0, \tag{4}$$

where $\mathbf{R}$ is the rotation matrix from the world $\mathcal{W}$ to the camera $\mathcal{C}$ frame and $\mathbf{V}^{\mathcal{C}}$ denotes the unit direction vector of L in the camera coordinate frame. Furthermore, the vector from the camera center $\mathbf{C}$ to the point $\mathbf{X}$ on line L is also lying on π, thus it is also perpendicular to $\mathbf{n}$:

$$\mathbf{n}^\top(\mathbf{R}\mathbf{X} + \mathbf{t}) = \mathbf{n}^\top \mathbf{X}^{\mathcal{C}} = 0, \tag{5}$$

where $\mathbf{t}$ is the translation from the world $\mathcal{W}$ to the reference camera $\mathcal{C}$ frame and $\mathbf{X}^{\mathcal{C}}$ denotes the point $\mathbf{X}$ on L in the camera coordinate frame.

The absolute pose $(\mathbf{R}, \mathbf{t})$ has 6 degrees of freedom (3 rotation angles (α, β, γ) and 3 translation components (t_x, t_y, t_z) along the 3 coordinate axes). Thus to solve for the pose using (4) and (5), we need a minimum of 3 line correspondences $\{(L_i, l_i)\}_{i=1}^{3}$, which is called the P3L problem. The solution is obtained in two steps: first the rotation $\mathbf{R}$ is solved using (4), which in general involves solving a system of 8-th order polynomials. Then translation is obtained from (5) by backsubstituting $\mathbf{R}$, which yields a linear system of equations in terms of $\mathbf{t}$ [36, 16, 37, 34]. Clearly, the main challange is the solution for $\mathbf{R}$ due to the nonlinearity of the equations as well as the additional constraints on $\mathbf{R}$ to be a valid rotation (*i.e*. orthonormal) matrix. Although for special line configurations (*e.g*. orthogonal, parallel or intersecting lines) [36] or with additional knowledge of *e.g*. the vertical direction [9, 1], a lower order polynomial may be achieved, usually the P3L polynomial will not be lower than 8 for general line configurations [5].

Let us have a closer look at the parametrization of rotations as the final form of (4) depends on this. It is well known, that the rotation group $SO(3)$ has 3 degrees of freedom. The most popular parametrization is *Euler angles*, which represents $\mathbf{R}$ via 3 rotation angles around the coordinate axes. Unfortunately, this representation involves trigonometric functions which would yield trigonometric equations in (4). One approach is to letting these trigonometric functions of one angle α to be two separate unknowns [20, 36, 37, 9], which –together with the trigonometric constraints– lead to polynomial equations. Alternatively, one can solve directly for the 9 elements of $\mathbf{R}$ in (4) –as a linear system– and then enforce orthonormality on the solution yielding again to (different) polynomial equations [35, 36]. Herein, in order to eliminate trigonometric functions, let us substitute $s = \tan(\alpha/2)$ [14, 2, 9, 1], for which $\cos(\alpha) = (1-s^2)/(1+s^2)$ and $\sin(\alpha) = 2s/(1+s^2)$, yielding a second order polynomial in s for one rotation around a coordinate axis.

3. Direct Least Squares Solver

In order to reduce the number of unknowns to 2 in (4), we eliminate one rotation in $\mathbf{R}$ by defining an intermediate coordinate system $\mathcal{M}$ [37, 36, 35] in which the rotation angle around the X axis can be easily obtained. First, let us select a line pair (L_0, l_0) with the longest projection length as longer edges are less affected by noise on their endpoints [37, 36, 35]. The origin of $\mathcal{M}$ is located at the

Algorithm 1 The proposed MRPnL algorithm.

Input: M: the number of the cameras ≥ 1 and the reference camera $\mathcal{C}_0$
$N_{\mathcal{C}_i}$ 2D-3D lines pairs ($\mathbf{n}_j^{\mathcal{C}_i}$, $\mathbf{X}_j$, and $\mathbf{V}_j$) for each camera $\mathcal{C}_i$

Output: The absolute pose $(\mathbf{R}, \mathbf{t}) : \mathcal{W} \rightarrow \mathcal{C}_0$ and the relative poses $(\mathbf{R}_i, \mathbf{t}_i) : \mathcal{C}_0 \rightarrow \mathcal{C}_i$

1: Normalize the 3D line points $\mathbf{X}_j$ by $\mathbf{N}$.
2: Rotation: Determine the intermediate rotation $\mathbf{R}_{\mathcal{M}}$ and apply it to the input line pairs. Then $\mathbf{R}_x^{\mathcal{M}}$ is calculated and the two remaining rotation $\mathbf{R}^{\mathcal{M}}$ is obtained by solving the polynomial system of equations (13), which together provides $\mathbf{R}$.
3: Translation: By back-substituting the rotation, solve the linear system of equations (5) for $\mathcal{C}_0$ or (17) for the other cameras via SVD.
4: Optional refinement of the absolute and relative poses for all cameras and lines simultaneously by solving the system (19) in the lest squares sense (see Section 3.3).
5: Denormalize to get the final pose estimates.

origin of $\mathcal{W}$ and its axes $(\mathbf{X}_{\mathcal{M}}, \mathbf{Y}_{\mathcal{M}}, \mathbf{Z}_{\mathcal{M}})$ are

$$\mathbf{Y}_{\mathcal{M}} = \frac{\mathbf{n}_0^{\mathcal{C}}}{\|\mathbf{n}_0^{\mathcal{C}}\|} \tag{6}$$

$$\mathbf{X}_{\mathcal{M}} = \frac{\mathbf{n}_0^{\mathcal{C}} \times \mathbf{V}_0^{\mathcal{W}}}{\|\mathbf{n}_0^{\mathcal{C}} \times \mathbf{V}_0^{\mathcal{W}}\|} \tag{7}$$

$$\mathbf{Z}_{\mathcal{M}} = \frac{\mathbf{X}_{\mathcal{M}} \times \mathbf{Y}_{\mathcal{M}}}{\|\mathbf{X}_{\mathcal{M}} \times \mathbf{Y}_{\mathcal{M}}\|} \tag{8}$$

where the Y axis of $\mathcal{M}$ aligns with $\mathbf{n}_0^{\mathcal{C}}$. The rotation $\mathbf{R}_{\mathcal{M}} = [\mathbf{X}_{\mathcal{M}}, \mathbf{Y}_{\mathcal{M}}, \mathbf{Z}_{\mathcal{M}}]^\top$ rotates the normals and direction vectors into the intermediate frame $\mathcal{M}$. The rotation $\mathbf{R}_x^{\mathcal{M}}$ around X axis within $\mathcal{M}$ is then easily calculated because it is the angle between the Z axis and $\mathbf{V}_0^{\mathcal{M}}$, hence the rotation matrix acting within the intermediate coordinate frame $\mathcal{M}$ is composed of the rotations around the remaining two axes as

$$(1+s^2)(1+r^2)\mathbf{R}^{\mathcal{M}} = \mathbf{R}_y^{\mathcal{M}}(s)\mathbf{R}_z^{\mathcal{M}}(r) = \begin{bmatrix} (1-s^2)(1-r^2) & -2r(1-s^2) & 2s(r^2+1) \\ 2r(s^2+1) & (s^2+1)(1-r^2) & 0 \\ -2s(1-r^2) & 4sr & (1-s^2)(r^2+1) \end{bmatrix} \tag{9}$$

and we obtain the new form of (4) as

$$(\mathbf{R}_{\mathcal{M}}\mathbf{n})^\top \mathbf{R}^{\mathcal{M}}(\mathbf{R}_x^{\mathcal{M}}\mathbf{R}_{\mathcal{M}}\mathbf{V}) = \mathbf{n}^{\mathcal{M}\top}\mathbf{R}^{\mathcal{M}}\mathbf{V}^{\mathcal{M}} = 0. \tag{10}$$

Expanding the above equation gives a 4-th order polynomial of (s, r) with coefficients in terms of $\mathbf{V}^{\mathcal{M}}$ and $\mathbf{n}^{\mathcal{M}}$:

$$\mathbf{a}^\top \mathbf{u} = \begin{bmatrix} V_1\,n_1 + V_2\,n_2 + V_3\,n_3 \\ 2\,V_1\,n_2 - 2\,V_2\,n_1 \\ 2\,V_3\,n_1 - 2\,V_1\,n_3 \\ -V_1\,n_1 - V_2\,n_2 + V_3\,n_3 \\ -V_1\,n_1 + V_2\,n_2 - V_3\,n_3 \\ V_1\,n_1 - V_2\,n_2 - V_3\,n_3 \\ 2\,V_1\,n_2 + 2\,V_2\,n_1 \\ 2\,V_3\,n_1 + 2\,V_1\,n_3 \\ 4\,V_2\,n_3 \end{bmatrix}^\top \cdot \begin{bmatrix} 1 \\ r \\ s \\ r^2 \\ s^2 \\ s^2r^2 \\ s^2r \\ sr^2 \\ sr \end{bmatrix} = 0 \quad (11)$$

Each line pair generates one such equation, yielding a system of N equations, which is solved in the least squares sense. For this purpose, let's take the sum of squares of the above system

$$e(s, r) = \sum_{i=1}^{N} (\mathbf{a}_i^\top \mathbf{u})^2 \quad (12)$$

and then find $\arg\min_{(s,r)} e(s, r)$. The first order optimality condition for (12) is

$$\nabla e(s, r) = \begin{bmatrix} \frac{\partial e(s,r)}{\partial s} \\ \frac{\partial e(s,r)}{\partial r} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} \mathbf{b_s}_i^\top \mathbf{u_s} \\ \sum_{i=1}^{N} \mathbf{b_r}_i^\top \mathbf{u_r} \end{bmatrix} = \mathbf{0} \quad (13)$$

where for each line pair

$$\mathbf{b_r}^\top \mathbf{u_r} = \begin{bmatrix} 2\,a_1a_4 \\ 6\,a_2a_4 \\ 2\,a_1a_7 + 2\,a_3a_4 + 2\,a_5a_9 \\ 4\,a_1a_2 + 2\,{a_4}^2 \\ 2\,a_1a_9 + 2\,a_4a_5 \\ 6\,a_2a_7 + 6\,a_4a_6 + 6\,a_8a_9 \\ 4\,a_1a_6 + 4\,a_2a_3 + 4\,a_4a_7 + 4\,a_5a_8 + 2\,{a_9}^2 \\ 6\,a_2a_9 + 6\,a_4a_8 \\ 4\,{a_2}^2 \\ 2\,a_3a_7 \\ 2\,a_3a_9 + 2\,a_5a_7 \\ 4\,a_1a_8 + 4\,a_2a_5 + 4\,a_4a_9 \\ 4\,{a_6}^2 \\ 8\,a_6a_8 \\ 8\,a_2a_6 + 4\,{a_8}^2 \\ 8\,a_2a_8 \\ 6\,a_6a_7 \\ 6\,a_6a_9 + 6\,a_7a_8 \\ 4\,a_3a_6 + 2\,{a_7}^2 \\ 4\,a_3a_8 + 4\,a_5a_6 + 4\,a_7a_9 \end{bmatrix}^\top \begin{bmatrix} 1 \\ r^2 \\ s^2 \\ r \\ s \\ s^2r^2 \\ s^2r \\ sr^2 \\ r^3 \\ s^4 \\ s^3 \\ sr \\ r^3s^4 \\ r^3s^3 \\ r^3s^2 \\ r^3s \\ r^2s^4 \\ r^2s^3 \\ rs^4 \\ rs^3 \end{bmatrix} \quad (14)$$

and similarly $\mathbf{b_s}^\top \mathbf{u_s}$ can also be expressed in terms of the coefficients $\mathbf{a}$ of each line pair. Thus the solution of the system of 2 polynomial equations (each of them is 7-th order) in (13) provides the rotation parameters (s, r). Herein, we use the automatic generator of Kukelova *et al.* [13] to generate a solver using Grobner basis[13, 15] for the system in (13). Once the solution(s) are obtained, the complete $\mathbf{R}$, acting between the world $\mathcal{W}$ and camera $\mathcal{C}$ frame, is obtained as $\mathbf{R} = \mathbf{R}_{\mathcal{M}}^\top (\mathbf{R}^{\mathcal{M}} \mathbf{R}_x^{\mathcal{M}}) \mathbf{R}_{\mathcal{M}}$.

The translation $\mathbf{t}$ is then obtained by backsubstituting $\mathbf{R}$ into (5) yielding a system of linear equations, which can be solved by SVD decomposition. While in (4), all quantities are already normalized ($\mathbf{n}$ and $\mathbf{V}$ are of unit length), (5) contains the 3D point $\mathbf{X}$ given in an arbitrary world coordinate system $\mathcal{W}$, which needs to be normalized for numerical stability [8]. Therefore, we transform our 3D line segments into a unit cube around the origin of $\mathcal{W}$ by a normalizing transformation $\mathbf{N}$ composed of a translation $-[u, v, z]^T$ with $[u, v, z]^T$ being the centroid of the 3D scene points; followed by a uniform scaling $s = \frac{1}{\max(|h|,|w|,|d|)}$, where h is the height, w is the width, and d is the depth of the 3D data. The solution is then obtained in this normalized space, hence the result $(\tilde{\mathbf{R}}, \tilde{\mathbf{t}})$ need to be denormalized. Since the equations used to solve for the rotation are unaffected by this normalization (thanks to uniform scaling!), $\tilde{\mathbf{R}}$ is the final rotation, while the translation $\tilde{\mathbf{t}}$ needs to be corrected by applying $\begin{bmatrix} \mathbf{I} & \tilde{\mathbf{t}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{N}$.

Although (13) might have several solutions, the solver will only return the real ones and then one has to select the geometrically valid $(\mathbf{R}, \mathbf{t})$ based on the visibility of the lines and the backprojection error (18) [36, 16, 35, 9, 1].

3.1. Multi-view Case and Relative Pose

When the 3D lines are viewed by a system of M calibrated perspective cameras, each 3D line L has up to M images $l_i, i = 1 \dots M$. One of the cameras is the reference camera $\mathcal{C}_0$ therefore the absolute pose of the camera system $(\mathbf{R}, \mathbf{t})$ is defined as the rigid transformation acting between $\mathcal{W}$ and $\mathcal{C}_0$, while individual camera frames $\mathcal{C}_i$ are related to the reference camera frame via the relative poses $(\mathbf{R}_i, \mathbf{t}_i) : \mathcal{C}_0 \rightarrow \mathcal{C}_i, i = 1, \dots, M-1$. Of course, the equations (4), (16) and (5) remain valid for $\mathcal{C}_0$, while for the other cameras $\mathcal{C}_i$, the projection of L yields similar equations but the unknown relative pose $(\mathbf{R}_i, \mathbf{t}_i)$ will also be involved. Hence (4) becomes:

$$\mathbf{n}_i^\top \mathbf{R}_i \mathbf{V}^{\mathcal{C}_0} = \mathbf{n}_i^\top \mathbf{R}_i \mathbf{R} \mathbf{V} = 0 \quad (15)$$

from which we get

$$(\mathbf{R}_{\mathcal{M}_i} \mathbf{n}_i)^\top \mathbf{R}^{\mathcal{M}_i} (\mathbf{R}_x^{\mathcal{M}_i} \mathbf{R}_{\mathcal{M}_i} \mathbf{R} \mathbf{V}) = {\mathbf{n}^{\mathcal{M}_i}}^\top \mathbf{R}^{\mathcal{M}_i} \mathbf{V}^{\mathcal{M}_i} = 0 \quad (16)$$

which –after a similar derivation as in the single camera case– yields also a system of polynomial equations of the same form as in (13), hence the same solver can be used

to solve for each camera $\mathcal{C}_i, i = 1, \ldots, M-1$. Once the solutions are obtained, each $\mathbf{R}_i$ is backsubstituted into the corresponding linear system similar to (5):

$$\mathbf{n}_i^\top(\mathbf{R}_i\mathbf{X}^{\mathcal{C}_0} + \mathbf{t}_i) = \mathbf{n}_i^\top(\mathbf{R}_i(\mathbf{R}\mathbf{X} + \mathbf{t}) + \mathbf{t}_i) = 0 \quad (17)$$

which is solved for $\mathbf{t}_i$ by SVD.

3.2. Robust Outlier Filtering

In real applications, putative line correspondences are extracted that are corrupted by outliers as well as noise. While noise is handled well by our least squares formulation of the equations, outliers must be removed via RANSAC [7] or the M-estimator sample consensus (MSAC) algorithm [30], which relies on a minimal solver and a backprojection error. The minimal set consists of 3 line-pairs, providing 2 equations for the rotation in (4) or (16), and 3 equations for the translation in (5) or (17). The equations are then solved as outlined before and outliers are filtered using the backprojection error of the 3D lines. Herein we used the error measure proposed in [16] that practically calculates the mean of the shortest distances $d_{\mathbf{x}_s}$ and $d_{\mathbf{x}_e}$ from the 2D line segment endpoints $\mathbf{x}_s$ and $\mathbf{x}_e$ to the corresponding infinite line determined by the back-projected 3D line onto the normalized plane:

$$\frac{d_{\mathbf{x}_s} + d_{\mathbf{x}_e}}{2(\|\mathbf{x}_e - \mathbf{x}_s\|)} \quad (18)$$

Note, that the error is normalized with the length of the 2D line segment, hence making the measure independent of the length of the detected 2D line segment.

3.3. Pose Refinement

We will now formulate a least-squares refinement for the multi-view case, based on the equations (4) and (15) by simply stacking for each line pair in $\mathcal{C}_0$ (4), (5) and for each camera $i = 1, \ldots, M-1$ and each line pair in $\mathcal{C}_i$ (15) and (17) containing the absolute pose $(\mathbf{R}, \mathbf{t})$ and the relative poses $(\mathbf{R}_i, \mathbf{t}_i)$:

$$\left.\begin{aligned} &\forall j = 1, \ldots, N_{\mathcal{C}_0}: \\ &\qquad {\mathbf{n}_j^{\mathcal{C}_0}}^\top \mathbf{R}\mathbf{V}_j = 0 \\ &\qquad {\mathbf{n}_j^{\mathcal{C}_0}}^\top (\mathbf{R}\mathbf{X}_j + \mathbf{t}) = 0 \\ &\forall i = 1, \ldots, M-1; \forall j = 1, \ldots, N_{\mathcal{C}_i}: \\ &\qquad {\mathbf{n}_j^{\mathcal{C}_i}}^\top \mathbf{R}_i\mathbf{R}\mathbf{V}_j = 0 \\ &\qquad {\mathbf{n}_j^{\mathcal{C}_i}}^\top (\mathbf{R}_i(\mathbf{R}\mathbf{X}_j + \mathbf{t}) + \mathbf{t}_i) = 0 \end{aligned}\right\} \quad (19)$$

A least-squares solution is then obtained by minimizing the squared error of the system, which can be solved via standard algorithms like *Levenberg-Marquardt* with the initialization obtained from the direct solver. Note, that this step is optional, and only executed for the overdetermined $n > 3$ case if the line parameters are noisy. Experiments show, that the initialization given by the direct solver is sufficiently stable, hence only a few iterations are needed to reach the optimum. The proposed algorithm, that we call *Multi-view RPnL* (MRPnL) is summarized in Algorithm 1.

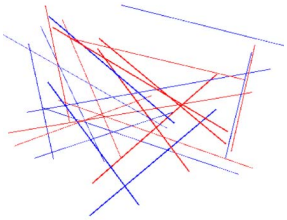

Figure 2. Illustration of 10% 2D noise on 10 random lines placed on 1 plane. Red is original, blue one is the noisy.

4. Experimental Results

For the quantitative evaluation of the proposed method, synthetic datasets were generated using the calibration parameters of a real perspective camera, with available physical parameters (sensor size), enabling us to represent our 3D scene in an equivalent metric space. Multiple sets of 1000 samples were generated containing 3D-2D line pairs. The 3D scene was created with a typical urban or indoor environment in mind, where only few planar surfaces are usually visible in a camera at once, thus we created 3 planes, placing them randomly (with a rotation of $\pm 30°$ around all 3 axes and $\pm[1-2]$ m horizontal and vertical translation and $\pm[0.5-1.5]$ m in depth) in the 3D space, each containing 20 random line segments, with a minimum length of 0.5 m. The 2D data was then generated by capturing images of the scene, practically projecting the 3D lines with a virtual camera of 2378x1580 pixel resolution. In each scene we placed 5 cameras with a random rotation of $\pm 50°$ around all 3 axes, and random translation of ± 1 m in the horizontal and vertical direction, while in the optical axis' direction at $[4-6]$ m from the scene.

The estimated pose was evaluated in terms of the angular distance ϵ, that represents the overall rotation error, and also in terms of translation error as the norm of the difference between the ground truth and estimated translation. To evaluate the robustness of the methods against noisy line detections, random noise was added to the line parameters. Practically we simulated noise by corrupting one endpoint of the line (similarly in 2D and 3D) as adding a random number to each coordinate of the point up to the specified percentage of the actual coordinate value. The unit direction vector was also modified in the same manner. We show results for 3%, 10% and 15% 2D and 3D noise levels, that translate to an average shift on the image of 22px with 3% noise up to 110px with 15% noise (example of 10% noise can be seen in Fig. 2).

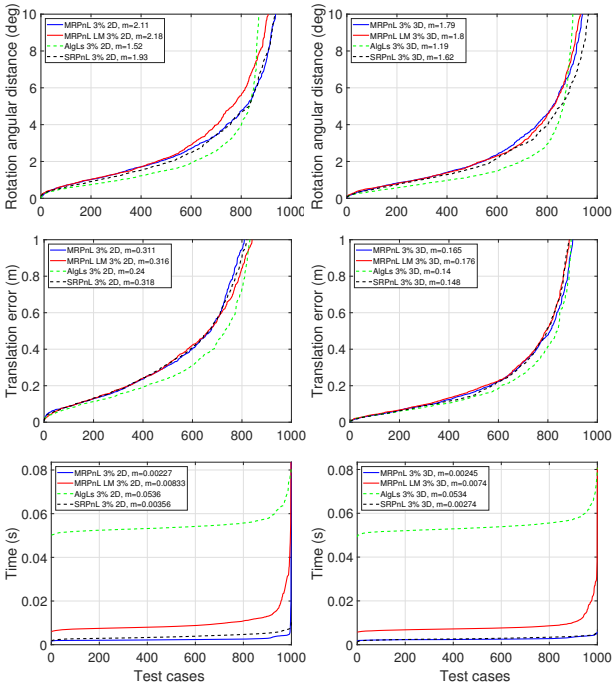

Figure 3. Rotation and translation error, and CPU time for MRPnL with and without refinement, SRPnL, and the globally optimal AlgLS with 3% noise and $n = 3$ lines, except for AlgLs $n = 4$. First column is with 2D noise, second column with 3D noise.

4.1. Comparison with State-of-the-Art

First, the performance of the proposed MRPnL direct solver and MRPnL LM containing the refinement step in Section 3.3 is evaluated for single camera case. Wang *et al*. published a recent comparison [35] of 6 state of the art methods with their SRPnL method, which proved to be dominating both in terms of CPU time as well as efficiency and robustness. Therefore herein, we only focus on comparing the proposed MRPnL algorithm with the most competitive methods from [35]: SRPnL and AlgLS [23].

The Matlab implementation of the competing methods are available from [35]. For a fair comparison, we used the automatic generator of Kukelova *et al*. [13], that provides a Matlab-based solver, but we remark that we also successfully used Kneip's generator [11], which produces a solver in C++ that is much faster. All experiments were done on an $i7$ computer with 16 GB of RAM.

Comparisons were performed in two different setups, first using the minimum number of line matches that each algorithm requires, then using all 60 line pairs of the scene, with only a single camera, since the formulation of SRPnL and AlgLS doesn't cover multi-view setups.

For the first setup, using $n = 4$ lines for AlgLS and $n = 3$ for the other methods, the obtained results are shown in Fig. 3. Results on plots are always sorted based on the error from best to worst. In this setup all the methods perform very similar in terms of median errors of the pose parameters, only AlgLS produces lower errors due to the higher number of line-pairs it is using ($n = 4$). The methods are robust for up to 3% 2D noise, where the median ϵ already reaches above 2°, only exception is AlgLS with 1.52°. In terms of runtime, MRPnL is the fastest with 2.4ms, then SRPnL follows with 3.5ms as the highest median runtime among multiple data sets. MRPnL LM takes more time (8.3ms), but is still much faster than AlgLS (53ms).

Using $n = 60$ line-matches, AlgLS and MRPnL LM have the best results with the lowest median rotation and translation errors, with 15% noise MRPnL LM handles better the 3D noise, while AlgLS favors the noise in 2D domain, both of them having the median ϵ below 1.5° with 2D noise and 1.05° with 3D noise (see Fig. 4 for results). The only other method that handles well 15% noise both in 2D and 3D domain is MRPnL (median ϵ 2.5° and 2.17° respectively, and translation error of 0.21m and 0.1m), while SRPnL can only obtain similar errors with up to 5% noise. Since MRPnL robustly provides a good initialization, LM refinement usually performs only 5 iterations, keeping the runtime comparable with SRPnL and MRPnL.

Based on the data presented in [35], we can confirm that the CPU time of the methods does not change significantly for $n < 200$ lines. Since in a realistic dataset of an urban environment we shouldn't expect to have such many inlier pairs (*e.g*. the large scale dataset presented by [22] also uses an average of 130 lines per image), we did not find an evaluation with hundreds of lines relevant.

4.2. Multi-view Case

The multi-view configuration presented in Section 3.1 with LM refinement (Section 3.3), was tested on data with 10% 2D noise using 5 cameras. Results are shown in Fig. 5, where –as a baseline– we also show the results achieved with a single camera. Clearly, the accuracy of pose estimates are consistent over all cameras. As for the CPU time, one can see that it scales with the number of cameras but still remains under 56 ms for 5 cameras, which is slightly faster than AlgLS for a single camera. It is thus clear that the proposed MRPnL LM algorithm performs well in a multi-view setup, median rotation errors remain below 1° and the translation below 22 cm.

4.3. Robustness to Outliers

Since in a multi-view system, filtering the outliers has to be performed for each view independently from the others, we evaluate the robustness to outliers on a single camera only. The proposed MRPnL direct solver proved to be the fastest and more robust to noise than SRPnL, thus it is well suited for outlier detection in a RANSAC algorithm (see Section 3.2). In our experiments, we used the built in M-estimator sample consensus (MSAC) algorithm function

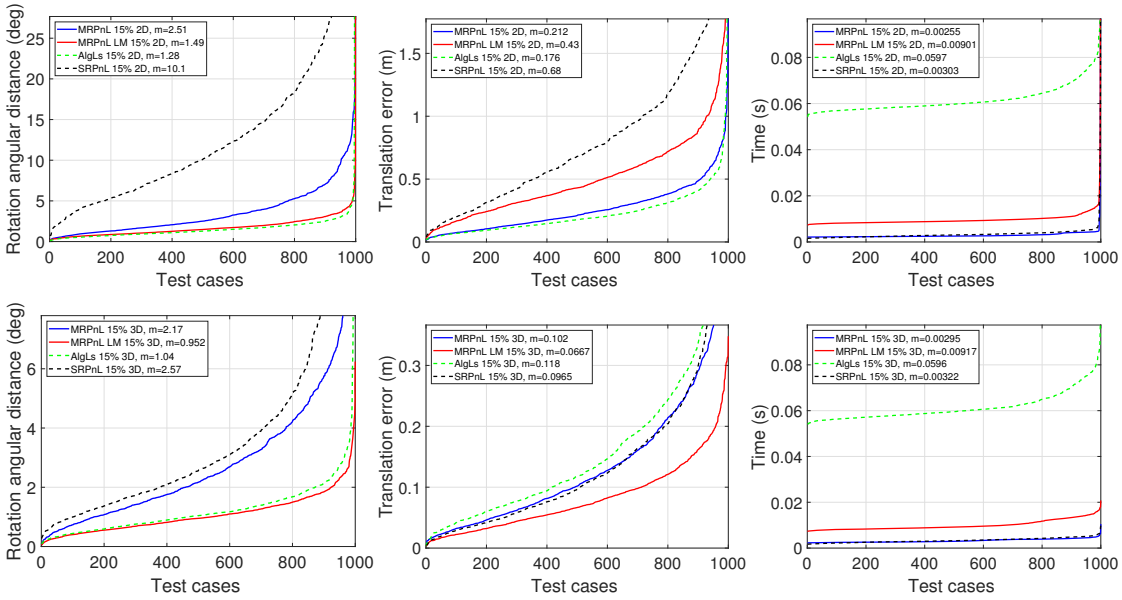

Figure 4. Rotation and translation error, and CPU time for MRPnL with and without refinement, SRPnL and the globally optimal AlgLS with 15% noise and all 60 lines used, first row with 2D noise, second row with 3D noise.

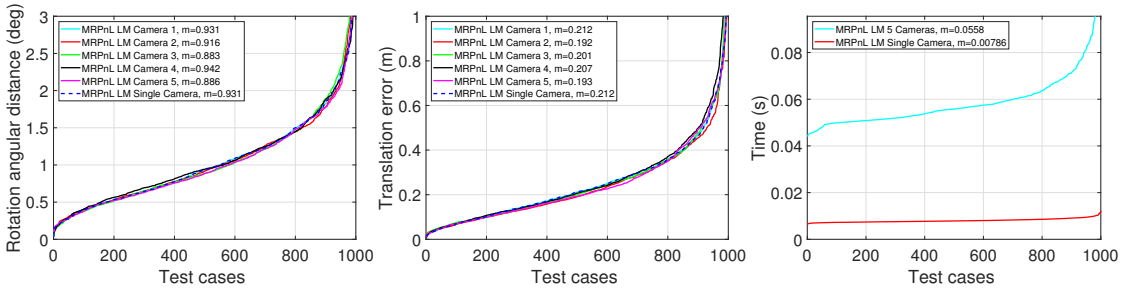

Figure 5. Overall rotation and translation error, and CPU time for MRPnL LM in a multi-view setup with 5 cameras, using 10% 2D noise.

of Matlab [30] together with the backprojection error presented in (18).

The synthetic dataset previously defined was extended by adding a specific number of outlier 2D-3D line-pairs with randomly generated coordinates, to obtain the outlier ratio of: 30% and 60% (26, 90 outliers respectively). The threshold for RANSAC was experimentally determined as the average between the maximum of the inliers' and minimum of the outliers' backprojection error calculated with the reference pose. In our tests, RANSAC with MRPnL was able to robustly filter out all outliers in most test cases, since there was a clear separation between the inliers and outliers, but we found that only a smaller inlier set can be obtained if the outlier lines are taken randomly from the same planes as the inliers, thus they are not different enough from the correct lines. Pose estimation errors of MRPnL on the obtained inliers with 10% noise and 60% outlier ratio are very similar to the baseline results obtained using only the inliers, we can observe an increase only in median translation errors, as shown in Fig. 6. The expense of such filtering is visible in the runtime plot in Fig. 6, where 30% outliers can be filtered relatively fast, but 60% outliers in almost two seconds.

4.4. Real Data

To evaluate the proposed algorithm on real data, we used a 2D-3D dataset captured in an outdoor urban environment, containing a dense 3D pointcloud of the scene, captured with a Riegl VZ400 Lidar (with an attached Nikon DSLR providing reference RGB images), and 2D 4K resolution video sequences captured by a flying UAV along different trajectories. We extracted intermittent frames from the video sequence (16 frames from a sequence of 1800), to better evaluate the robustness to changing scene and lighting conditions. The ground truth pose of each camera was estimated with UPnP [12] using reflective markers placed on the scene, automatically detected by the scanner in 3D, and manually selected and matched in 2D. For many multi-view localization and visual odometry applications the most important criteria of a good result is the correct projection between 2D and 3D domain. This can be evaluated easily by

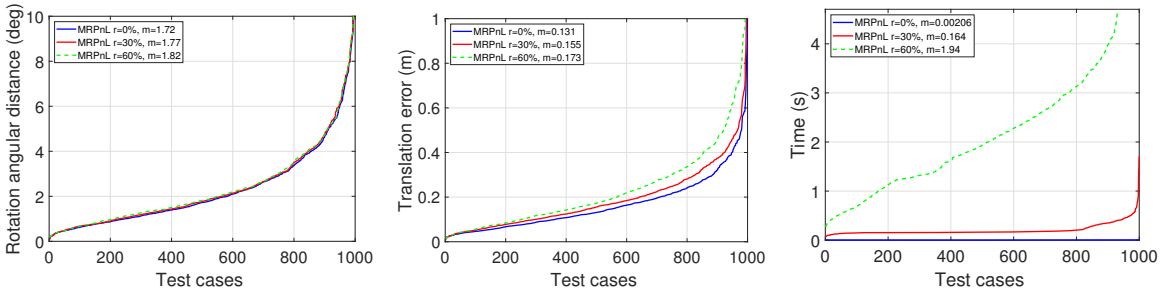

Figure 6. MRPnL LM pose estimation results on the inlier line-pairs obtained by RANSAC with 10% noise and $r = 30\%$ $r = 60\%$ outlier ratio, compared to the baseline results without RANSAC on the inlier set ($r = 0\%$).

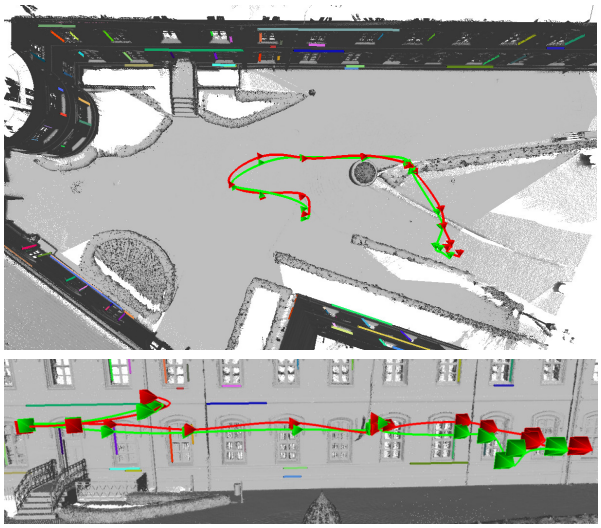

Figure 7. MRPnL LM trajectory estimation results on 16 frames of a longer drone sequence. Ground truth camera poses and the trajectory are shown in green, the estimated ones in red, while the used 3D lines (81 in total) are also visible (better viewed in color).

forward projection error measured in the marker points (for the ground truth poses the maximum error was 12 cm, and the median 4 cm). 2D lines on the frames were extracted using OpenCV LSD detector [32]. In order to have a known inlier set, 2D-3D matching is done by manually matching 2D lines to lines detected on the LIDAR reference RGB images, that directly provides the corresponding 3D lines. Other methods (*e.g.* [22]) rely on VisualSFM, and use the estimated camera poses to project 2D lines into 3D, while with our approach, learnable line segment descriptors [31] could also be used for automatic 2D-2D matching.

Despite the fact that only a relatively small number of lines were used (an average of 13 lines and maximum 21 lines per image, compared to *e.g.* [22], where they used 130 lines and 50 points per image), the proposed MRPnL solver can estimate the absolute and relative poses quite robustly, 15 out of the 16 frames have a maximum forward projection error of less than 30 cm, the median forward projection error being 8 cm. Applying the LM refinement to the whole system increased the algorithm runtime from 80ms to 480ms, but all error measures show improvement, median forward projection error is reduced to 7.4 cm. Results of the camera path estimation can be seen in Fig. 7, where green marks the ground truth trajectory and camera positions and red the estimated one. All the used 81 3D lines are also shown in random colors.

Up to 30% outlier ratio is well tolerated in the real case too, showing similar results as above, even if the outliers are quite similar to the inliers, since they are randomly selected from the same visible scene planes. Results show similar performance to the synthetic experiments, errors increase only when the number of inliers gets drastically reduced by the outlier filtering due to no clear separation between inliers and outliers. We have shown, that the proposed MRPnL method is able to handle the challenging path estimation of 4DoF quadrotor UAVs that can have an unsystematic movement, making sudden turns, floating around in any direction.

5. Conclusion

A novel algebraic approach has been proposed for computing the absolute and relative poses of a multi-view perspective camera system using line correspondences, which works without reformulation both for minimal problems (3 line pairs per camera) as well as for the general $n > 3$ and multiple camera cases. Unlike previous approaches [37, 36, 35], rotation is solved first through a two-variate 7-th order system of polynomial equations using a Grobner basis direct LSE solver, which reduces numerical error propagation and works both for minimal and general line sets. Then the translation is solved via a linear system. Experimental tests on large synthetic as well as real datasets confirm the state of the art performance of the proposed algorithm. Comparative results show that our method outperforms recent alternative methods (AlgLS [23], ASPnL [36], SRPnL [35]) in terms of speed, accuracy, and robustness. Furthermore, unlike these methods, our algorithm works for multi-view scenarios and is robust up to 60% outlier ratio when combined with a RANSAC-like method.

References

[1] H. Abdellali and Z. Kato. Absolute and relative pose estimation of a multi-view camera system using 2d-3d line pairs and vertical direction. In *Proceedings of International Conference on Digital Image Computing: Techniques and Applications*, pages 1–8, Canberra, Australia, Dec. 2018. IEEE. 2, 3, 4

[2] C. Albl, Z. Kukelova, and T. Pajdla. Rolling shutter absolute pose problem with known vertical direction. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 3355–3363, Las Vegas, NV, USA, June 2016. 3

[3] A. Bartoli and P. Sturm. The 3D line motion matrix and alignment of line reconstructions. *International Journal of Computer Vision*, 57(3):159–178, 2004. 2

[4] F. Camposeco, T. Sattler, and M. Pollefeys. Minimal solvers for generalized pose and scale estimation from two rays and one point. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Proceedings of European Conference Computer Vision*, volume 9909 of *Lecture Notes in Computer Science*, pages 202–218, Amsterdam, The Netherlands, Oct. 2016. Springer. 1, 2

[5] H. Chen. Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):530–541, 1991. 1, 3

[6] A. Elqursh and A. Elgammal. Line-based relative pose estimation. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 3049–3056, Colorado Springs, CO, USA, June 2011. IEEE. 1

[7] M. A. Fischler and R. 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. 1, 2, 5

[8] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2004. 1, 2, 4

[9] N. Horanyi and Z. Kato. Multiview absolute pose using 3D - 2D perspective line correspondences and vertical direction. In *Proceedings of ICCV Workshop on Multiview Relationships in 3D Data*, pages 1–9, Venice, Italy, Oct. 2017. IEEE. 2, 3, 4

[10] T. Ke and S. I. Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 1–9, Honolulu, HI, USA, July 2017. IEEE. 1, 2

[11] L. Kneip. *Polyjam*, 2015 [online]. url:https://github.com/laurentkneip/polyjam. 6

[12] L. Kneip, H. Li, and Y. Seo. UPnP: an optimal O(n) solution to the absolute pose problem with universal applicability. In D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Proceedings of European Conference Computer Vision, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 127–142, Zurich, Switzerland, Sept. 2014. Springer. 1, 2, 7

[13] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *Proceedings of European Conference on Computer Vision*, pages 302–315. Springer Berlin Heidelberg, 2008. 4, 6

[14] Z. Kukelova, M. Bujnak, and T. Pajdla. Closed-form solutions to minimal absolute pose problems with known vertical direction. In R. Kimmel, R. Klette, and A. Sugimoto, editors, *Proceedings of Asian Conference on Computer Vision, Part II*, volume 6493 of *LNCS*, pages 216–229, Queenstown, New Zealand, Nov. 2010. Springer. 2, 3

[15] V. Larsson, M. Oskarsson, K. Åström, A. Wallis, Z. Kukelova, and T. Pajdla. Beyond grobner bases: Basis selection for minimal solvers. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 3945–3954, Salt Lake City, UT, USA, June 2018. IEEE Computer Society. 4

[16] G. H. Lee. A minimal solution for non-perspective pose estimation from line correspondences. In *Proceedings of European Conference on Computer Vision*, pages 170–185, Amsterdam, The Netherlands, Oct. 2016. Springer. 2, 3, 4, 5

[17] G. H. Lee, F. Fraundorfer, and M. Pollefeys. Motion estimation for self-driving cars with a generalized camera. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 2746–2753, Portland, OR, USA, June 2013. 1

[18] G. H. Lee, M. Pollefeys, and F. Fraundorfer. Relative pose estimation for a multi-camera system with known vertical direction. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 540–547, Columbus, OH, USA, June 2014. 1

[19] V. Lepetit, F.Moreno-Noguer, and P.Fua. EPnP: an accurate O(n) solution to the PnP problem. *International Journal of Computer Vision*, 81(2), 2009. 1, 2

[20] S. Li, C. Xu, and M. Xie. A robust O(n) solution to the perspective-n-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1444–1450, 2012. 1, 2, 3

[21] J. Li-Chee-Ming and C. Armenakis. UAV navigation system using line-based sensor pose estimation. *Geo-spatial Information Science*, 21(1):2–11, jan 2018. 1

[22] P. Miraldo, T. Dias, and S. Ramalingam. A minimal closed-form solution for multi-perspective pose estimation using points and lines. In *The European Conference on Computer Vision*, pages 1–17, Munich, Germany, Sept. 2018. Springer. 2, 6, 8

[23] F. M. Mirzaei and S. I. Roumeliotis. Globally optimal pose estimation from line correspondences. In *International Conference on Robotics and Automation*, pages 5581–5588, Shanghai, China, May 2011. IEEE, IEEE. 2, 6, 8

[24] M. Persson and K. Nordberg. Lambda twist: An accurate fast robust perspective three point (p3p) solver. In *The European Conference on Computer Vision*, pages 1–15, Munich, Germany, Sept. 2018. Springer. 2

[25] H. Pottmann and J. Wallner. *Computational Line Geometry*. Mathematics and Visualization. Springer, 2009. 2

[26] A. Pumarola, A. Vakhitov, A. Agudo, F. Moreno-Noguer, and A. Sanfeliu. Relative localization for aerial manipulation with PL-SLAM. In *Springer Tracts in Advanced Robotics*, pages 239–248. Springer International Publishing, 2019. 2

[27] L. Tamas, R. Frohlich, and Z. Kato. Relative pose estimation and fusion of omnidirectional and lidar cameras. In

L. de Agapito, M. M. Bronstein, and C. Rother, editors, *Proceedings of the ECCV Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving*, volume 8926 of *Lecture Notes in Computer Science*, pages 640–651, Zurich, Switzerland, Sept. 2014. Springer. 1

[28] L. Tamas and Z. Kato. Targetless calibration of a lidar - perspective camera pair. In *Proceedings of ICCV Workshop on Big Data in 3D Computer Vision*, pages 668–675, Sydney, Australia, Dec. 2013. IEEE, IEEE. 1

[29] C. J. Taylor and D. J. Kriegman. Structure and motion from line segments in multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1021–1032, Nov. 1995. 2

[30] P. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, apr 2000. 5, 7

[31] A. Vakhitov, V. Lempitsky, and Y. Zheng. Stereo relative pose from line and point feature triplets. In *Computer Vision – ECCV 2018*, pages 662–677. Springer International Publishing, 2018. 8

[32] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):722–732, apr 2010. 8

[33] N. von Schmude, P. Lothe, and B. Jähne. Relative pose estimation from straight lines using parallel line clustering and its application to monocular visual odometry. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and and Technology Publications, 2016. 1

[34] P. Wang, G. Xu, and Y. Cheng. A novel algebraic solution to the perspective-three-line problem. *Computer Vision and Image Understanding*, 2018. in press. 2, 3

[35] P. Wang, G. Xu, Y. Cheng, and Q. Yu. Camera pose estimation from lines: a fast, robust and general method. *Machine Vision and Applications*, feb 2019. 2, 3, 4, 6, 8

[36] C. Xu, L. Zhang, L. Cheng, and R. Koch. Pose estimation from line correspondences: A complete analysis and a series of solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1209–1222, 2017. 1, 2, 3, 4, 8

[37] L. Zhang, C. Xu, K. Lee, and R. Koch. Robust and efficient pose estimation from line correspondences. In K. M. Lee, Y. Matsushita, J. M. Rehg, and Z. Hu, editors, *Proceesings of Asian Conference on Computer Vision*, volume 7726 of *Lecture Notes in Computer Science*, pages 217–230, Daejeon, Korea, Nov. 2012. Springer. 2, 3, 8

[38] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu. StructSLAM: Visual SLAM with building structure lines. *IEEE Transactions on Vehicular Technology*, 64(4):1364–1375, apr 2015. 1