Multiview Absolute Pose Using 3D – 2D Perspective Line Correspondences and Vertical Direction

Nora Horanyi[1] and Zoltan Kato[1,2]

[1] Institute of Informatics, University of Szeged, P.O. Box 652, H-6701, Szeged, Hungary
[2] Department of Mathematics and Informatics, J. Selye University, Komarno, Slovakia,

horanyi@inf.u-szeged.hu, kato@inf.u-szeged.hu

Abstract

In this paper, we address the problem of estimating the absolute pose of a multiview calibrated perspective camera system from 3D - 2D line correspondences. We assume, that the vertical direction is known, which is often the case when the camera system is coupled with an IMU sensor, but it can also be obtained from vanishing points constructed in the images. Herein, we propose two solutions, both can be used as a minimal solver as well as a least squares solver without reformulation. The first solution consists of a single linear system of equations, while the second solution yields a polynomial equation of degree three in one variable and one systems of linear equations which can be efficiently solved in closed-form. The proposed algorithms have been evaluated on various synthetic datasets as well as on real data. Experimental results confirm state of the art performance both in terms of quality and computing time.

1. Introduction

Absolute pose estimation consists in determining the position and orientation of a camera with respect to a 3D world coordinate frame. It is a fundamental building block 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. This problem is known as non-perspective absolute pose estimation as such a camera system may be modeled as a virtual camera with many projection centers.

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 [6, 13, 14, 8]. Various solutions have been developed for both large n as well as for the $n = 3$ minimal case (see [8] 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 [24] for a detailed overview). Mirzaei *et al.* [17] construct a polynomial system of equations from line correspondences to solve for the camera orientation. The system consists of three 5^{th} order equations and one cubic equation with four unknowns, which yields 40 candidate solutions. They also develop an algorithm for perspective pose estimation from three or more line correspondences [16], where the problem is formulated as a non-linear least-squares and solved as an eigenvalue problem using the Macaulay matrix without a need for initialization. Unfortunately, this algorithm yields 27 solutions, which makes it difficult to identify the correct solution in practical applications.

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 [21, 20] 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 [11] and UAVs, the problem of multiview perspective 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 multiview PnP (NPnP) problem [3, 8, 11, 12].

In this work, we deal with multiview absolute pose estimation from 3D–2D perspective line correspondences (also known as the NPnL problem) with known vertical direction. While several point-based methods exist [3, 8], little work has been done on using line correspondences. One notable work is the minimal NP3L solver of Lee [10], which deals with 6 DOF pose parameter estimation. Today, the vast majority of modern cameras, smart phones, UAVs, and camera mounted mobile platforms are equiped with cheap and precise *inertial measurement unit* (IMU). Such devices provide the vertical direction from which one can calculate 2 rotation angles, thus reducing the free parameters from 6 to 4. The accuracy of this *up-vector* is typically between $0.5^\circ - 0.02^\circ$ [1]. While robust minimal solutions based on point correspondences exist [1, 9, 12], none of these methods work for line correspondences.

In this paper, we propose two new solutions to the NPnL problem with known vertical direction. Both algorithms can be used as a minimal NP3L solver with 3 line correspondences suitable for hypothesis testing like RANSAC. Furthermore, the same algorithms can be used without reformulation for $n > 3$ lines as well as for classical single-view PnL problems. The performance and robustness of the proposed methods have been evaluated on large synthetic datasets as well as on real data.

2. Perspective Projection of Lines

Given a calibrated camera $\mathbf{P}$ and 3D lines L_i in the world coordinate frame, the projection of the lines are 2D lines l_i in the image plane. The perspective 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. A homogeneous 3D point $\mathbf{X}$ is mapped by $\mathbf{P}$ into a homogeneous 2D image point $\mathbf{x}'$ as [7]

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

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 absolute perspective pose estimation [8, 14, 13, 9]: 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 $\mathcal{C}$.

Unlike points, 3D lines may have various representations in the projective space [18, 2]. Plücker line coordinates are popular as they are *complete* (*i.e.* every 3D line can be represented) and allow for a linear projection model similar to (2) [7, 2, 10, 19, 25, 15]. However, Plücker coordinates are not *minimal* because a 3D line has 4 degrees of freedom but its Plücker coordinate is a homogeneous 6-vector. Furthermore, transforming a Plücker line between coordinate frames using a standard homogeneous 4×4 rigid motion matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{3}$$

is indirect, *i.e.* two points of the line have to be transformed and then form their Plücker coordinates again. In [2], a 6×6 3D line motion matrix representation is proposed for the transformation, which allows for a direct and linear transformation at the price of a larger matrix. This approach is also used in [10, 19] for absolute pose estimation.

Herein, 3D lines are 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 (see Fig. 1) [22]. The projection of L in a central perspective camera is a line l in the image plane, 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 L and l lie on the projection plane π passing through the camera projection center $\mathbf{C}$. 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 $l = (\mathbf{v}, \mathbf{x})$ as $\mathbf{n} = (\mathbf{v} \times \mathbf{x})/\|\mathbf{v} \times \mathbf{x}\|$. Since L lies also on π, its direction vector $\mathbf{V}$ is perpendicular to $\mathbf{n}$. Hence

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

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 camera $\mathcal{C}$ frame and $\mathbf{X}^{\mathcal{C}}$ denotes the point $\mathbf{X}$ on L in the camera coordinate frame.

3. Multi-view Projection

Let us now investigate the case when the 3D lines are viewed by N perspective cameras. We assume that the cameras are fully calibrated, *i.e.* their intrinsics $\mathbf{K}^i$ as well as their relative pose $(\mathbf{R}^i, \mathbf{t}^i) : \mathcal{C}^i \rightarrow \mathcal{C}$ with respect to a common camera coordinate frame $\mathcal{C}$ are known. The common coordinate frame $\mathcal{C}$ is often attached to one of the cameras (*e.g.* the left camera in a stereo setup), but a multi-camera system may have a coordinate frame detached from

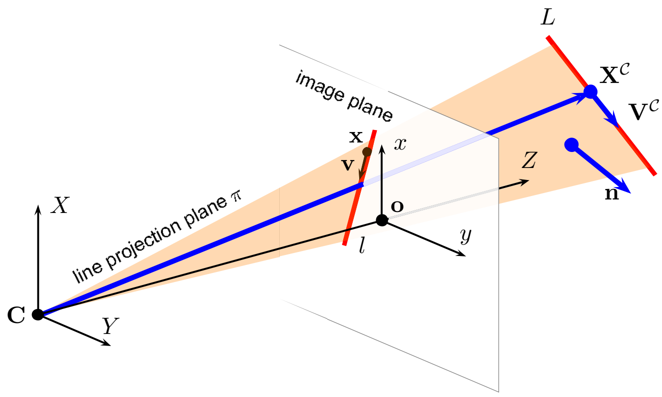

Figure 1: Perspective projection of a 3D line $L \rightarrow l$.

the cameras (*e.g.* the centroid of the mounting frame, or the IMU device). Therefore the absolute pose of the camera system $(\mathbf{R}, \mathbf{t})$ is defined as the rigid transformation acting between $\mathcal{W}$ and $\mathcal{C}$, while individual camera frames $\mathcal{C}^i$ are related to the world coordinate frame via the sequence of rigid transformations

$$\forall i : \mathbf{M}^{\mathcal{W}\rightarrow i} = \begin{bmatrix} \mathbf{R}^i & \mathbf{t}^i \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}. \quad (6)$$

In fact, the whole camera system can be regarded as a generalized non-perspective camera with N projection center [3]. In such a non-central camera, each 3D line L has up to N images $l^i, i = 1 \ldots N$, where N is the number of cameras (or projection centers). Given a pair of corresponding image lines (l^i, l^j) and their projection plane normals $(\mathbf{n}^i, \mathbf{n}^j)$, the unit direction vector $\mathbf{V}^{\mathcal{C}}$ of L can be expressed in the camera frame $\mathcal{C}$ as

$$\mathbf{V}^{\mathcal{C}} = \frac{\mathbf{R}^{i\top}\mathbf{n}^i \times \mathbf{R}^{j\top}\mathbf{n}^j}{\|\mathbf{R}^{i\top}\mathbf{n}^i \times \mathbf{R}^{j\top}\mathbf{n}^j\|}, \quad (7)$$

which yields the following relation

$$\mathbf{V}^{\mathcal{C}} = \mathbf{R}\mathbf{V} \Rightarrow (\mathbf{R}\mathbf{V}) \times (\mathbf{R}^{i\top}\mathbf{n}^i \times \mathbf{R}^{j\top}\mathbf{n}^j) = \mathbf{0} \quad (8)$$

Thus a natural approach to our absolute pose estimation problem would be to reconstruct 3D line directions in the camera frame using *e.g.* (7) and then solving (8) for $\mathbf{R}$. While this is a very attractive, simple, and geometrically intuitive approach, the quality of such a pose estimate would be critically dependent on the reconstruction accuracy, which is known to be quite poor for practically important setups like narrow baseline stereo [4]. In general, (8) becomes numerically unstable whenever $\mathbf{R}^{i\top}\mathbf{n}^i$ and $\mathbf{R}^{j\top}\mathbf{n}^j$ are nearly parallel, which is often the case for narrow baseline and near-parallel principal axes. Furthermore, having a noisy estimate of the normals would severely deteriorate the accuracy of their cross product introducing large errors in a system of equations constructed from (8).

Essentially (8) states that $\mathbf{V}^{\mathcal{C}}$ is perpendicular to both $\mathbf{n}^i$ and $\mathbf{n}^j$, yielding the multi-view form of (4):

$$\forall i \text{ where } L \text{ is visible: } \mathbf{n}^{i\top}\mathbf{R}^i\mathbf{V}^{\mathcal{C}} = \mathbf{n}^{i\top}\mathbf{R}^i\mathbf{R}\mathbf{V} = 0 \quad (9)$$

While this equation is mathematically equivalent to (8), it is numerically more favorable as it provides *separate* equations for each normal thus avoiding multiplication of noisy $\mathbf{n}^i$ measurements. Similarly, (5) can be written for the multi-camera case as:

$$\forall i \text{ where } L \text{ is visible: } \mathbf{n}^{i\top}(\mathbf{R}^i\mathbf{X}^{\mathcal{C}} + \mathbf{t}^i) = \mathbf{n}^{i\top}(\mathbf{R}^i(\mathbf{R}\mathbf{X} + \mathbf{t}) + \mathbf{t}^i) = 0 \quad (10)$$

3.1. Using the Vertical Direction

Let us now have a closer look at the parameterization of $\mathbf{R}$ assuming that the vertical direction is available. This knowledge can be obtained from *e.g.* an *inertial measurement unit* (IMU), which mainly consists of accelerometers capable to measure the earths gravity vector. Alternatively, one can obtain the same information from a calibrated image by detecting a vanishing point (in a man-made environment, we can get the vertical vanishing point, but the knowledge of any other direction would also do) [7]. In the following, we discuss the mathematical representation of this information in $\mathbf{R}$.

Assuming that the camera coordinate system is a standard right-handed system with the X axis pointing up (see Fig. 1), the coordinates of the world vector $(1, 0, 0)^\top$ are known in the camera coordinate frame $\mathcal{C}$. Given this *up-vector*, we can compute rotation $\mathbf{R}_v$ around Y and Z axes, which aligns the world X axis with the camera X axis:

$$\mathbf{R}_v = \mathbf{R}_Z(\gamma)\mathbf{R}_Y(\beta) = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \quad (11)$$

The only unknown parameter in the rotation matrix $\mathbf{R}$ is the rotation $\mathbf{R}_X(\alpha)$ around the vertical X axis:

$$\mathbf{R}_X(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \quad (12)$$

thus the $\mathcal{W} \rightarrow \mathcal{C}$ rotation $\mathbf{R}$ has the following form:

$$\mathbf{R} = \mathbf{R}_v\mathbf{R}_X(\alpha) \quad (13)$$

With this parameterization, the equations (9) and (10) have the form for all camera i where L is visible:

$$\mathbf{n}^{i\top}\mathbf{R}^i\mathbf{R}_v\mathbf{R}_X(\alpha)\mathbf{V} = 0 \quad (14)$$
$$\mathbf{n}^{i\top}(\mathbf{R}^i(\mathbf{R}_v\mathbf{R}_X(\alpha)\mathbf{X} + \mathbf{t}) + \mathbf{t}^i) = 0 \quad (15)$$

4. Efficient Solutions

We aim to compute $\mathbf{R}_X(\alpha)$ and $\mathbf{t}$ using the equations in (14) - (15). We have 4 unknowns: the rotation angle α and the translation components t_1, t_2, t_3. Although each 3D–2D line correspondence $L \leftrightarrow l$ provides 2 equations, only one contains $\mathbf{t}$. Therefore we need at least 3 line correspondences. In the following, we propose two solutions. Both of them use the fact that the images are normalized (*i.e.* image points are multiplied by the inverse of $\mathbf{K}^i$ as in (2)); the relative pose $(\mathbf{R}^i, \mathbf{t}^i)$ of each camera is known w.r.t. the common camera frame $\mathcal{C}$; and the vertical direction is known, *i.e.* the rotation matrix $\mathbf{R}_v$ of (11) is available.

4.1. Linear Solution: NPnLupL

The equations (14) - (15) are linear in $\mathbf{t}$, but $\mathbf{R}_X(\alpha)$ is defined in terms of $\cos(\alpha)$ and $\sin(\alpha)$. Letting these trigonometric functions of α to be two separate unknowns c and s [14, 24, 26], respectively, one can linearize (14) - (15) by substituting

$$(\mathbf{R}^i\mathbf{R}_v)\mathbf{R}_X(\alpha) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & -s \\ 0 & s & c \end{bmatrix} \quad (16)$$

into (14) - (15). Stacking these pairs of equations for n correspondences, we get $2n$ homogeneous linear equations with unknowns $\mathbf{p} = (c, s, t_1, t_2, t_3, 1)^\top$. Hence we have to solve an $\mathbf{A}\mathbf{p} = \mathbf{0}$ system of equations in the least squares sense, which is straightforward using SVD of $\mathbf{A}$. Note that the elements of the $2n \times 6$ matrix $\mathbf{A}$ are expressed in terms of $\mathbf{n}^i$, $\mathbf{R}^i\mathbf{R}_v$, $\mathbf{V}$, $\mathbf{X}$, and $\mathbf{t}^i$ using (14) and (15). Since c and s are estimated as separate unknowns, they may not satisfy the trigonometric constraint $c^2 + s^2 = 1$. Thus they should be normalized before constructing $\mathbf{R}_X(\alpha)$:

$$\cos(\alpha) = \frac{c}{\sqrt{c^2 + s^2}} \quad \text{and} \quad \sin(\alpha) = \frac{s}{\sqrt{c^2 + s^2}} \quad (17)$$

At the price of higher computational complexity, a somewhat more sophisticated normalization involves an additional 3D registration step [14, 24, 26], which aligns the 3D world $\{\mathbf{X}_i\}$ and camera $\{\mathbf{X}_i^{\mathcal{C}}\}$ point sets using a standard 3D registration scheme [23].

A major drawback of this linear solution is that orthonormality constraint on $\mathbf{R}_X(\alpha)$ has been ignored, thus the solution can be quite far from a rigid body transformation for noisy input data. In spite of this, experiments show that our linear solver represents a good tradeoff between accuracy and computing time, yielding quite stable pose estimates under moderate noise levels.

4.2. Cubic Polynomial Solution: NPnLupC

Another way to eliminate $\cos(\alpha)$ and $\sin(\alpha)$ is to use the substitution $q = \tan(\alpha/2)$ [9, 1], for which $\cos(\alpha) = (1 - q^2)/(1 + q^2)$ and $\sin(\alpha) = 2q/(1 + q^2)$. Therefore

$$(1 + q^2)\mathbf{R}_X(q) = \begin{bmatrix} 1 + q^2 & 0 & 0 \\ 0 & 1 - q^2 & -2q \\ 0 & 2q & 1 - q^2 \end{bmatrix}. \quad (18)$$

Substituting $\mathbf{R}_X(q)$ into (14) yields a quadratic equation in the single unknown q. Since we have $n \geq 3$ line correspondences, we obtain n quadratic equations in q:

$$a_i q^2 + b_i q + c_i = 0, \quad i = 1 \dots n, \quad (19)$$

where the coefficients a_i, b_i, c_i are computed in terms of $\mathbf{n}^i$, $\mathbf{R}^i\mathbf{R}_v$, and $\mathbf{V}$ using (14). We will solve this nonlinear system of equations in terms of least square residual. The squared error of the equations is a quartic function in q

$$\sum_{i=1}^{n} \left(a_i^2 q^4 + 2a_i b_i q^3 + (2a_i c_i + b_i^2) q^2 + 2b_i c_i q + c_i^2 \right), \quad (20)$$

whose minima is found by computing the roots of its derivative

$$\sum_{i=1}^{n} \left(4a_i^2 q^3 + 6a_i b_i q^2 + (4a_i c_i + 2b_i^2) q + 2b_i c_i \right) = 0 \quad (21)$$

Such a third order polynomial equation can be solved in a closed form and results in maximum 3 solutions, at least one of them being real. For each real root, we have to determine a $\mathbf{t}$ by back substituting each possible $\mathbf{R}_X(q)$ into (15), which yields a simple linear system of equations in $\mathbf{t}$:

$$\mathbf{n}^{i\top}(\mathbf{X}^i + \mathbf{R}^i\mathbf{t}) = 0, \quad (22)$$

with $\mathbf{X}^i = \mathbf{R}^i\mathbf{R}_v\mathbf{R}_X(q)\mathbf{X} + \mathbf{t}^i$. The final solution is selected by checking that lines are in front of the camera system, or simply by evaluating the reprojection error of each solution and selecting the one with minimal error. In the non-perspective case, the reprojection error characterizes the difference between the observed image line l^i and the reprojected image line $\hat{l}^i$ for all cameras [22]:

$$\epsilon = \sum_{i=1}^{N} \sum_{j=1}^{n} \hat{\mathbf{n}}_j^{i\top} (\mathbf{A}^\top \mathbf{B} \mathbf{A}) \hat{\mathbf{n}}_j^i, \quad \text{with}$$
$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_j^i \\ \mathbf{b}_j^i \end{bmatrix}, \text{ and } \mathbf{B} = \frac{|l_j^i|}{3(n_{j1}^{i2} + n_{j2}^{i2})} \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} \quad (23)$$

where $|l_j^i|$ denotes the length of the image line with the 2D homogeneous endpoints $\mathbf{a}_j^i = (a_{j1}^i, a_{j2}^i, 1)^\top$ and $\mathbf{b}_j^i = (b_{j1}^i, b_{j2}^i, 1)^\top$; $\hat{\mathbf{n}}_j^i$ is the reprojection plane normal in camera i computed from the corresponding 3D line $L_j = (\mathbf{X}_j, \mathbf{V}_j)$ as

$$\hat{\mathbf{n}}_j^i = (\mathbf{R}^i(\mathbf{R}\mathbf{X}_j + \mathbf{t}) + \mathbf{t}^i) \times (\mathbf{R}^i\mathbf{R}\mathbf{V}_j). \quad (24)$$

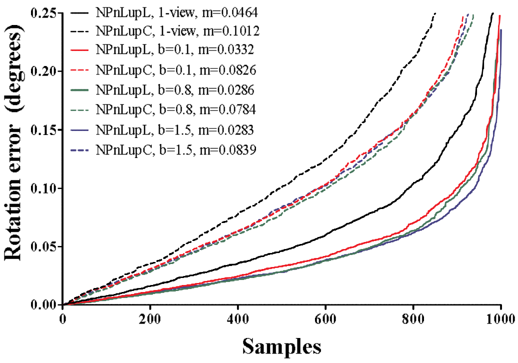

Figure 2: Comparison of the rotational errors w.r.t. the baseline for 30 lines in case of a single-view and stereo system configurations.

n lines	NPnLupL	NPnLupC	RPnL
Run time (s)	0.0009	0.0013	0.0088

3 lines	NP3LupL	NP3LupC	UP3P	NP3L
Run time (s)	0.0004	0.0005	0.0063	0.0298

Table 1: Comparison of the run times of different methods w.r.t number of lines.

The main advantage of this solution is that the trigonometric constraint on α is explicitly taken into account, thus we expect an increased robustness under noisy observations. However, the estimation of α and $\mathbf{t}$ is decoupled, which may lead to slightly less accurate solutions as any error in α is directly propagated into the linear system of $\mathbf{t}$. Furthermore, computational complexity is slightly higher than the purely linear solver.

Finally, Table 1 shows the typical runtime for all tested methods in case of 3 and n lines.

5. Experimental Results

For the quantitative evaluation of our non-perspective pose estimation algorithm with line correspondences, we generated various benchmark datasets of 3D–2D line pairs. 3D lines were generated by placing three 2D planes in the 3D Euclidean space and about 10 lines were placed on each of these planes, whose size was normalized into a 1m^3 cube. Then the planes were placed relative to each other with a random translation of $0 - 1$unit and rotation of $0^\circ, \ldots, 45^\circ$ around the Z axis and $20^\circ, \ldots, 60^\circ$ around the vertical X axis. This arrangement of the 3D lines was motivated by common urban structural properties, in which environment the real data experiments were performed too.

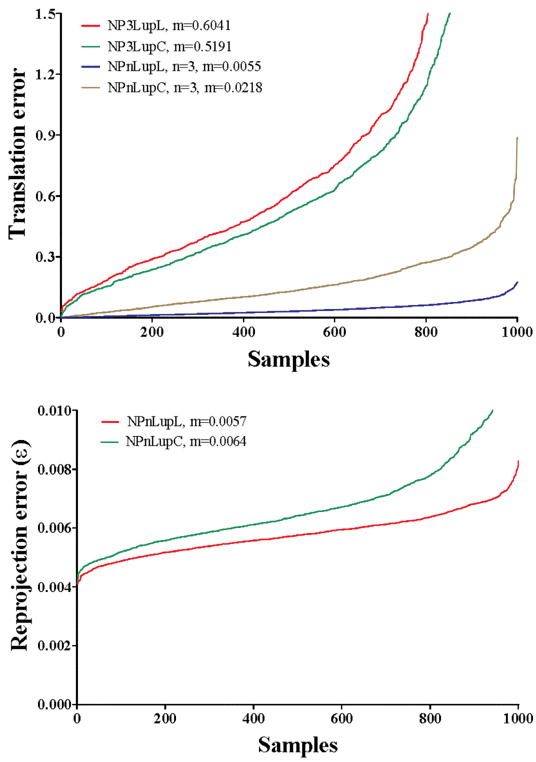

Figure 3: Upper plot shows the translation errors in case of 3 camera for both n and 3 lines. The plot below shows the mean re-projection errors ϵ of the proposed algorithms for n lines.

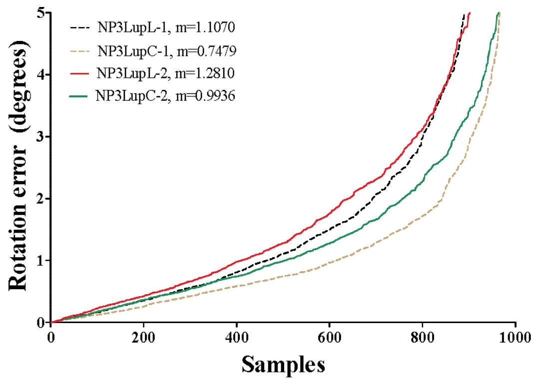

Figure 4: Comparison of two possible minimal case scenarios in a 3-camera system. Dashed lines: one 3D line and its corresponding 2D lines from each camera; connected lines: a different 3D–2D line pair for each camera.

The synthetic 2D images of the 3D lines were generated with a camera system being rotated in the range of $-20^\circ, \ldots, 20^\circ$ around all three axis and random displacement of 9units. All cameras had identical intrinsic param-

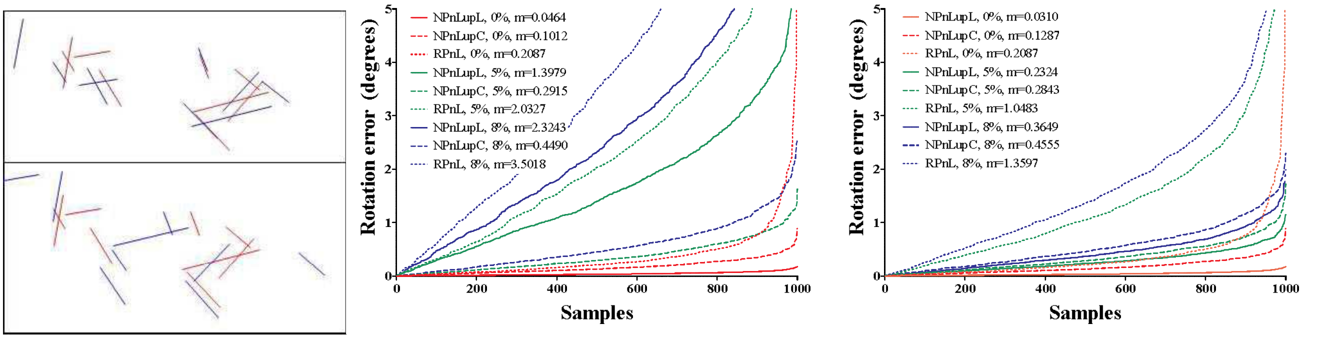

Figure 5: Comparison of the effect of various noise types (2D and 3D) and levels (5% and 8%) in a 3-camera system configuration. The first figure shows a sample of our noisy synthetic data with 5% (up) and 8% noise level (down). Red lines are the originals while blue ones are the noisy lines. The middle figure compares various state-of-the-art solvers' rotation errors w.r.t. different 2D noise levels. The last figure presents the same experiments with different 3D noise levels.

eters $\mathbf{K}$: focal length $f_x = 800, f_y = 800$, and the principal point $\mathbf{o}$ was set to the center of the 1024×768 image plane. Thus random 2D projections were obtained with the so defined random camera matrix $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$. Separate datasets were generated for single camera, standard stereo, and multi-camera systems emulated with 3 cameras. In case of the standard stereo setup, where the right camera is only horizontally translated, we used three different baselines of 0.1, 0.8 and 1.5. For the three-camera setup, datasets were generated with random translations corresponding to 0.05, 0.15, and 0.4 baselines and random relative rotation between the cameras around the Y and Z axis in range of $-5^\circ, \ldots, 5^\circ$, and around the X axis $15^\circ, \ldots, 25^\circ$. Our algorithms were implemented in Matlab and run on a standard desktop computer.

The evaluation of the algorithms were done in two scenarios: either all of the 3D–2D line pairs are used (about 30 line pairs per sample - this will be denoted by n) or only the minimum number line pairs are used (3 line pairs - this will be denoted by 3). We also compare our results with state of the art methods. Since to the best of our knowledge, there is no prior method for non-perspective pose estimation from line correspondences and known vertical direction, we compare our method with the line-based single view RPnL algorithm [26] of Zhang *et al.*; the point-based non-perspective UPnP [8] of Kneip *et al.*; and the line-based non-perspective minimal solver NP3L [10] of Lee.

First, we evaluate the sensitivity of our algorithms with n lines for the baseline in case of the standard stereo setup (parallel optical axes, only horizontal translation between cameras), which is the most challenging configuration as projections planes are nearly parallel for narrow baselines. Fig. 2 shows, that for n lines, the linear solver is more accurate, but overall both methods perform quite well independently of the baseline length, having a median rotation error less than 0.11° in all samples.

Next, we compare the performance in the minimal and n-line cases. Fig. 3 shows the translation error in case of n lines as well as 3 lines. Obviously, the algorithms perform better for n lines, but overall the estimates are quite accurate. Note also that the cubic solver outperforms the linear one in the minimal case. In Fig. 4, we compare two possible minimal case scenarios in a 3-camera system: 1) one 3D line and its corresponding 2D lines from each camera; 2) a different 3D–2D line pair for each camera. The first case is useful when 3D lines are limited but there is no occlusion. The second scenario corresponds to occlusions, when not all 3D lines are visible form all cameras. The accuracy of our algorithms are not influenced by these differences.

5.1. Robustness

In order to evaluate the sensitivity of our algorithms to line measurement noise, we add random noise to the generated test cases in the following way: The 2D lines are corrupted with additive random noise on one endpoint of the line and the direction vector of the line. The amount of noise is 5% and 8%, meaning that a random number is added to each coordinate up to the specified percentage of the actual coordinate value. This corresponds to a quite high noise rate: $[-20, +20]$ pixels (4 pixels mean and 12 pixels standard deviation) for the 5% case and $[-30, +30]$ pixels (4 pixels mean and 20 pixels standard deviation) for the 8% case.

In Fig. 5, we compared the robustness of the proposed algorithms in a 3-camera system. NPnLupL outperforms NPnLupC and RPnL at 0% and for all 3D noise levels, but for 2D noise NPnLupC performs better than the other two. RPnL is consistently outperformed by our solvers in all cases. Of course, we know the vertical direction, hence RPnL has to solve a more difficult task. However, if an IMU is available, then it is clearly worth to use this vertical information instead of relying on purely visual data.

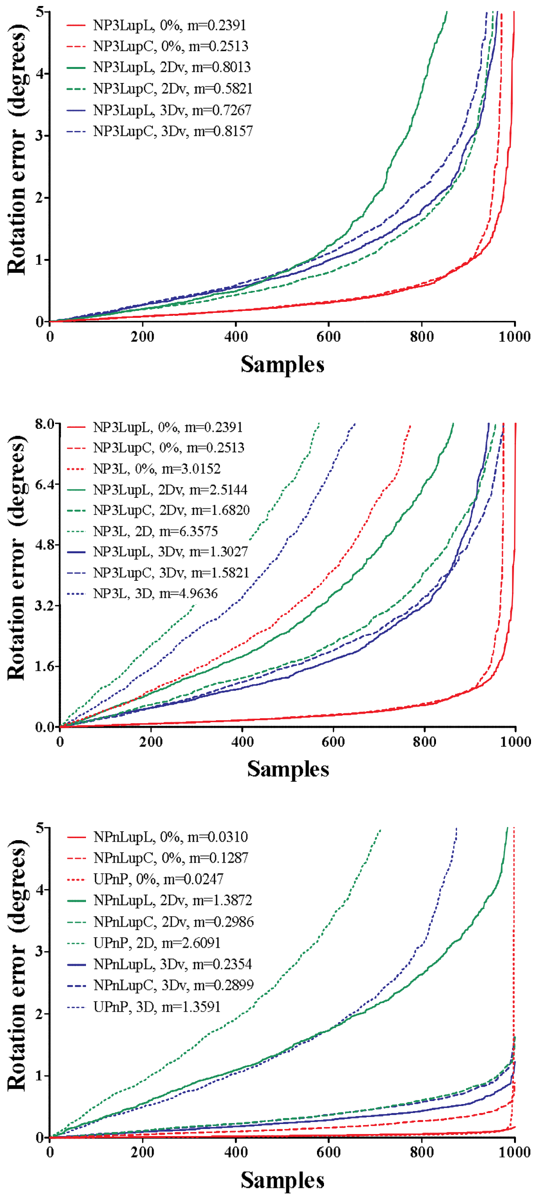

Figure 6: Comparison of various configurations and methods w.r.t varying line numbers and varying noise levels(2D: 5% 2D noise, 2Dv: 5% 2D noise with 0.5° vertical noise, 3D: 5% 3D noise, 3Dv: 5% 3D noise with 0.5° vertical noise, m:median error value). The plot on the top indicates the efficiency of our minimal solutions ($n = 3$) in standard stereo configuration. The middle plot compares the NP3L minimal solver with three cameras. The last plot compares the UPnP and our least square solvers with three cameras.

To have a fair comparison with the other methods, we also added a $\pm 0.5^\circ$ random noise to the vertical direction (this is a typical noise level of a low quality IMU), and then run our algorithms on the 5% noisy datasets. The comparative results of these experiments are shown in Fig. 6, where we show error plots for the standard stereo setup minimal case, 3-camera system minimal case (compared with NP3L [10]), as well as 3-camera case with n-lines (compared with UPnP [8]. NP3L [10] is consistently outperformed by our methods, while for UPnP [8] our NPnLupC outperforms it in the noisy cases. Note, however, that in the maximal case when we have 3 cameras this algorithm has more than 150 point pairs to work with (we used 2 points per line).

Fig. 7	NPnLupL	NPnLupC	UPnP
Rotation error (deg)	0.0177	0.0191	0.0694
Translation error	0.0356	0.0262	0.0141
Fig. 8	NPnLupL	NPnLupC	UPnP
Rotation error (deg)	0.0166	0.0176	0.0498
Translation error	0.0402	0.0319	0.0119

Table 2: Comparison of the maximal rotational and translational error of various methods on the real data.

5.2. Real Data

In Fig. 7 and Fig. 8, we show two real datasets. 2D images were captured with a standard Canon DSLR camera while the 3D point cloud was captured with a Riegl VZ400 Lidar scanner with an angular resolution of 0.05°. Each camera location is shown in the Lidar coordinate system as well as the 3D lines used for pose estimation. The corresponding 3D-2D lines are shown with the same color as the camera. Pose estimation errors are shown in Table 2 in comparison with UPnP [8].

RPnL is not included in these tests because it works only for a single camera while in these test cases we had a multiview camera system of three and four cameras. As for NP3L, we used the Matlab implementation provided by the author of [10], which implemented only three cameras with the minimal three line correspondences, hence we could not run it neither with n-lines nor on the 4-camera test case. Evaluation on synthetic data already shown the performance of our method compared to RPnL and NP3L. The purpose of this test was to show that our line-based method is able to provide state-of-the-art estimates under real conditions, just like the point-based UPnP. Although our translation error is approximately two times larger than for UPnP, its rotation error is almost four times bigger than ours. Furthermore, UPnP was using two times more correspondences (two endpoints of each line) than our method. It is thus fair to say, that both methods perform pretty well, as the errors are almost negligible (see Table 2).

Figure 7: Lidar laser scan for testing our pose estimation algorithms with 4-camera system. 2D detected lines are shown next to the 3D point cloud which colors are the same to their corresponding camera.

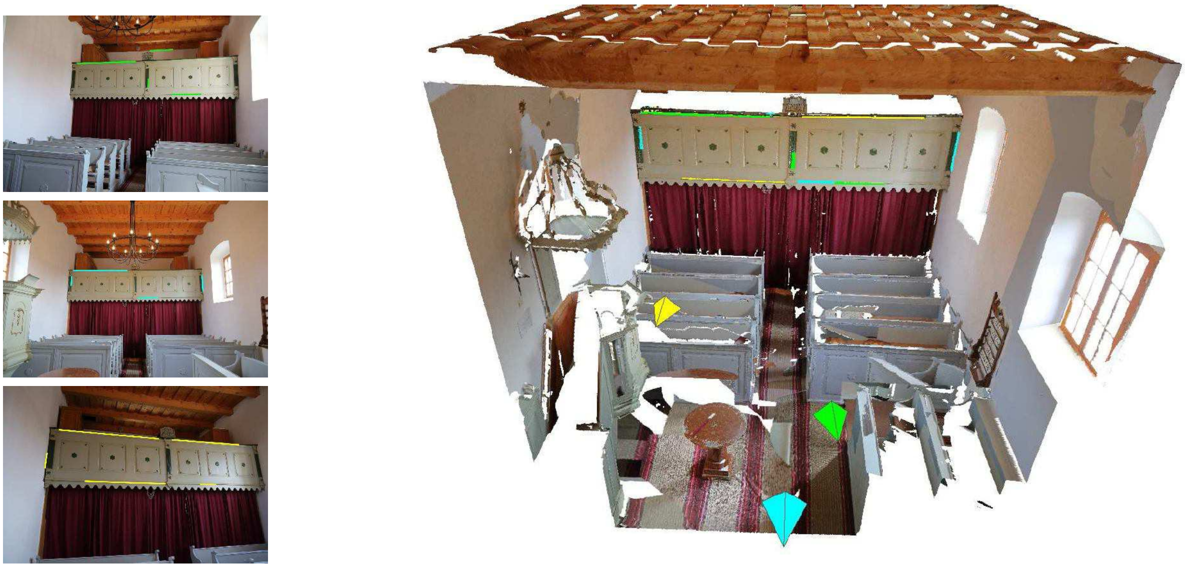

Figure 8: Lidar laser scan for testing our pose estimation algorithms with 3-camera system. 2D detected lines are shown next to the 3D point cloud which colors are the same to their corresponding camera.

6. Conclusion

We proposed a linear and a cubic solutions to the NPnL problem from line correspondences with known vertical direction. Both method can be used as a minimal solver (*e.g.* within RANSAC) as well as a general least squares solver. The methods work for single- and multi-view camera systems without reformulation. The minimal number of line correspondences has been discussed for various common camera configurations. The proposed methods have been evaluated on synthetic and real datasets. While the linear solver is computationally more efficient, it is more sensitive to noise and low number of correspondences, while the cubic solver is much more robust at the price of a slightly increased CPU time. Both methods compare favorably to state of the art approaches.

Acknowledgement

This work was partially supported by the NKFI-6 fund through project K120366; the Agence Universitaire de la Francophonie (AUF) and the Romanian Institute for Atomic Physics (IFA), through the AUF-RO project NETASSIST; the Re- search & 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.

References

[1] 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. 2, 4

[2] 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

[3] 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. 2, 3

[4] M. K. Chandraker, J. Lim, and D. J. Kriegman. Moving in stereo: Efficient structure and motion using lines. In *International Conference on Computer Vision*, pages 1741–1748, Kyoto, Japan, Oct. 2009. 3

[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

[6] 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

[7] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2004. 2, 3

[8] 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, 6, 7

[9] 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, 4

[10] 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, 6, 7

[11] 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, 2

[12] 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. 2

[13] 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

[14] 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, 4

[15] P. Miraldo, H. Araujo, and N. Gonalves. Pose estimation for general cameras using lines. *IEEE Transactions on Cybernetics*, 45(10):2156–2164, Oct. 2015. 2

[16] 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. 1

[17] F. M. Mirzaei and S. I. Roumeliotis. Optimal estimation of vanishing points in a Manhattan world. In *International Conference on Computer Vision*, pages 2454–2461, Barcelona, Spain, Nov. 2011. IEEE, IEEE Computer Society. 1

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

[19] B. Pribyl, P. Zemcík, and M. Cadík. Camera pose estimation from lines using Plücker coordinates. In X. Xie, M. W. Jones, and G. K. L. Tam, editors, *Proceedings of the British Machine Vision Conference*, pages 45.1–45.12, Swansea, UK, Sept. 2015. BMVA Press. 2

[20] 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

[21] 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

[22] 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, 4

[23] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. 4

[24] 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*, 2016. 1, 4

[25] L. Zhang and R. Koch. Hand-held monocular SLAM based on line segments. In *Proceedings of the Irish Machine Vision and Image Processing Conference*, pages 7–14, Dublin, Ireland, 2011. IEEE Computer Society. 2

[26] 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. 4, 6