Robust and Efficient Pose Estimation from Line Correspondences

Lilian Zhang[1], Chi Xu[2], Kok-Meng Lee[3], and Reinhard Koch[1]

[1] Institute of Computer Science, University of Kiel, Germany
[2] Bioinformatics Institute, A*STAR, Singapore
[3] School of Mechanical Engineering, Georgia Institute of Technology, USA

Abstract. We propose a non-iterative solution for the Perspective-n-Line (PnL) problem, which can efficiently and accurately estimate the camera pose for both small number and large number of line correspondences. By selecting a rotation axis in the camera framework, the reference lines are divided into triplets to form a sixteenth order cost function, and then the optimum is retrieved from the roots of the derivative of the cost function by evaluating the orthogonal errors and the reprojected errors of the local minima. The final pose estimation is normalized by a 3D alignment approach. The advantages of the proposed method are as follows: (1) it stably retrieves the optimum of the solution with very little computational complexity and high accuracy; (2) small line sets can be robustly handled to achieve highly accurate results and; (3) large line sets can be efficiently handled because it is O(n).

1 Introduction

Determining the pose of a calibrated camera from n correspondences between 3D reference features and their 2D projections has numerous applications in robotics [1], computer vision [2] and augmented reality [3]. For point features, the Perspective-n-Point (PnP) problem has been well studied with some recent remarkable progress [4,5,6]. For line features, the Perspective-n-Line (PnL) problem remains a challenging topic although the recent work of Mirzaei and Roumeliotis [7] achieves great improvement. The reason is that applications in practice require solutions to keep accurate for small number of correspondences and to remain efficiency when dealing with many correspondences.

To address the PnL problem, in this paper we propose a Robust Perspective-n-Line (RPnL) solution which is the counterpart of the Robust Perspective-n-Point (RPnP) algorithm [6]. In the framework of RPnL, the camera orientation is parametrized by a rotation axis and an angle around this axis. We first divide the line correspondences into a set of line triplets by selecting a rotation axis. For each of the line triplet, we build an eighth order polynomial. Then the rotation axis in the camera frame is estimated by picking the global minimum of a cost function in a least square sense. After the estimation of the rotation axis, the rotation angle and translation vector are solved by SVD efficiently. A 3D alignment scheme [8] is employed to normalize the estimated camera pose.

K.M. Lee et al. (Eds.): ACCV 2012, Part III, LNCS 7726, pp. 217–230, 2013.

As demonstrated in the experimental results, the advantages of RPnL compared to the existing solutions of the PnL problem are as follows: (i) it's one of the non-iterative algorithms, hence no initialization is required; Furthermore, the computational complexity of RPnL is linear in the number of correspondences; (ii) When only a few line correspondences are available ($3 < n \leqslant 5$), RPnL drastically improve the accuracy and robustness compared to existing methods. (iii) When there are many line correspondences, RPnL achieves the same accuracy as the State-of-the-art methods in a fraction of their computational time.

The remainder of the paper is organized as follows. After a brief review of the related work in Sec.2, we derive the Perspective-3-Line (P3L) polynomial for the P3L problem in Sec.3. Then the framework of RPnL algorithm is introduced in Sec.4. We evaluate the performance of RPnL and the state-of-the-art methods with extensive simulations and validate the proposed algorithm on the real image sequences as well in Sec.5. Finally, the conclusion is drawn in Sec.6.

2 Related Work

The problem of estimating camera pose from 2D-3D line correspondences has been addressed for more than two decades. In one of the earliest works, Dhome *et al.* [9] proposed a closed-form solution for the P3L problem by a polynomial approach. To derive the P3L polynomial, 3D lines are transformed into a model coordinate system and 2D lines are transformed into a virtual image plane in the viewer coordinate system. Later, Chen [1] proposed another approach to derive the P3L polynomial by introducing a canonical configuration. Unfortunately, the description of the overall rotation was unclear because three rotation axes in Eq.(4) of [1] were neither aligned with the camera coordinate frame nor aligned with the world coordinate frame. Some extra rotation are required to align the coordinate systems. We give a simpler and clearer derivation of the P3L polynomial by fixing a rotation axis and enforcing the orthogonal constraints in Sec.3.

The solutions of P3L problem are not uniquely determined because there may exist 8 solutions [1]. To find a unique pose solution, at least four line correspondences should be established. Liu *et al.* [10] propose an iterative method to first estimate the camera orientation and then the translation. Kumar and Hanson [11] improve the iterative algorithm which estimates the camera orientation and translation simultaneously (named as R_and_T). Christy and Horaud [12] propose an iterative algorithm to estimate the camera pose with either a weak perspective or a para-perspective camera model. It's well known that these iterative algorithms require an initialization and may converge to a local minimum. Besides, in the absence of a good initialization, the computational cost of the iterative algorithms is generally high.

In the work of Liu *et al.* [10], a non-iterative algorithm is also proposed when there are more than eight line correspondences. Ansar and Daniilidis [13] improve this algorithm to make it work for four or more line correspondences by employing the lifting approach to convert the polynomial system to a linear system about the components of the rotation matrix. This algorithm has $O(n^2)$

computational complexity and its accuracy is severely affected by the image noise. Recently, Mirzaei and Roumeliotis [7] present an algebraic approach to estimate the global optimum of the camera pose. The algorithm is non-iterative and has $O(n)$ computational complexity. The camera orientation is represented by the Cayley-Gibbs-Rodriguez (CGR) parametrization in their work. After relaxing the constraints on the rotation matrix, three cubic equations with three unknowns are generated to form a polynomial system with 27 candidate solutions. In their latter work [14], they show that the non-relaxation polynomial system consists of three fifth-order equations and one cubic equation with four unknowns which yields 40 candidate solutions. In their work, the polynomial system is solved by using algebraic geometry and the optimal solution is picked from those candidates in a least-square sense. The polynomial system is still computationally expensive because of the construction of the 120×120 Macaulay matrix. Besides, it returns too many candidate solutions.

Actually, the algorithm proposed in [7] for the PnL problem is the counterpart of the approach presented in [5] for the PnP problem because both of them employing algebraic geometry to solve a polynomial system. In this work, inspired by the success of RPnP approach [6] for the PnP problem, we seek to establish its counterpart for the PnL problem although it is more challenging. Considering their sub problems, a fourth order polynomial can be generated [15,16] for the Perspective-3-Point problem while a higher order polynomial must be involved in for the P3L problem when 3D lines are in general configuration. These higher order polynomials increase the number of local minima of the cost function for the PnL problem. Furthermore, the mathematical expression of the perspective projection of lines is much different from that of points. These challenges are addressed in the following and the RPnL algorithm is proposed and validated.

3 The Perspective-3-Line Polynomial

Given a calibrated camera and 3 reference lines $L_i (i = 0, 1, 2)$ with their corresponding 2D projections on the image plane as l_i, an eighth order polynomial can be achieved by using the 3D/2D correspondences. The derivation of the P3L polynomial is as follows.

Let $L_i = (V_i, P_i)$ be the 3D line, in which V_i is the normalized vectors giving the direction of the line and P_i is a point on the line. Let $l_i = (s_i, e_i)$ be the projection of L_i on the image plane, in which s_i and e_i are the endpoints of l_i. For a given l_i, a projection plane Π_i can be determined which passes through the projection center O_c, l_i and L_i. The normal of Π_i is denoted as n_i (see Fig.1).

By selecting a line L_0, we can form a model coordinate framework $O_m X_m Y_m Z_m$ whose Z_m-axis aligns with V_0 and whose origin is located at the origin of the world frame. This can be achieved by choosing a rotation matrix R_w^m which rotates the direction vector in the world frame V_0^w to the model frame V_0^m and setting it as Z_m-axis, i.e. $V_0^m = R_w^m V_0^w = [0, 0, 1]^T$. One possible choice of R_w^m is $[\Lambda_1^T; \Lambda_2^T; V_0^T]$ where Λ_1 and Λ_2 are one pair of the orthogonal base of the null space formed by the linear system $V_0^T X = 0$. Then, apply the rotation

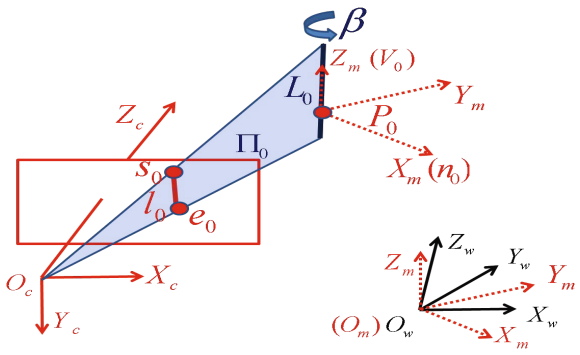


Fig. 1. The geometry of P3L: the projection of the 3D line on the image plane and the coordinate frameworks

matrix to the rest of lines so as to transform V_i^w in the world frame to the model coordinate frame as $V_i^m = R_w^m V_i^w$.

As L_0 lies on the plane Π_0, the line direction vector V_0 is perpendicular to the plane normal n_0. Hence, the rotation matrix from the camera to the model coordinate framework R_m^C must satisfy the constraint that $n_0^T \; R_m^C V_0^m = 0$. In order to enforce this constraint, R_m^C can be parameterized as

$$\begin{aligned} R_m^C &= R' \; Rot(X,\alpha)Rot(Z,\beta) \\ &= \begin{bmatrix} r'_1 & r'_4 & r'_7 \\ r'_2 & r'_5 & r'_8 \\ r'_3 & r'_6 & r'_9 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{1}$$

in which R' is an arbitrary orthogonal rotation matrix whose first column $[r'_1 \; r'_2 \; r'_3]^T$ equals to n_0, $Rot(X,\alpha)$ denotes a rotation around the X-axis, and $Rot(Z,\beta)$ denotes a rotation around the Z-axis. In this way, R_m^C can be determined by 2 unknown variables α and β. It's easy to verify that the constraint is fulfilled, i.e.

$$n_0^T \; R' \; Rot(X,\alpha)Rot(Z,\beta) \; V_0^m = [1,0,0] \; Rot(X,\alpha)Rot(Z,\beta) \; [0,0,1]^T \equiv 0 \tag{2}$$

By using the geometrical constraint that $V_i (i = 1, 2)$ should be perpendicular to the normal n_i of the plane Π_i, in the camera frame, we have the following two constraints

$$\begin{cases} n_1 \cdot V_1^C = n_1^T R_m^C V_1^m = 0 \\ n_2 \cdot V_2^C = n_2^T R_m^C V_2^m = 0 \end{cases} \tag{3}$$

Let $n_1^T R' = [nx'_1, ny'_1, nz'_1]$, $n_2^T R' = [nx'_2, ny'_2, nz'_2]$, $V_1^m = [vx_1, vy_1, vz_1]^T$ and $V_2^m = [vx_2, vy_2, vz_2]^T$. By substituting Eq.(1) into Eq.(3), we have

$$\begin{cases} \sigma_1 \cos\beta + \sigma_2 \sin\beta + \sigma_3 = 0 \\ \sigma_4 \cos\beta + \sigma_5 \sin\beta + \sigma_6 = 0 \end{cases} \tag{4}$$

in which

$$\begin{cases} \sigma_1 = vy_1\ ny'_1 \cos\alpha + vy_1\ nz'_1 \sin\alpha + vx_1\ nx'_1 \\ \sigma_2 = vx_1\ ny'_1 \cos\alpha + vx_1\ nz'_1 \sin\alpha - vy_1\ nx'_1 \\ \sigma_3 = vz_1\ nz'_1 \cos\alpha - \ vz_1\ ny'_1 \sin\alpha \\ \sigma_4 = vy_2\ ny'_2 \cos\alpha + vy_2\ nz'_2 \sin\alpha + vx_2\ nx'_2 \\ \sigma_5 = vx_2\ ny'_2 \cos\alpha + vx_2\ nz'_2 \sin\alpha - vy_2\ nx'_2 \\ \sigma_6 = vz_2\ nz'_2 \cos\alpha - \ vz_2\ ny'_2 \sin\alpha \end{cases}$$

By solving Eq.(4), we have

$$\cos\beta = \frac{\sigma_2\sigma_6 - \sigma_3\sigma_5}{\sigma_1\sigma_5 - \sigma_2\sigma_4}, \quad \sin\beta = \frac{\sigma_3\sigma_4 - \sigma_1\sigma_6}{\sigma_1\sigma_5 - \sigma_2\sigma_4}. \tag{5}$$

Substituting Eq.(5) into $cos^2\beta + sin^2\beta = 1$, we have

$$(\sigma_2\sigma_6 - \sigma_3\sigma_5)^2 + (\sigma_3\sigma_4 - \sigma_1\sigma_6)^2 = (\sigma_1\sigma_5 - \sigma_2\sigma_4)^2. \tag{6}$$

Substituting $\sin^2\alpha = 1 - \cos^2\alpha$ into Eq.(6) and rearranging the terms, we have

$$\sum_{k=0}^{4} u_k \cos^k\alpha + sin\alpha \sum_{k=0}^{3} v_k \cos^k\alpha = 0, \tag{7}$$

where u_k and v_k are coefficients which can be computed from nx'_i, ny'_i, nz'_i, vx_i, vy_i and vz_i for $i = 1, 2$. Taking the squares of both terms in Eq.(7) and letting $x = \cos\alpha$, an eighth order polynomial can be constructed as:

$$f(x) = \sum_{k=0}^{8} \delta_k x^k = 0, \tag{8}$$

where δ_k are computed from u_k and v_k as:

$$\begin{cases} \delta_0 = u_0^2 - v_0^2 \\ \delta_1 = 2(u_0u_1 - v_0v_1) \\ \delta_2 = u_1^2 + 2u_0u_2 + v_0^2 - v_1^2 - 2v_0v_2 \\ \delta_3 = 2(u_0u_3 + u_1u_2 + v_0v_1 - v_1v_2 - v_0v_3) \\ \delta_4 = u_2^2 + 2u_0u_4 + 2u_1u_3 + v_1^2 + 2v_0v_2 - v_2^2 - 2v_1v_3 \\ \delta_5 = 2(u_1u_4 + u_2u_3 + v_1v_2 + v_0v_3 - v_2v_3) \\ \delta_6 = u_3^2 + 2u_2u_4 + v_2^2 - v_3^2 + 2v_1v_3 \\ \delta_7 = 2(u_3u_4 + v_2v_3) \\ \delta_8 = u_4^2 + v_3^2 \end{cases} \tag{9}$$

Eq.(8) is called the P3L polynomial. In case of spatial lines in special configurations, such as orthogonal, parallel or intersection, a lower order of polynomial may be derived [17,18]. To the best of our knowledge, the P3L polynomial won't be lower than eighth order for three spatial lines in the general configuration. It's worth to point out that we decompose the over-all rotation from the world frame to the camera frame into a sequence of simple rotations by a different approach from [1,9] which is simpler than them despite the fact that all of them are derived by enforcing the orthogonal constraints and triangular constraints.

4 The Robust PnL Algorithm

4.1 Selecting a Rotation Axis to Form the Model Frame

Given n reference lines $L_i (i = 1, \cdots n)$ which are projected onto the normalized image plane as l_i, we firstly select a line L_{i0} from $\{L_i\}$ as a rotation axis V_0, based on which the model coordinate framework $O_m X_m Y_m Z_m$ is created (see Fig.1). The line with the longest projection length $|s_i e_i|$ is selected as the rotation axis because longer edges are less affected by noise on their endpoints. Furthermore, the line with the second longest projection length is selected as an auxiliary line L_{i1}, so as to construct a polynomial equation system together with the rest of lines. The auxiliary line is introduced to keep the complexity of the proposed RPnL algorithm be linear in the number of line correspondences.

4.2 Determination of the Rotation Axis

The line set $\{L_i\}$ can be divided into $n-2$ triplets $\{L_{i0}L_{i1}L_j | j = 1 \cdots n-2,\ j \neq i0,\ j \neq i1\}$. According to Eq.(8), each triplet yields an eighth order polynomial:

$$\begin{cases} f_1(x) = \sum_{k=0}^{8} \delta_{1k} x^k = 0 \\ f_2(x) = \sum_{k=0}^{8} \delta_{2k} x^k = 0 \\ \cdots \\ f_{n-2}(x) = \sum_{k=0}^{8} \delta_{(n-2)k} x^k = 0 \end{cases} \tag{10}$$

Instead of directly solving the nonlinear equation system (10) by the linearization technique which would lead to an inconsistent result from redundant equations, we explore the local minima of the system in terms of the least square residual. Firstly, a cost function F is defined as the square sum of the polynomials in Eq.(10), i.e. $F = \sum_{i=1}^{n-2} f_i^2(x)$. The minima of F can be determined by finding the roots of its derivative $F' = \sum_{i=1}^{n-2} f_i(x) f_i'(x) = 0$. F' is a 15-th order polynomial which can be easily solved by the eigenvalue method [19].

The 16-th order polynomial F has at most 8 minima. A brief proof is as follows: Assuming F has m stationary points, in which there are m_1 minima and m_2 maxima, $m_1 + m_2 \leq m$. As there exists at least one maximum between two minima, we have $m_1 - 1 \leq m_2$. As the stationary points of F are the real roots of F', we have $m \leq 15$. Therefore, $2m_1 - 1 \leq m_1 + m_2 \leq 15$, and we have $m_1 \leq 8$. □

In general, there are only a few real roots among the minima. These real roots are picked as the candidate solutions. As soon as x is solved, the rotation angle α around the X-axis in Eq.(1) can be calculated and the rotation axis V_0 in the camera frame can be determined, i.e. $V_0^C = R'\ Rot(X, \alpha)\ [0, 0, 1]^T$ .The unknown variables remained in the camera pose are the the rotation angle β around the axis V_0 and the translation vector $\mathbf{T}$.

4.3 Solving the Rotation Angle and the Translation Vector

When the rotation axis V_0 (i.e. Z_m axis of the model coordinate frame) is determined, from Eq.1, the rotation matrix from the camera to the model coordinate framework R_m^C can be expressed as:

$$R_m^C = \bar{R}\ Rot(Z,\beta) = \begin{bmatrix} \bar{r}_1 & \bar{r}_4 & \bar{r}_7 \\ \bar{r}_2 & \bar{r}_5 & \bar{r}_8 \\ \bar{r}_3 & \bar{r}_6 & \bar{r}_9 \end{bmatrix} \begin{bmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{11}$$

in which $\bar{R} = R'\ Rot(X,\alpha)$, $c = \cos\beta$ and $s = \sin\beta$. According to the projection from 3D lines to 2D lines in the normalized image plane [20], we have:

$$n_i = R_m^C\left((P_i^m - \mathbf{T}) \times V_i^m\right) = (R_m^C P_i^m - \bar{\mathbf{T}}) \times (R_m^C V_i^m) \tag{12}$$

where P_i^m is a point on the line L_i^m with direction V_i^m in the model coordinate frame. The rotated translation vector $\bar{\mathbf{T}} = R_m^C\mathbf{T} = [\bar{t}_x, \bar{t}_y, \bar{t}_z]^T$ and '$\times$' means the cross product of two vectors. So we have

$$n_i^T R_m^C V_i^m = 0, \quad n_i^T(R_m^C P_i^m - \bar{\mathbf{T}}) = 0; \quad i = 1, 2, \cdots, n. \tag{13}$$

For n lines, by substituting Eq.11 into Eq.13 and stacking these constraints, we get $2n$ homogenous linear equations with parameter vector $[c, s, \bar{t}_x, \bar{t}_y, \bar{t}_z, 1]^T$. Letting $n_i = [nx_i, ny_i, nz_i]^T$, $(n_i^T\bar{R}) = [n\bar{x}_i, n\bar{y}_i, n\bar{z}_i]$, $P_i^m = [px_i, py_i, pz_i]^T$ and $V_i^m = [vx_i, vy_i, vz_i]^T$, we have:

$$\begin{bmatrix} & & \cdots & & & \\ n\bar{x}_i vx_i + n\bar{y}_i vy_i & n\bar{y}_i vx_i - n\bar{x}_i vy_i & 0 & 0 & 0 & n\bar{z}_i vz_i \\ n\bar{x}_i px_i + n\bar{y}_i py_i & n\bar{y}_i px_i - n\bar{x}_i py_i & -nx_i & -ny_i & -nz_i & n\bar{z}_i pz_i \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} c \\ s \\ \bar{t}_x \\ \bar{t}_y \\ \bar{t}_z \\ 1 \end{bmatrix} = 0 \tag{14}$$

The rotation angle β and the rotated translation vector $\bar{\mathbf{T}}$ can be estimated by solving this linear system in Eq.14 with SVD. Then, the rotation matrix from the camera frame to the world frame is computed as $R_w^C = R_m^C R_w^m$ and the translation vector is computed as $\mathbf{T} = (R_m^C)^T\ \bar{\mathbf{T}}$.

4.4 Determination of the Camera Pose

Since the solution of $[c, s]$ is estimated from the linear system, it may not satisfy the triangular constraint $c^2 + s^2 = 1$ due to noise in the data. R_m^C computed from Eq.11 should be normalized to a rotation matrix, so does the estimation of R_w^C. This is achieved by a standard 3D alignment scheme[8]. First, we translate the points on lines from the world frame $\{P_i^w\}$ to the camera frame $\{P_i^C\}$ by using the un-normalized estimation of R_w^C and $\mathbf{T}$, then project these points onto the interpretation plane of lines $\{\bar{P}_i^C\}$ as follows:

$$P_i^C = R_w^C(P_i^w - \mathbf{T}), \quad \bar{P}_i^C = P_i^C - (P_i^C \cdot n_i)\ n_i; \quad i = 1, \cdots, n \tag{15}$$

Finally, we retrieve the normalized camera pose by aligning two point sets $\{P_i^w\}$ and $\{\bar{P}_i^C\}$. In order to improve the alignment accuracy, the information of line direction can also be employed. This is achieved by increasing the number of points in the two sets. To this end, for each 3D line, we add another point $\hat{P}_i^w$ into the world point set. In our implementation, $\hat{P}_i^w$ is the closest point on the line L_i to the origin of the world coordinate which can be computed from (P_i^w, V_i^w) as: $\hat{P}_i^w = P_i^w - (P_i^w \cdot V_i^w)\, V_i^w$. If P_i^w and $\hat{P}_i^w$ are coincident, then we shift $\hat{P}_i^w$ along the line direction. After that, we translate these additional world points into the camera frame and project them onto the interpretation plane of lines by Eq.15 to get $\{\hat{\bar{P}}_i^C\}$. The normalized camera pose is estimated from the alignment of the enlarged two point sets $\{P_i^w, \hat{P}_i^w\}$ and $\{\bar{P}_i^C, \hat{\bar{P}}_i^C\}$.

After the pose normalization, we get a few candidate pose solutions which correspond to the minima of the polynomial system in Sec.4.2. For each candidate solution, we first evaluate it by the orthogonal error E_{er} which is defined as

$$E_{er} = \sum_{i=1}^{n} \left(n_i^T \; R_w^C \; V_i\right)^2 \tag{16}$$

We exclude those solutions with large orthogonal errors. From the remaining candidates, we select the one resulting in smallest re-projection residual as the optimum. The re-projection residual E_{re} is defined as the difference between the observed image line and the re-projected image line as follow [21]:

$$E_{re} = \sum_{i=1}^{n} \int_0^{\ell_i} h_i^2(s) d(s) = \sum_{i=1}^{n} \frac{\ell_i}{3} (h_{is}^2 + h_{is} h_{ie} + h_{ie}^2) \tag{17}$$

where ℓ_i is the length of image line l_i, h_{is} and h_{ie} are the distances of observed line endpoints to the re-projected line. Sometimes, the re-projection residuals of two pose solutions E_{re}^1 and E_{re}^2 are very close. This happens when one of the solutions transforms the world scene in front of the camera while the other solution transforms the world scene behind the camera. In this case, we choose the former one as the final solution.

5 Results

5.1 Experiments with Synthetic Data

Given a virtual perspective camera with image size 640×480 pixels and focal length 800 pixels, the 3D reference lines are randomly generated in the camera coordinate frame. Different levels of Gaussian noise are added to the projected image lines, and for each noise level 2000 test data sets are generated.

The following methods are compared:

(1) **RPnL**, the proposed algorithm. (2) **RPnL++**, the proposed algorithm plus a R_and_T optimization scheme [11]. (3) **Mirzaei**, a non-iterative solution by Mirzaei and Roumeliotis [14]. It is one of the most accurate solutions

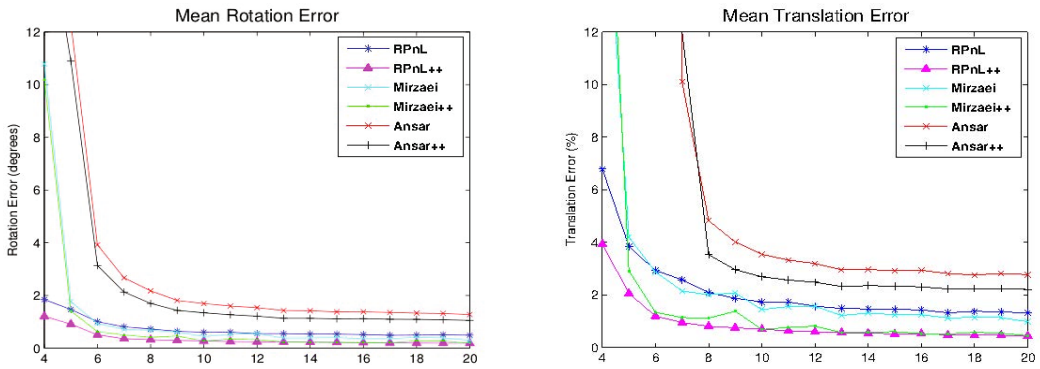


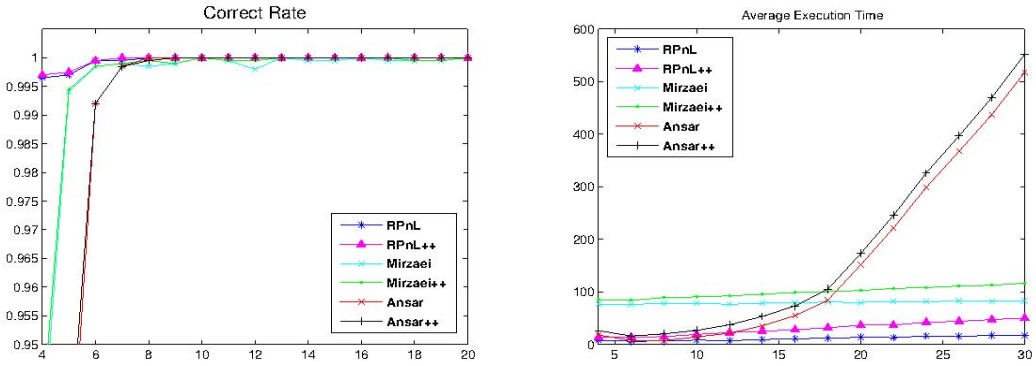
Fig. 2. The mean rotation and translation errors of the compared methods are plotted as a function of the number of the lines. The noise level $\sigma = 5$ pixels.

for PnL so far. The local minima are retrieved by solving a 27×27 multiplication matrix and the global optimum is picked by evaluating the orthogonal errors defined as in Eq.16. The approach is not very efficient because of the high dimensional Macaulay Matrix. (4) **Mirzaei++**, Mirzaei plus a R_and_T optimization scheme. (5) **Ansar**, a non-iterative solution by Ansar and Danilidis [13] which converts the polynomial system to a set of linear systems by re-parameterizing the nonlinear terms with new variables. (6) **Ansar++**, Ansar plus a R_and_T optimization scheme. In our implementation, **RPnL++**, **Mirzaei++** and **Ansar++** take the pose solutions which are estimated by **RPnL**, **Mirzaei** and **Ansar** as the initial value respectively, then use the same R_and_T algorithm to iteratively search the optimum.

The following evaluations are conducted:

A. The accuracy: As can be seen in Fig. 2, RPnL is as accurate as the best state-of-the-art solution. When $n \leq 5$, the mean rotation error and mean translation error of RPnL are significantly better than that of existing non-iterative solutions. When $n > 5$, the accuracy of RPnL is similar to that of Mirzaei. The pose estimation error of these non-iterative solutions can be further reduced by employing an iterative optimization scheme such as R_and_T [11]. After applying the iterative optimization, Fig. 2 illustrates that the accuracy of RPnL++ is better than other methods for both $n \leq 5$ and $n > 5$.

B. The robustness: Robustness is one of the most significant advantages of RPnL comparing to other methods, which can be seen in Fig. 3.(a). The correct rates of the compared methods are plotted, which are the ratio of the number of correct solutions over 2000 tests (a solution is deemed as a correct result when the deviation of the estimated rotation from the ground truth is less than 30 degrees). When $n \leq 5$, the correct rate of our solution is noticeably better than the other methods. When $n > 5$, RPnL is still slightly better. The reason may lies in two facts that: RPnL chooses a line with the longest projection in the image plane as rotation axis which is less sensitive to the image noise and RPnL has lower complexity than Mirzaei which makes RPnL more numerically stable. For the robustness of Ansar, Fig. 3.(a) shows that when few reference lines are available, the algorithm is very sensitive to the image noise. When the number



(a) Correct Rate (b) Computational Time

Fig. 3. The correct rate (a) and the average computational time (b) of the compared methods are plotted as a function of the number of the lines. The noise level $\sigma = 5$ pixels.

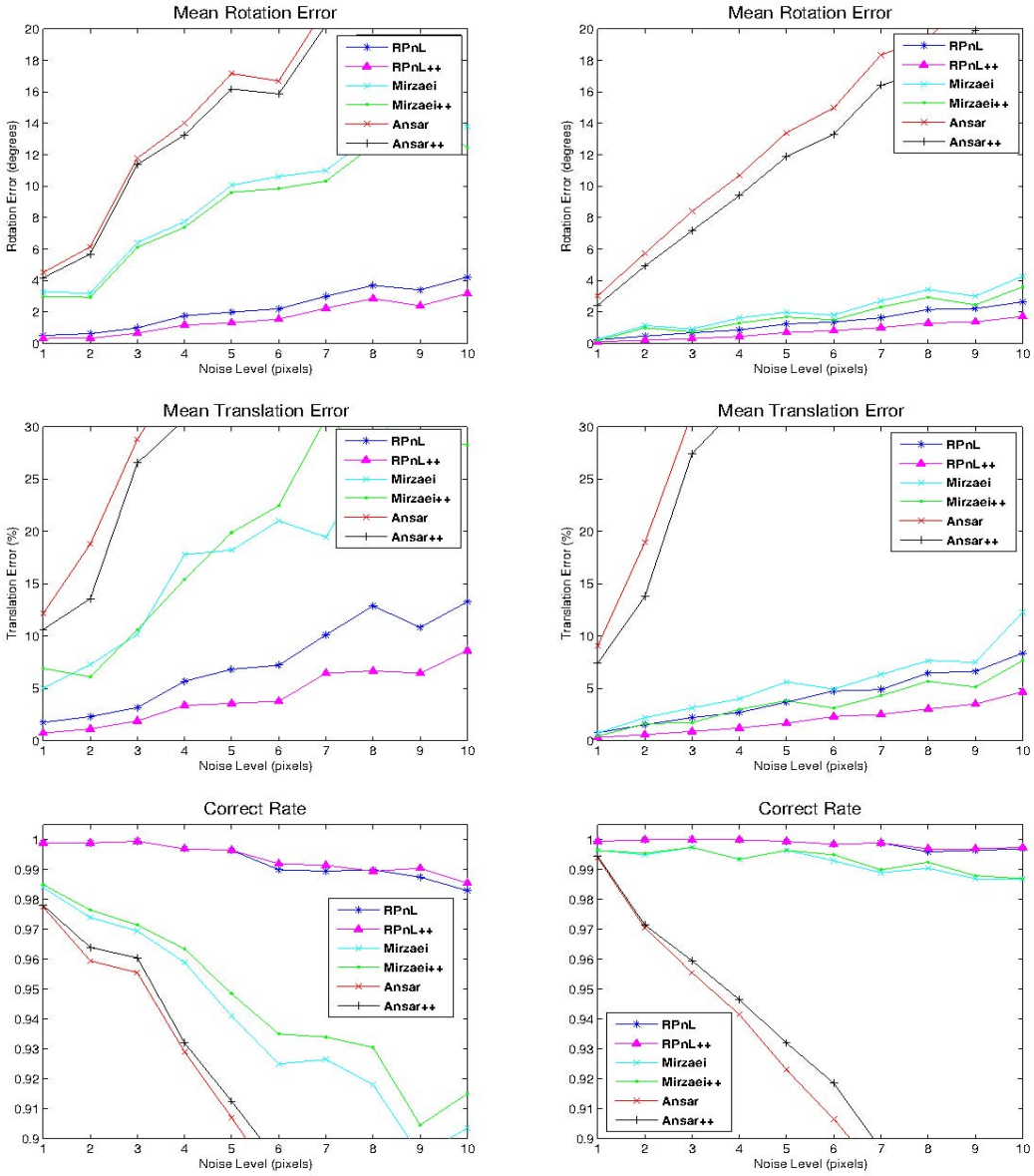
of reference lines are large, the solution of Ansar turns to be stable because the number of linear equations are much more than the lifting variables.

C. The efficiency: RPnL is highly efficient and its computational time grows linearly with n. As can be seen in Fig. 3.(b), the average executive times are plotted as a function of the number of lines from 4 to 30. The method was implemented in MATLAB and 2000 runs are performed. The efficiency of RPnL is significantly better than the other methods, and its computational time takes only a fraction of that of Mirzaei which is also linear in the number of correspondences. Fig. 3.(b) clearly shows that the computationally complexity of Ansar is $O(n^2)$.

D. The small line set: RPnL can accurately deal with the small line set. As can be seen in Fig. 4, RPnL can achieve higher accuracy than existing methods for small line set in the presence of image noise. The correct rates shown in the Fig. 4 also demonstrate that the existing methods are very sensitive to noise when $n = 4$ or $n = 5$ while RPnL suffers less performance decrease from the increase of the noise level. Here the noise level is measured from the noise added to the endpoints of lines in pixel. The correct rates of RPnL++, Mirzaei++ and Ansar++ reported in Fig. 4 show that in the absence of a good initialization, the iterative algorithms may converge to a local minimum. The performance of the iterative algorithms mainly depend on the performance of the non-iterative algorithms which are employed to initialize them. However, they can always slightly improve the accuracy and robustness of their initialization approaches in the cost of losing some computational efficiency.

5.2 Experiments with Real Images

In order to validate the proposed algorithm in real situations, we apply RPnL on two image sequences with known 3D line models. For each image, we extract lines by LSD line detector [22], then manually establish the correspondences between



(a) $n = 4$ (b) $n = 5$

Fig. 4. The mean rotation errors, the mean translation errors and the correct rate of the compared methods for $n = 4$ (first column) and $n = 5$ (second column) as a function of the image noise level which varies from 1 to 10 pixels

the image lines and 3D models (Depending on the applications, the 2D/3D line correspondences could also be established automatically). The camera pose is estimated by RPnL algorithm. In order to demonstrate the accuracy of the results, we reproject the 3D line model into the image by using the estimated camera pose. Fig.5 shows the 3D models and some sampled results from the image sequences (The complete results for images in these two sequences are reported in the supplementary video). As shown in the left side of top two rows of Fig.5, the first 3D model is a cube with some line segments on the

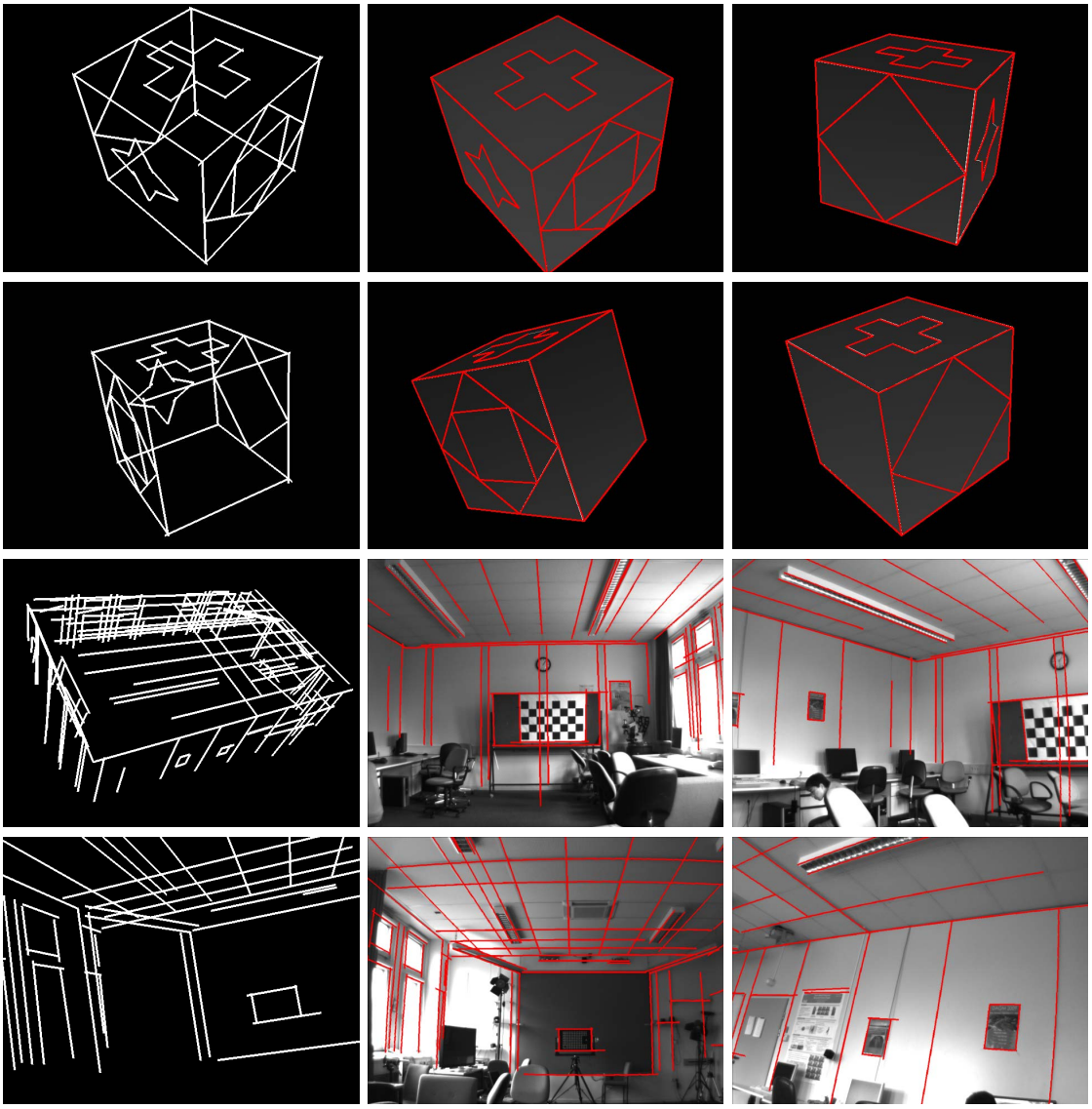


Fig. 5. Illustration the results of RPnL for real images. The first column shows the 3D line models and the last two columns show the sampled images with projection of lines by using the estimated camera pose from RPnL.

facades whose 3D positions are perfectly known. In this situation, the noise is only from the image measurement. The results in the top two rows show that the reprojected lines are perfectly aligned with the image lines which indicates that the camera pose is well estimated. Since the perfect 3D model may be not available in practice, we conduct another experiment on the second image sequence. As shown in the left side of bottom two rows of Fig.5, the second 3D model is a laboratory which is generated from images of our office by a structure-and-motion algorithm. The 3D position of line segments (especially their endpoints) are not perfectly reconstructed. In this case, the noise exist both in the 3D model and in the image measurement. The RPnL can still successfully

estimate the camera pose as illustrated in the bottom row of Fig.5. Notice that the reprojected lines and the image lines are well aligned despite of their endpoint differences, e.g. the border of the poster on the wall and the border of the check board. The results well demonstrate the robustness and accuracy of RPnL.

6 Conclusions

The perspective-n-line problem is a classic topic, however, many issues in this field remain open: (1) The small line sets ($3 < n \leq 5$) are sensitive to the noise; (2) The computational complexity to discover the global optimum is expensive; (3) It is hard to find a solution be both accurate and efficient.

To address these issues, a counterpart of RPnP is proposed in this paper which is named as RPnL. In the framework of RPnL, lines are separated into triplets by selecting a rotation axis, then a sixteenth order cost function is built from a set of P3L polynomials. The optimum solution of the pose is retrieved from the local minima. Experiment results show that RPnL, although of much lower computational complexity, is as accurate as the state-of-the-art algorithms. For small number of line correspondences, RPnL achieve higher accuracy than existing non-iterative methods. The implementation of RPnL and RPnL++ are available on our website[1].

RPnL is suitable for applications that need to handle both small and large number of line correspondences, such as the line feature based object tracking or camera localization in computer vision and augmented reality. Our future work will try to apply it to these applications.

Acknowledgement. Lilian Zhang was supported by China Scholarship Council (No.2009611008). Special thanks to the members of the Kiel Multimedia Information Processing group for their kind help.

References

1. Chen, H.: Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions. TPAMI 13, 530–541 (1991)
2. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)
3. Zhou, F., Duh, H.L., Billinghurst, M.: Trends in augmented reality tracking, interaction and display: A review of ten years of ismar. In: ISMAR, pp. 193–202 (2008)
4. Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o(n) solution to the pnp problem. IJCV 81, 155–166 (2009)
5. Hesch, J., Roumeliotis, S.I.: A direct least-squares (dls) method for pnp. In: ICCV, pp. 383–390 (2011)
6. Li, S., Xu, C., Xie, M.: A robust o(n) solution to the perspective-n-point problem. TPAMI, 1–8 (2012)

[1] `http://www.mip.informatik.uni-kiel.de/tiki-index.php?page=Lilian+Zhang`

7. Mirzaei, F.M., Roumeliotis, S.I.: Globally optimal pose estimation from line correspondences. In: ICRA, pp. 5581–5588 (2011)
8. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. TPAMI 13, 376–380 (1991)
9. Dhome, M., Richetin, M., Lapreste, J.T., Rives, G.: Determination of the attitude of 3d objects from a single perspective view. TPAMI 11, 1265–1278 (1989)
10. Liu, Y., Huang, T., Faugeras, O.: Determination of camera location from 2-d to 3-d line and point correspondences. TPAMI 12, 28–37 (1990)
11. Kumar, R., Hanson, A.R.: Robust methods for estimating pose and a sensitivity analysis. CVGIP: Image Understanding 60, 313–342 (1994)
12. Christy, S., Horaud, R.: Iterative pose computation from line correspondences. CVIU 73, 137–144 (1999)
13. Ansar, A., Daniilidis, K.: Linear pose estimation from points or lines. TPAMI 25, 282–296 (2003)
14. Mirzaei, F.M., Roumeliotis, S.I.: Optimal estimation of vanishing points in a manhattan world. In: ICCV, pp. 2454–2461 (2011)
15. Li, S., Xu, C.: A stable direct solution of perspective-three-point problem. IJPRAI 25, 627–642 (2011)
16. Kneip, L., Scaramuzza, D., Siegwart, R.: A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation, pp. 2969–2976 (2011)
17. Qin, L.J., Zhu, F.: A new method for pose estimation from line correspondences. Acta Automatica Sinica 34, 130–134 (2008)
18. Liu, C., Zhu, F., Ou, J., Yu, Y.: Z-shaped perspective-three-line problem's unique solution conditions. In: International Workshop on Intelligent Networks and Intelligent Systems, pp. 132–135 (2010)
19. Press, W., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical recipes: the art of scientific computing. Cambridge Univ. Press (2007)
20. Zhang, L., Koch, R.: Hand-held monocular slam based on line segments. In: IMVIP, pp. 8–15 (2011)
21. Taylor, C., Kriegman, D.: Structure and motion from line segments in multiple images. TPAMI 17, 1021–1032 (1995)
22. von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: Lsd: A fast line segment detector with a false detection control. TPAMI 32, 722–732 (2010)