A Robust and Efficient Algorithm for the PnL Problem Using Algebraic Distance to Approximate the Reprojection Distance

Lipu Zhou, Yi Yang, Montiel Abello, Michael Kaess
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
{lipuz,yiy4,mabello,kaess}@andrew.cmu.edu

Abstract

This paper proposes a novel algorithm to solve the pose estimation problem from 2D/3D line correspondences, known as the Perspective-n-Line (PnL) problem. It is widely known that minimizing the geometric distance generally results in more accurate results than minimizing an algebraic distance. However, the rational form of the reprojection distance of the line yields a complicated cost function, which makes solving the first-order optimality conditions infeasible. Furthermore, iterative algorithms based on the reprojection distance are time-consuming for a large-scale problem. In contrast to previous works which minimize a cost function based on an algebraic distance that may not approximate the reprojection distance of the line, we design two simple algebraic distances to gradually approximate the reprojection distance. This speeds up the computation, and maintains the robustness of the geometric distance. The two algebraic distances result in two polynomial cost functions, which can be efficiently solved. We directly solve the first-order optimality conditions of the first problem with a novel hidden variable method. This algorithm makes use of the specific structure of the resulting polynomial system, therefore it is more stable than the general Gröbner basis polynomial solver. Then, we minimize the second polynomial cost function by the damped Newton iteration, starting from the solution of the first cost function. Experimental results show that the first step of our algorithm is already superior to the state-of-the-art algorithms in terms of accuracy and applicability, and faster than the algorithms based on Gröbner basis polynomial solver. The second step yields comparable results to the results from minimizing the reprojection distance, but is much more efficient. For speed, our algorithm is applicable to real-time applications.

Introduction

The Perspective-n-Line (PnL) problem is to calculate the rotation and the translation of a camera from N 2D/3D line correspondences. It has broad applications in robotics and 3D vision, such as Structure from Motion (SfM) (Micusik and Wildenauer 2017), and Simultaneous Localization and Mapping (SLAM) (Zhang and Koch 2014), and Augmented Reality (AR) (Zhou, Duh, and Billinghurst 2008). Because of its importance, plenty of algorithms have been proposed to address this problem in the literature. However, recent work (Přibyl, Zemčík, and Čadík 2017) has shown that none of the existing methods universally outperforms the others. Algorithms based on Direct Linear Transformation (DLT) (Přibyl, Zemčík, and Čadík 2017) are fast, but are not stable or even not feasible when the number of lines N is small. Nonlinearly formulated algorithms may become very computationally demanding when N gets large, such as(Mirzaei and Roumeliotis 2011b; Xu et al. 2017; Ansar and Daniilidis 2003). In addition, many PnL algorithms are not applicable to the planar configuration (*i.e.* all the 3D lines are located on a plane). This paper aims to achieve globally optimal solution for any number and configuration of lines with real-time speed.

Minimizing a geometric distance is known to lead to more accurate results than an algebraic distance (Hartley and Zisserman 2003). However, the reprojection distance of the PnL problem, *i.e.* the distance between the projection of a 3D line and the end points of the corresponding 2D line in the image, results in a non-convex rational cost function. Thus, it is intractable to directly solve its first-order optimality conditions. Additionally, the iterative algorithm requires high quality initialization to converge to the globally minimal solution, and will become time-consuming when the number of lines grows large. Therefore, different algebraic distances are proposed in the literature to simplify the computation. For example, Xu *et al.* (Xu et al. 2017) used the residual of the equation from a minimal solution as the cost function. But those algebraic distances may not approximate the reprojection distance. This may lead to a suboptimal solution. The central idea of this paper is to approximate the reprojection distance with simpler functions. We design two algebraic distances to achieve this goal.

The first algebraic distance derives from the distance between a 2D line and the projection of two points on the corresponding 3D line. We construct a fourth-order polynomial cost function for the rotation. Gröbner basis method (Byröd, Josephson, and Åström 2009) is generally used in the previous works (Mirzaei and Roumeliotis 2011a; 2011b; Zheng et al. 2013; Vakhitov, Funke, and Moreno-Noguer 2016) to solve the first-order optimality conditions of a polynomial cost function, but this method may have numeric problems which are difficult to address. We solve the first-order optimality conditions by a novel hidden variable method (Cox, Little, and O'shea 2006; Gelfand, Kapranov,

and Zelevinsky 2008). This method improves the stability of the polynomial solver by taking advantage of the special structure of the resulting polynomial system. It does not include numerically unstable functions, such as the matrix inverse involved in the Gröbner basis method.

The second algebraic distance formulation involves fixing the denominator of the reprojection distance, which results in a polynomial cost function.The reprojection distance is approximated around the point where the denominator is fixed. We apply the damped Newton iteration to minimize the second problem, as it is efficient to calculate the Hessian matrix and gradient of a polynomial function.

The simulation and experimental results show that the first step of our algorithm already outperforms the state-of-the-art methods in terms of accuracy and applicability, and is faster than the algorithms based on Gröbner basis polynomial solver (Mirzaei and Roumeliotis 2011b; Vakhitov, Funke, and Moreno-Noguer 2016). The result from the second cost function is comparable to the result from the iterative algorithm using the reprojection cost function, but is much faster. Lastly, our algorithm is fast enough for real-time applications.

Related Work

The PnL problem has been widely studied in the literature. At least three 2D/3D line correspondences are required to compute the pose. The minimal problem of PnL is called the P3L problem. Some algorithms have been proposed to address the P3L problem (Dhome et al. 1989; Chen 1990; Xu et al. 2017). They showed that there may exist at most 8 solutions to the P3L problem. Recently, Xu *et al.* (Xu et al. 2017) systematically analyzed the relationship between the number of solutions and the configuration of the three lines. The P3L algorithm is generally used in the RANSAC framework (Fischler and Bolles 1987) to eliminate outliers.

As mentioned above, the geometric distance of the PnL problem is complicated. Iterative algorithms give a straight forward way to address the minimization problem. In an early work (Liu, Huang, and Faugeras 1990), rotation and translation were calculated separately. They calculated the rotation by iteratively linearizing the equation. In (Kumar and Hanson 1994), the rotation and translation were jointly optimized. David *et al.* (David et al. 2003) simultaneously estimated the pose and the correspondence between 2D and 3D lines in an iterative manner. Recent work (Zhang et al. 2016) presented a noise model to describe the probabilistic relationship between the 3D line and its image. Using this model, they derived a maximum likelihood estimation of the camera pose. Iterative methods may converge to a local minimum. In addition, iterative algorithms are time-consuming for a large-scale problem.

To reduce the computational complexity, some works seek to linearize the reprojection distance or an algebraic distance. The DLT method (Hartley and Zisserman 2003) gives a fast way to solve the PnL problem. It needs at least 6 lines. Recently, Přibyl *et al.* (Bronislav Přibyl 2015) exploit the relationship between the projection of the Plücker coordinates and the 2D line to construct linear equations. This algorithm needs at least 9 lines. In their later work (Přibyl, Zemčík, and Čadík 2017), they combined the traditional DLT method (Hartley and Zisserman 2003) and their Plücker coordinates based DLT method (Bronislav Přibyl 2015) to further improve the accuracy, and reduced the minimum number of line correspondences to 5. Ansar and Daniilidis (Ansar and Daniilidis 2003) proposed an algorithm to linearize a quadratic equation system to a linear system. The $O\left(n^2\right)$ computational complexity of this algorithm makes it impracticable for a large-scale problem. As these methods ignore the nonlinear part of the constraints, they are not accurate or even feasible when N is small.

To solve the problem of linearization, nonlinear algebraic distances are designed to replace the reprojection distance in some recent works. Xu *et al.* (Xu et al. 2017) used the minimal solution to construct a 16th order polynomial cost function. This algorithm is accurate and efficient when N is small, but becomes less accurate and computationally demanding as N grows large as demonstrated in (Přibyl, Zemčík, and Čadík 2017). Mizaei *et al.* (Mirzaei and Roumeliotis 2011a; 2011b) decoupled the estimation of rotation and translation. They directly solve the first-order optimality conditions of the rotation matrix to obtain all stationary points. Some works extended the Perspective-n-Point (PnP) algorithms to the PnL problem. Vakhitov *et al.* (Vakhitov, Funke, and Moreno-Noguer 2016) and Xu *et al.* (Xu et al. 2017) adapted the EPnP algorithm (Lepetit, Moreno-Noguer, and Fua 2008) to the PnL problem. Vakhitov *et al.* (Vakhitov, Funke, and Moreno-Noguer 2016) showed that OPnP algorithm (Zheng et al. 2013) can be used to solve the PnL problem. As these algebraic distances do not approximate the geometric distance, this may result in suboptimal results. This paper aims to approximate the reprojection function to efficiently obtain the globally optimal solution.

Problem Formulation

Throughout this paper we use italic, boldfaced lowercase and boldfaced uppercase letters to represent scalars, vectors and matrices, respectively. To simplify the notation, we use the normalized pixel coordinates (Hartley and Zisserman 2003), *i.e.* multiplying the homogeneous pixel coordinates by the inverse of the camera intrinsic parameter matrix $\mathbf{K}$ before solving the PnL problem.

The PnL problem is to estimate the pose of a camera, including rotation $\mathbf{R}$ and translation $\mathbf{t}$ relative to a world frame, by $N\,(N \geq 3)$ 2D/3D lines correspondences. The minimal case is $N = 3$, called the P3L problem, which has at most 8 solutions (Xu et al. 2017). For $N \geq 4$, there generally only exists one solution.

We use the Cayley–Gibbs–Rodriguez (CGR) parametrization (Mirzaei and Roumeliotis 2011b) to represent $\mathbf{R}$ as

$$\mathbf{R} = \frac{\bar{\mathbf{R}}}{1 + \mathbf{s}^T\mathbf{s}}, \bar{\mathbf{R}} = \left(\left(1 - \mathbf{s}^T\mathbf{s}\right)\mathbf{I}_3 + 2[\mathbf{s}]_\times + 2\mathbf{s}\mathbf{s}^T\right), \quad (1)$$

where $\mathbf{s} = [s_1; s_2; s_3]$ is a 3-dimensional vector and $[\mathbf{s}]_\times = \begin{bmatrix} 0 & -s_3 & s_2 \\ s_3 & 0 & -s_1 \\ -s_2 & s_1 & 0 \end{bmatrix}$. As other minimal rotation param-

eterizations, the CGR parameterization also has a singular case, that is when the rotation angle equals to π. This can be easily solved by rotating the measurements to an intermediate frame, then rotating the solutions back to the original frame, as mentioned by (Mirzaei and Roumeliotis 2011a). In practice, the RANSAC algorithm is generally used before the least-squares algorithm. The solution from the RANSAC algorithm can be used to rotate the measure to a non-singular case. The advantage of the CGR parameterization is that it does not require additional constraints, such as the unity norm condition of the quaternion parameterization. This leads to a simple formulation.

We adopt the Plücker coordinates (Přibyl, Zemčík, and Čadík 2017; Hartley and Zisserman 2003) to represent the 3D line. For the ith 3D line $\mathbf{L}_i$, we denote its Plücker coordinates as a six-dimensional vector $\mathbf{L}_i = [\mathbf{d}_i; \mathbf{m}_i]$. $\mathbf{d}_i$ and $\mathbf{m}_i$ are the direction vector and moment vector of L_i, respectively, and they satisfy $\mathbf{d_i} \cdot \mathbf{m_i} = 0$, where $\cdot$ represents the dot product. Let $\mathbf{c} = -\mathbf{R}^T\mathbf{t}$. Denote the projection of $\mathbf{L}_i$ to the image plane as $\mathbf{l}_i$, as demonstrated in Figure 1. The relationship between $\mathbf{L}_i$ and $\mathbf{l}_i$ is:

$$\mathbf{l}_i \sim \left[\mathbf{R}, -\mathbf{R}[\mathbf{c}]_\times\right] \mathbf{L}_i, \tag{2}$$

where $\sim$ represents the two homogeneous entities are equal up to scale. Let $\mathbf{l}_i = \left[l_i^1, l_i^2, l_i^3\right]$. Assume the ideal 2D line $\mathbf{l}_i$ corresponding to $\mathbf{L}_i$ is detected by a line detection algorithm, such as (Von Gioi et al. 2010), as $\hat{\mathbf{l}}_i$. Because of noise, for example motion blur and rounding error, the ideal projection $\mathbf{l}_i$ and the observation $\hat{\mathbf{l}}_i$ are different. Suppose $\hat{\mathbf{p}}_{ij} = [\hat{x}_{ij}; \hat{y}_{ij}; 1],\ j = 1, 2$ are the homogeneous coordinates of the two endpoints of $\hat{\mathbf{l}}_i$. $\hat{\mathbf{p}}_{ij}$ does not generally lie on $\mathbf{l}_i$. The signed reprojection distance for the ith 2D/3D line correspondence is:

$$e_{ij}^{rep} = \frac{\hat{\mathbf{p}}_{ij} \cdot \mathbf{l}_i}{\sqrt{\left(l_i^1\right)^2 + \left(l_i^2\right)^2}} \tag{3}$$

The cost function based on the reprojection distance is:

$$C_{rep} = \frac{1}{2} \sum_i \sum_j \left(e_{ij}^{rep}\right)^2 \tag{4}$$

According to the CGR parameterization of $\mathbf{R}$ in (1), the denominator and the numerator of $\left(e_{ij}^{erp}\right)^2$ are both 6th degree polynomial functions involving $\mathbf{s}$ and $\mathbf{c}$, respectively. As the denominator of e_{ij}^{rep} is different for each 2D/3D line correspondence, the degrees of the polynomials in the denominator and the numerator of the summation C_{rep} will quickly increase as the number of measurements increases. This makes calculating the first-order optimality conditions of (4) intractable. This paper seeks to solve this problem.

Optimal Solution from Algebraic Distance

The central idea of this paper is to approximate the reprojection distance e_{ij}^{rep} by simpler functions, whose gradient and Hessian matrix can be easily computed. We use two algebraic distances to achieve this.

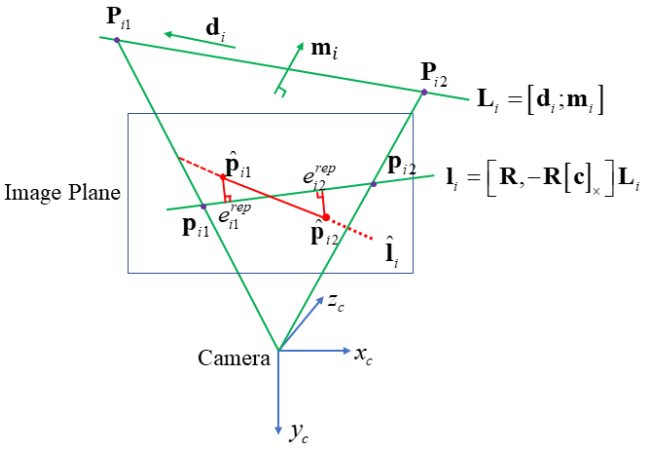


Figure 1: Schematic of the reprojection distance.

First Algebraic Distance

Assume $\mathbf{P}_{ij}, j = 1, 2$ are two 3D points on $\mathbf{L_i}$. Denote the homogeneous coordinates of the projection of $\mathbf{P}_{ij}$ in the image plane as $\mathbf{p}_{ij}$. Using the normalized pixel, we have $\mathbf{p}_{ij} \sim \mathbf{R}\mathbf{P}_i^j + \mathbf{t}$. Theoretically, $\mathbf{p}_{ij}$ should be on the 2D line $\hat{\mathbf{l}}_i$, thus we have

$$\hat{\mathbf{l}}_i \cdot (\mathbf{R}\mathbf{P}_{ij} + \mathbf{t}) = 0. \tag{5}$$

In contrast to (3), we can decouple $\mathbf{R}$ and $\mathbf{t}$ from (5). As $\mathbf{t}$ is unconstrained and linear in (5), we first consider the solution of $\mathbf{t}$. Given $\mathbf{R}$, we derive a linear equation of $\mathbf{t}$ from (5) as:

$$\hat{\mathbf{l}}_i \cdot \mathbf{t} = -\hat{\mathbf{l}}_i \cdot \mathbf{R}\mathbf{P}_{ij}. \tag{6}$$

This is a linear unconstrained least-squares problem. Solving for $\mathbf{t}$, we obtain a closed-form solution of $\mathbf{t}$ as

$$\mathbf{t} = -\left(\mathbf{L}^T\mathbf{L}\right)^{-1}\mathbf{L}^T\mathbf{b}\left(\mathbf{R}\right), \tag{7}$$

where $\mathbf{L} = \left[\hat{\mathbf{l}}_1, \hat{\mathbf{l}}_1, \cdots, \hat{\mathbf{l}}_N, \hat{\mathbf{l}}_N\right]^T$, $\mathbf{b}\left(\mathbf{R}\right) = \left[\hat{\mathbf{l}}_1^T\mathbf{R}\mathbf{P}_{11}, \hat{\mathbf{l}}_1^T\mathbf{R}\mathbf{P}_{12}, \cdots, \mathbf{l}_N^T\mathbf{R}\mathbf{P}_{N1}, \mathbf{l}_N^T\mathbf{R}\mathbf{P}_{N2}\right]^T$. Substituting the CGR parametrization of $\mathbf{R}$ in (1) into (7), we have

$$\mathbf{t} = -\frac{\left(\mathbf{L}^T\mathbf{L}\right)^{-1}\mathbf{L}^T\mathbf{b}\left(\bar{\mathbf{R}}\right)}{1 + \mathbf{s}^T\mathbf{s}} \tag{8}$$

Then substituting (1) and (8) into (5), we obtain

$$\frac{\hat{\mathbf{l}}_i \cdot \bar{\mathbf{R}}\mathbf{P}_{ij} + \hat{\mathbf{l}}_i \cdot \left(\mathbf{L}^T\mathbf{L}\right)^{-1}\mathbf{L}^T\mathbf{b}\left(\bar{\mathbf{R}}\right)}{1 + \mathbf{s}^T\mathbf{s}} = 0. \tag{9}$$

Cancelling the denominator $1 + \mathbf{s}^T\mathbf{s}$ from (9), we get a polynomial constraint involving only $\mathbf{s}$:

$$e_{ij}^{alg\,1} = \hat{\mathbf{l}}_i \cdot \bar{\mathbf{R}}\mathbf{P}_{ij} + \hat{\mathbf{l}}_i \cdot \left(\mathbf{L}^T\mathbf{L}\right)^{-1}\mathbf{L}^T\mathbf{b}\left(\bar{\mathbf{R}}\right) = 0. \tag{10}$$

Considering the noise, we form the following least-squares problems for (10),

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} C_{alg1}, \quad C_{alg1} = \frac{1}{2}\sum_i \sum_j \left(e_{ij}^{alg1}\right)^2 \tag{11}$$

As e_{ij}^{alg1} is a second-order polynomial function of $\mathbf{s}$, C_{alg1} is a fourth-order polynomial function. To find the optimal solution of C_{alg1}, we consider the first-order optimality conditions of it,

$$f_1 = \frac{\partial C_{alg1}}{\partial s_1} = 0,\ f_2 = \frac{\partial C_{alg1}}{\partial s_2} = 0,\ f_3 = \frac{\partial C_{alg1}}{\partial s_3} = 0 \tag{12}$$

This is a ternary polynomial system of degree 3. We do not adopt the generally used Gröbner basis method (Zheng et al. 2013; Mirzaei and Roumeliotis 2011a; 2011b) to solve (12), since it may be plagued by numeric problems. Due to the special structure of (12), this equation system can be effectively solved by the hidden variable method according to the theory in Cox, Little, and O'Shea (2006); Gelfand, Kapranov, and Zelevinsky (2008). This method does not involve numerically unstable functions, such as the matrix inverse used in the Gröbner basis method. Therefore, it could be more stable.

Without loss of generality, we treat s_1 and s_2 as unknowns, and s_3 as a constant. Thus we have

$$\begin{aligned} f_i =& a_i s_1^3 + b_i s_1^2 s_2 + c_i s_1 s_2^2 + d_i s_2^3 + p_{i1}(s_3) s_1^2 + \\ & p_{i2}(s_3) s_1 s_2 + p_{i3}(s_3) s_2^2 + p_{i4}(s_3) s_1 + \\ & p_{i5}(s_3) s_2 + p_{i6}(s_3) = 0, i = 1, 2, 3, \end{aligned} \tag{13}$$

where $p_{in}(s_3)(n = 1, \cdots, 6)$ are polynomial functions of s_3. Then we convert f_i to a homogeneous equation by introducing an auxiliary variable s_0 to make all the monomials in f_i have degree three. This results in the following homogeneous equation system:

$$\begin{aligned} F_i =& a_i s_1^3 + b_i s_1^2 s_2 + c_i s_1 s_2^2 + d_i s_2^3 + p_{i1}(s_3) s_0 s_1^2 + \\ & p_{i2}(s_3) s_0 s_1 s_2 + p_{i3}(s_3) s_0 s_2^2 + p_{i4}(s_3) s_0^2 s_1 + \\ & p_{i5}(s_3) s_0^2 s_2 + p_{i6}(s_3) s_0^3 = 0, i = 1, 2, 3. \end{aligned} \tag{14}$$

We can find that $F_i = f_i$ when $s_0 = 1$. That is to say, given s_3, if $[s_1, s_2]$ is a solution of (13), $[1, s_1, s_2]$ will be a solution of (14), and vice versa.

Assume that a, b and c are non-negative integers and satisfy $a + b + c = 2$, such as $a = 2, b = 0, c = 0$. There are clearly 6 choices for a, b and c. Each of them will result in a 4th-order polynomial equation. For each choice, we can write F_1, F_2 and F_3 as

$$\begin{aligned} F_1 &= s_0^{a+1} P_1 + s_1^{b+1} Q_1 + s_2^{c+1} R_1, \\ F_2 &= s_0^{a+1} P_2 + s_1^{b+1} Q_2 + s_2^{c+1} R_2, \\ F_3 &= s_0^{a+1} P_3 + s_1^{b+1} Q_3 + s_2^{c+1} R_3. \end{aligned} \tag{15}$$

Let us take $a = 2, b = 0, c = 0$ as an example. According to (15), we have

$$F_i = s_0^3 P_i + s_1 Q_i + s_2 R_i, i = 1, 2, 3 \tag{16}$$

where $P_i = p_{i6}(s_3)$, $Q_i = a_i s_1^2 + b_i s_1 s_2 + c_i s_2^2 + p_{i1}(s_3) s_0 s_1 + p_{i2}(s_3) s_0 s_2 + p_{i4}(s_3) s_0^2$, $R_i = d_i s_0 s_2^2 + p_{i3}(s_3) s_0 s_2 + p_{i5}(s_3) s_0^2$.

The representation of P_i, Q_i and R_i is not uniform, but the solution of the polynomial system is independent of the choice of them as proved in (Gelfand, Kapranov, and Zelevinsky 2008). The above equations can be treated as a linear homogeneous system for s_0^{a+1}, s_1^{b+1} and s_2^{c+1}. If the polynomial system $F_1 = 0, F_2 = 0, F_3 = 0$ has a non-trivial solution, the determinant of the coefficient matrix in (15) should be zero. Then, we get

$$F_{abc} = \det \begin{pmatrix} P_1 & Q_1 & R_1 \\ P_2 & Q_2 & R_2 \\ P_3 & Q_3 & R_3 \end{pmatrix} = 0, \tag{17}$$

where $\det(\cdot)$ represents the determinant of a matrix. We can obtain 6 F_{abc} in (17) for the 6 possible combinations of $a + b + c = 2$. On the other hand, the solutions of $F_1 = 0, F_2 = 0, F_3 = 0$ are also the solutions of the following equations

$$u F_i = 0, \quad u = s_0, s_1, s_2, \quad i = 1, 2, 3. \tag{18}$$

We can have 9 uF_i in (18). F_{abc} and uF_i are all fourth-order polynomials in s_0, s_1 and s_2. Combining these equations, we have a linear homogeneous system of 15 equations and 15 monomials

$$\mathbf{M}(s_3)\,\mathbf{S} = \mathbf{0} \tag{19}$$

where $\mathbf{M}(s_3)$ is a 15×15 matrix with polynomials in s_3 as elements, and $\mathbf{S} = [s_0^4, s_0^3 s_1, s_0^3 s_2, s_0^2 s_1^2, s_0^2 s_1 s_2, s_0^2 s_2^2, s_0 s_1^3, s_0 s_1^2 s_2, s_0 s_1 s_2^2, s_0 s_2^3, s_1^4, s_1^3 s_2, s_1^2 s_2^2, s_1 s_2^3, s_2^4]^T$.

The linear homogeneous system $\mathbf{M}(s_3)\,\mathbf{S} = \mathbf{0}$ has a non-trivial solution if and only if $\det(\mathbf{M}(s_3)) = 0$. This can be formulated as a polynomial eigenvalue problem and can be efficiently addressed (Fitzgibbon 2001). After computing s_3, we solve the linear homogeneous equation (19) for $\mathbf{S}$. The solutions of s_1 and s_2 can be found from the second and third terms in $\mathbf{S}$, *i.e.* $s_0^3 s_1$ and $s_0^3 s_2$, by setting $s_0 = 1$ as mentioned above. Then we can get $\mathbf{R}$ from (1) and $\mathbf{t}$ from (8). As there are multiple solutions of (12), we choose the one with the smallest cost in (4) as the optimal solution.

Second Algebraic Distance

The global minimizer of the algebraic distance (11) is probably not the optimal solution of (4). The global minimizer of the reprojection distance generally provides better results (Hartley and Zisserman 2003). After obtaining the initial estimation of $\mathbf{R}$ and $\mathbf{t}$, iterative optimization algorithms on the reprojection error can be used to refine the solution. However, this is time-consuming for a large-scale problem. To solve this problem, we introduce a second algebraic distance that approximates the reprojection distance (3), but its Hessian matrix and gradient are easy to calculate.

One problem of e_{ij}^{rep} in (3) is that its denominator involves $\mathbf{R}$ and $\mathbf{t}$, and varies for different 2D/3D line correspondences. Thus, it is difficult to calculate the cost function in (4). However, if we fix the denominator of (3) at a certain $\mathbf{R_0}$ and $\mathbf{t_0}$, we can get an approximation of (3) around $\mathbf{R_0}$ and $\mathbf{t_0}$. We define our second algebraic distance as:

$$e_{ij}^{alg2} = \frac{\hat{\mathbf{p}}_{ij} \cdot \mathbf{l}_i}{\sqrt{\left(l_i^1(\mathbf{R}_0, \mathbf{t}_0)\right)^2 + \left(l_i^2(\mathbf{R}_0, \mathbf{t}_0)\right)^2}}. \tag{20}$$

This yields the least-squares problem as:

$$\hat{\mathbf{s}}, \hat{\mathbf{t}} = \arg\min_{\mathbf{s},\mathbf{t}} C_{alg2}, \quad C_{alg2} = \frac{1}{2} \sum_i \sum_j \left(e_{ij}^{alg2}\right)^2 \tag{21}$$

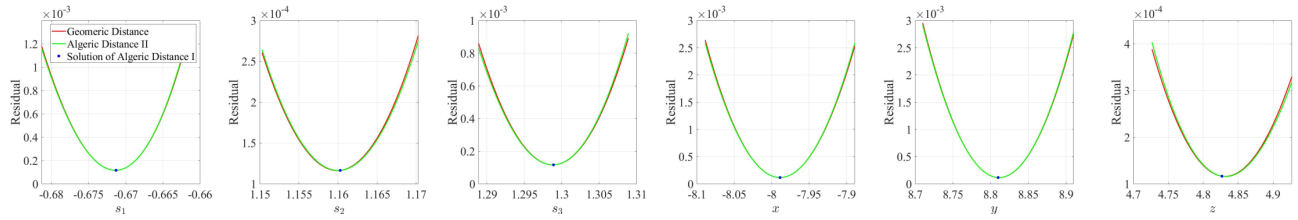


Figure 2: Compare the reprojection distance based cost function C_{rep} with our second algebraic distance based cost function C_{alg2} using 10 lines with zero-mean 2 pixels standard deviation Gaussian Noise. The denominator of e_{ij}^{rep} is fixed at the solution of the first algebraic distance. We can find that C_{rep} and C_{alg2} are very similar. Each figure is obtained by varying one variable and fixing the remaining variables.

When $\mathbf{R_0}$ and $\mathbf{t_0}$ are close to the optimal solution, e_{ij}^{alg2} will approximate e_{ij}^{rep} around the optimal solution. Thus C_{alg2} will accordingly approximate C_{rep} around the optimal solution. Figure 2 compares C_{alg2} with C_{rep} for 10 lines. It is obvious that C_{alg2} can well approximate C_{rep} using the solution from the first step. As the denominator of (20) is a scalar now, e_{ij}^{alg2} is a polynomial function. We can easily calculate the summation C_{alg2}.

We do not directly solve the first-order optimality condition of C_{alg2}. Instead, we use the damped Newton iteration to minimize this function, as the Hessian matrix and the gradient can be efficiently calculated for the polynomial function. The iterative algorithm is initialized with the solution from the first step. For the kth iteration, we compute the Hessian matrix $\mathbf{H}$ and the gradient vector $\nabla\mathbf{C}_{alg2}$ of $\mathbf{C}_{alg2}$. Then we update the solution with $[\mathbf{s}_{k+1};\mathbf{t}_{k+1}] = [\mathbf{s}_k;\mathbf{t}_k] - (\mathbf{H}+\lambda\mathbf{I}_6)^{-1}\nabla\mathbf{C}_{alg2}$. λ is adjusted according to the LM algorithm (Moré 1978) to ensure the cost function is reduced at every step.

We can repeat this process again when the minimization of C_{alg2} converges. But we did not observe significant improvement for additional iterations in our experiments. This is because the result from the first step is close to the optimal solution. In this case, C_{alg2} can well approximate C_{rep}, as illustrated in Figure 2. As a result, the global minimum of C_{alg2} approximates the global minimum of C_{rep}.

We summarize the main part of our algorithm as follows:

Input: $N(N \geq 3)$ 2D/3D line correspondences
Output: camera pose $\mathbf{R}$ and $\mathbf{t}$ relative to the world frame

1. Construct the cost function C_{alg1} in (11).
2. Solve the first-order optimality conditions of C_{alg1} with the hidden variable method introduced above to get the CGR parameters $\mathbf{s}$. Then solve for $\mathbf{t}$ by (8).
3. Choose $\mathbf{s}_0$ and $\mathbf{t}_0$ with the minimal cost of (4) as the solution.
4. Construct C_{alg2} using $\mathbf{s}_0$ and $\mathbf{t}_0$, and minimize C_{alg2} by the damped Newton iteration.

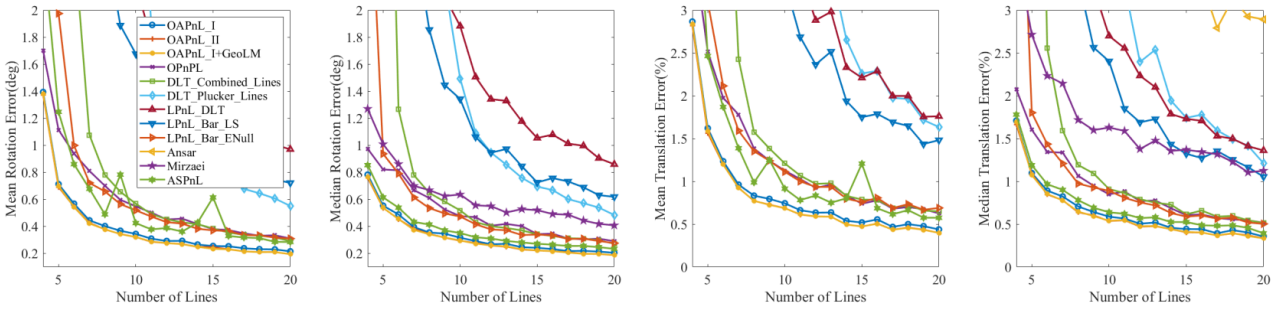
Experiments

In this section, we compare our algorithm, referred to as **OAPnL**, with previous works including **OPnPL** (Vakhitov, Funke, and Moreno-Noguer 2016), **DLTCombined_Lines** (Bronislav Přibyl 2015), **DLT_Plucker_Lines** (Přibyl, Zemčík, and Čadík 2017), **LPnL_DLT** (Xu et al. 2017), **LPnL_Bar_LS** (Xu et al. 2017), **LPnL_Bar_ENull** (Xu et al. 2017), **ASPnL** (Xu et al. 2017), **Ansar** (Ansar and Daniilidis 2003), **Mirzaei** (Mirzaei and Roumeliotis 2011b). We evaluate the results from the two steps of our algorithm denoted as **OAPnL_I** and **OAPnL_II**, respectively. We also minimize the reprojection cost (4) by the LM method (Moré 1978), initialized by the solution of **OAPnL_I**. We denote it as **OAPnL_I+GeoLM**. We do not consider the 2D/3D endpoint mismatching problem as most related works, since it can be solved by shifting the endpoint and computing a new PnL problem (Vakhitov, Funke, and Moreno-Noguer 2016). Thus we focus on the accuracy of the PnL algorithm itself. The algorithms are evaluated by accuracy and computational time.

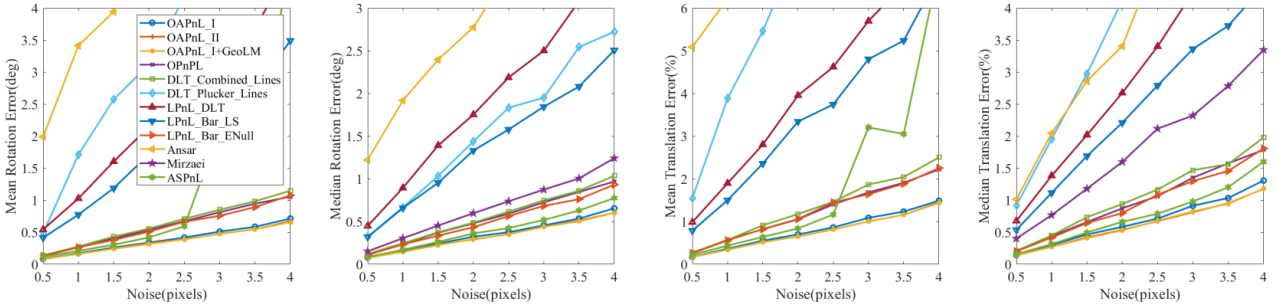
Experiments with Synthetic Data

The virtual camera has resolution 640×480 pixels and focal length 800. The camera is randomly placed within a $[-10m, 10m]^3$ cube. We uniformly sample the Euler angles α, β, γ of the rotation matrix ($\alpha, \gamma \in [0°, 360°]$ and $\beta \in [0°, 180°]$). N 2D/3D line correspondences are randomly yielded for each trial using the method in (Xu et al. 2017). Specifically, we first generate the end points of the 2D lines, then the corresponding 3D lines are reconstructed by back-projecting the end points of the 2D lines. The depths of the 3D points are within $[4m, 10m]$. In addition to the configuration that the 2D line segments are uniformly distributed in the whole image (denoted as **centered case**), we also considered two challenging configurations mentioned is the literature, *i.e.* **uncentered case** (Xu et al. 2017) and **planar case** (Vakhitov, Funke, and Moreno-Noguer 2016). In the **uncentered case**, the 2D line segments are within the region $[0, 160] \times [0, 120]$ pixels. In the **planar case**, all the 3D lines are on a plane. We randomly generate a plane in front of the camera, then calculate the intersection lines between the back-projections of the 2D lines and the plane.

The result of each experiment is obtained from 500 independent trials. Denote the estimated rotation and translation as $\hat{\mathbf{R}}$ and $\hat{\mathbf{t}}$, and the ground truth as $\mathbf{R}_{gt}$ and $\mathbf{t}_{gt}$. We evaluate the rotation error by the angle of the axis-angle representation of $\mathbf{R}_{gt}^{-1}\mathbf{R}$ as Přibyl, Zemčík, and Čadík (2017), and the translation error by $\left\|\mathbf{t}_{gt} - \hat{\mathbf{t}}\right\|_2 / \left\|\mathbf{t}_{gt}\right\|_2$.

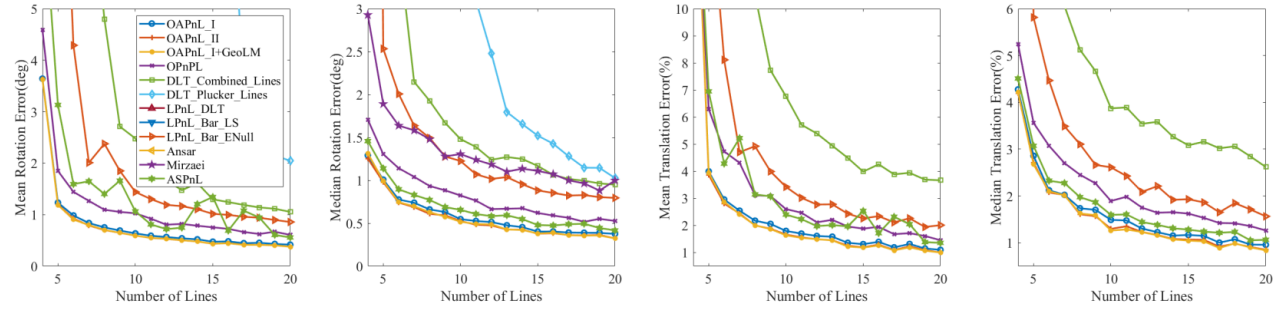


(a) Centered non-planar case: $N = 4, 5, \cdots, 20$ and $\delta = 2$ pixels

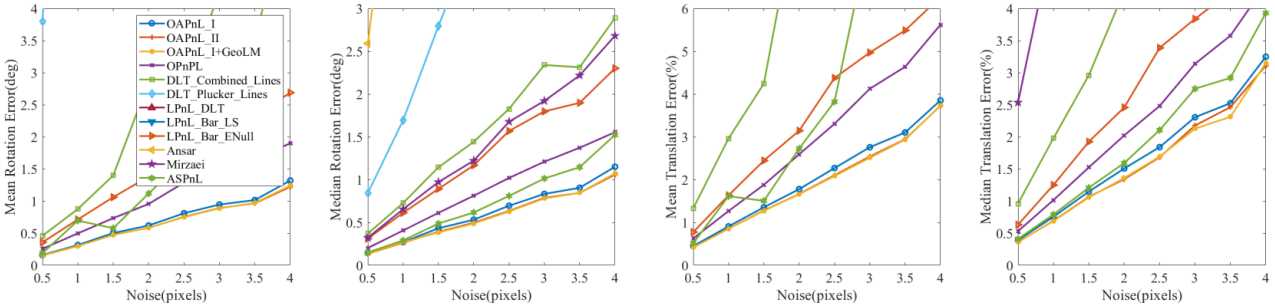


(b) Centered non-planar case: $N = 10$ and $\delta = 0.5, 1, \cdots, 4$ pixels

Figure 3: Experimental results for the centered non-planar configurations.



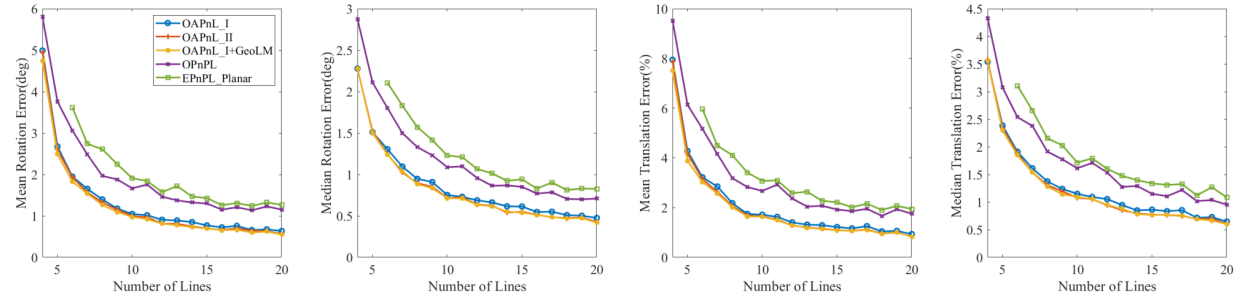
(a) Uncentered non-planar case: $N = 4, 5, \cdots, 20$ and $\delta = 2$ pixels



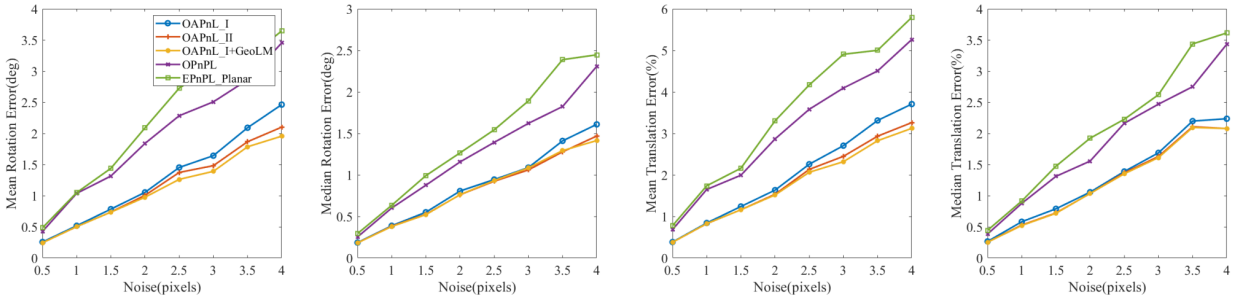
(b) Uncentered non-planar case: $N = 10$ and $\delta = 0.5, 1, \cdots, 4$ pixels

Figure 4: Experimental results for the uncentered non-planar configurations.

Nonplanar case We first consider the non-planar case. We evaluate the robustness of different algorithms by two experiments. Denote the standard deviation of a zero mean Gaussian noise as δ. The first experiment varies the num-



(a) Planar case: $N = 4, 5, \cdots, 20$ and $\delta = 2$ pixels



(b) Planar case: $N = 10$ and $\delta = 0.5, 1, \cdots, 4$ pixels

Figure 5: Experimental results for the centered planar configuration

ber of 2D/3D line correspondences N from 4 to 20 with a fixed standard deviation $\delta = 2$ pixels. The second experiment keeps $N = 10$, while δ is increased from 0.5 to 4 pixels. The means and medians of the rotation and translation errors are shown in Figure 3 and 4. The uncentered case is more challenging than the centered case. The estimation error of each algorithm is larger in the uncentered case. Our algorithm is superior to the previous works. The results from **OAPnL_I** are already more accurate than the previous works, and are close to the results from **OAPnL_I+GeoLM**. **OAPnL_II** gives better result than **OAPnL_I**, and is comparable with **OAPnL_I+GeoLM**.

Planar case Most of the algorithms fail to deal with the planar case, including some nonlinear formulation algorithms, such as ASPnL (Xu et al. 2017) and Mirzaei (Mirzaei and Roumeliotis 2011b). This phenomenon is also observed in (Přibyl, Zemčík, and Čadík 2017) and (Vakhitov, Funke, and Moreno-Noguer 2016). As in the non-planar case, we evaluate the robustness of different algorithms by two experiments, *i.e.* varying $N \in [4, 20]$ while fixing $\delta = 2$ pixels, and increasing δ from 0.5 to 4 pixels with step 0.5 while keeping $N = 10$. For the line distribution, we consider the centered case. We compared our algorithm with **OPnPL** and **EPnPL_Planar**. **EPnPL_Planar** is an extension of the planar **EPnP** algorithm (Lepetit, Moreno-Noguer, and Fua 2008) to the PnL problem introduced in Vakhitov, Funke, and Moreno-Noguer (2016). Figure 5 shows the results of different algorithms. It is clear that our algorithm outperforms other algorithms. In most cases, **OAPnL_II** gives comparable result to **OAPnL_I+GeoLM**. When the noise is large as shown in Figure 5 (b), **OAPnL_I+GeoLM** gives slightly better result than **OAPnL_II**. This is because the solution of **OAPnL_I** may diverge away from the optimal solution when the noise increases. This enlarges the difference between e_{ij}^{alg2} and e_{ij}^{rep} around the optimal solution.

Experiments with Real Data

We also compare our algorithm with previous works using real images. Ten datasets (including VGG dataset and MPI dataset (Jain et al. 2010)) with ground truth camera poses and 2D/3D line correspondences are used to evaluate the algorithms. The details of the datasets are listed in Table 1.

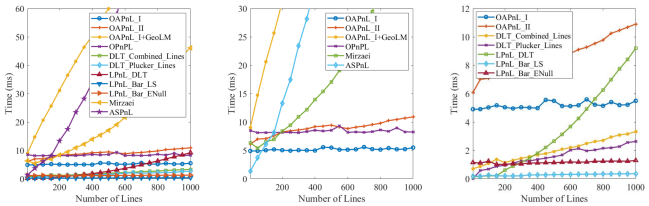
In this experiment, we use the absolute translation error $\left\|\mathbf{t}_{gt} - \hat{\mathbf{t}}\right\|_2$ as (Přibyl, Zemčík, and Čadík 2017). The rotation error is the same as the experiment with synthetic data. We compute the mean rotation and translation errors of our algorithm and **OPnPL**. The estimation errors of other algorithms are from Přibyl, Zemčík, and Čadík (2017). The experimental results are shown in Table 1. Our algorithm **OAPnL_I** and **OAPnL_II** outperform other algorithms. **OAPnL_II** gives similar results as **OAPnL_I+GeoLM**.

Computational Time

We evaluate the computational time of different algorithms on a laptop with a i7 2.9 GHZ cpu. The number of lines N varies from 5 to 1000. We conducted 500 independent trials for each N. We do not plot the computational time of **Ansar**'s algorithm, as it is too slow. Figure 6 (a) gives the average computational time of all the algorithms in milliseconds (ms) w.r.t. N. Our algorithm is slower than most of the algorithms based on linearization, but is faster than

Table 1: Experimental results and dataset characteristics. $\Delta\theta(^\circ)$ is the angle of the angle-axis representation of $\mathbf{R}_{gt}^{-1}\hat{\mathbf{R}}$. $\Delta\mathbf{t}(m)$ is the absolute translation error $\left\|\mathbf{t}_{gt}-\hat{\mathbf{t}}\right\|_2$. The best results are labeled by bold font.

Dataset		BB	STR	TFH	MH	COR	MC1	MC2	MC3	ULB	WDC
#images		66	20	72	10	11	3	3	3	3	5
#lines		870	1841	828	30	69	295	302	177	253	380
Mirzaei	$\Delta\theta$	88.18	0.90	32.24	0.46	0.22	4.83	15.47	5.00	2.51	36.52
	$\Delta\mathbf{t}$	168.47	1.92	11.04	0.04	0.10	1.53	7.37	1.82	1.27	6.44
OPnPL	$\Delta\theta$	0.19	0.09	0.42	0.28	0.07	0.03	0.03	0.06	0.06	0.13
	$\Delta\mathbf{t}$	0.81	0.07	0.31	0.02	0.02	0.01	0.01	0.02	0.02	0.06
DLT_Combd_Lines	$\Delta\theta$	0.40	0.22	0.39	0.41	0.11	0.11	0.15	0.16	0.20	0.23
	$\Delta\mathbf{t}$	1.88	0.38	0.32	0.04	0.04	0.04	0.07	0.05	0.08	0.12
DLT_Plücker_Lines	$\Delta\theta$	1.04	0.93	1.11	17.58	0.38	0.28	0.22	0.48	0.77	0.34
	$\Delta\mathbf{t}$	1.88	0.38	0.32	0.04	0.04	0.04	0.07	0.05	0.08	0.12
LPnL_Bar_ENull	$\Delta\theta$	0.30	0.11	0.57	0.32	0.10	0.04	0.03	0.07	0.39	0.08
	$\Delta\mathbf{t}$	1.13	0.16	0.45	0.02	0.04	0.01	0.02	0.02	0.18	0.05
LPnL_Bar_LS	$\Delta\theta$	1.98	0.15	1.10	0.45	0.13	0.03	0.03	0.09	0.49	0.18
	$\Delta\mathbf{t}$	7.23	0.27	1.05	0.04	0.05	0.01	0.02	0.03	0.22	0.11
ASPnL	$\Delta\theta$	37.82	22.08	7.76	0.25	0.10	0.15	0.20	2.08	4.89	0.51
	$\Delta\mathbf{t}$	76.61	30.47	6.11	0.02	0.03	0.04	0.08	0.74	2.22	0.23
OAPnL_I	$\Delta\theta$	**0.18**	**0.08**	0.60	0.24	0.05	0.03	0.02	0.07	0.12	0.09
	$\Delta\mathbf{t}$	**0.78**	0.06	0.48	0.02	0.02	0.01	**0.01**	0.02	0.05	0.04
OAPnL_II	$\Delta\theta$	**0.18**	**0.08**	0.10	**0.22**	**0.03**	**0.01**	**0.01**	**0.02**	**0.03**	**0.04**
	$\Delta\mathbf{t}$	**0.78**	**0.03**	0.07	**0.01**	**0.01**	**0.003**	**0.01**	**0.01**	**0.01**	**0.02**
OAPnL_I+LM	$\Delta\theta$	**0.18**	**0.08**	**0.09**	**0.22**	**0.03**	**0.01**	**0.01**	**0.02**	**0.03**	**0.04**
	$\Delta\mathbf{t}$	**0.78**	**0.03**	**0.06**	**0.01**	**0.01**	**0.003**	**0.01**	**0.01**	**0.01**	**0.02**



(a) (b) (c)

Figure 6: (a) Computation time of all the algorithms. (b) Computational time of algorithms based on nonlinear formulation. (c) Computation time of our algorithm and algorithms based on linear formulation

LPnL_DLT when N is large. However, our algorithm is superior to them in terms of accuracy and applicability. Figure 6 (b) illustrates the computational time of the algorithms based on nonlinear formulation. **OAPnL_I** is faster than all of them except **ASPnL** when N is small, as **ASPnL** only needs to solve a 15th order polynomial equation. But the running time of **ASPnL** quickly increases, thus it is not suitable for real-time applications when N is large. **OAPnL_II** is much more efficient than **OAPnL_I+GeoLM**. **OAPnL_I** and **OAPnL_II** can process 1000 lines around 5ms and 11ms, respectively. Thus, our algorithm is applicable to real-time applications.

Conclusions and Future Work

In this paper we propose a novel algorithm to address the PnL problem. We design two algebraic distances to approximate the reprojection distance. The PnL problem is solved by consecutively minimizing two polynomial cost functions derived from the two algebraic distances. We introduce a novel hidden variable method to solve the first-order optimality conditions of the first problem. This method utilizes the special structure of the resulting polynomial system. This makes it more stable than the general Gröbner basis based methods adopted in the previous works. This method can be extended to other problems which are formulated as a similar polynomial system. We adopt the damped Newton iteration to minimize the second minimization problem, as the Hessian matrix and the gradient can be efficiently calculated for the polynomial cost function. We evaluated our algorithm by experiments on synthetic and real data. The results show that the first algebraic distance alone outperforms the state-of-the-art methods in terms of accuracy and applicability. The second step is comparable to the iterative method based on the reprojection distance, but much faster. Our proposed algorithm is scalable and applicable to real-time applications.

A more efficient method to solve (19) is to compute $\det(\mathbf{M}(s_3))$ by the quotient-free Gaussian Elimination (Hartley and Li 2012). We plan to implement this algorithm to further improve the speed of our algorithm.

References

Ansar, A., and Daniilidis, K. 2003. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25(5):578–589.

Bronislav Přibyl, Pavel Zemčík, M. Č. 2015. Camera pose estimation from lines using plücker coordinates. In *Proceedings of the British Machine Vision Conference (BMVC 2015)*, 1–12. The British Machine Vision Association and Society for Pattern Recognition.

Byröd, M.; Josephson, K.; and Åström, K. 2009. Fast and stable polynomial equation solving and its application to computer vision. *International Journal of Computer Vision* 84(3):237–256.

Chen, H. H. 1990. Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions. In *Proceedings Third International Conference on Computer Vision*, 374–378. IEEE.

Cox, D. A.; Little, J.; and O'shea, D. 2006. *Using algebraic geometry*, volume 185. Springer Science & Business Media.

David, P.; DeMenthon, D.; Duraiswami, R.; and Samet, H. 2003. Simultaneous pose and correspondence determination using line features. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, II–424–II–431 vol.2.

Dhome, M.; Richetin, M.; Lapreste, J. T.; and Rives, G. 1989. Determination of the attitude of 3d objects from a single perspective view. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11(12):1265–1278.

Fischler, M. A., and Bolles, R. C. 1987. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in computer vision*. Elsevier. 726–740.

Fitzgibbon, A. W. 2001. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, 125–132.

Gelfand, I. M.; Kapranov, M.; and Zelevinsky, A. 2008. *Discriminants, resultants, and multidimensional determinants*. Springer Science & Business Media.

Hartley, R., and Li, H. 2012. An efficient hidden variable approach to minimal-case camera motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(12):2303–2314.

Hartley, R., and Zisserman, A. 2003. *Multiple view geometry in computer vision*. Cambridge university press.

Jain, A.; Kurz, C.; Thormählen, T.; and Seidel, H.-P. 2010. Exploiting global connectivity constraints for reconstruction of 3d line segment from images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010)*.

Kumar, R., and Hanson, A. 1994. Robust methods for estimating pose and a sensitivity analysis. *CVGIP: Image Understanding* 60(3):313 – 342.

Lepetit, V.; Moreno-Noguer, F.; and Fua, P. 2008. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision* 81(2):155.

Liu, Y.; Huang, T. S.; and Faugeras, O. D. 1990. Determination of camera location from 2-d to 3-d line and point correspondences. *IEEE Transactions on pattern analysis and machine intelligence* 12(1):28–37.

Micusik, B., and Wildenauer, H. 2017. Structure from motion with line segments under relaxed endpoint constraints. *International Journal of Computer Vision* 124(1):65–79.

Mirzaei, F. M., and Roumeliotis, S. I. 2011a. Globally optimal pose estimation from line correspondences. In *Proceedings of 2011 IEEE International Conference on Robotics and Automation*, 5581–5588.

Mirzaei, F. M., and Roumeliotis, S. I. 2011b. Optimal estimation of vanishing points in a manhattan world. In *Proceedings of 2011 International Conference on Computer Vision*, 2454–2461.

Moré, J. J. 1978. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*. Springer. 105–116.

Přibyl, B.; Zemčík, P.; and Čadík, M. 2017. Absolute pose estimation from line correspondences using direct linear transformation. *Computer Vision and Image Understanding* 161:130 – 144.

Přibyl, B.; Zemčík, P.; and Čadík, M. 2017. Absolute pose estimation from line correspondences using direct linear transformation. *Computer Vision and Image Understanding*.

Vakhitov, A.; Funke, J.; and Moreno-Noguer, F. 2016. Accurate and linear time pose estimation from points and lines. In Leibe, B.; Matas, J.; Sebe, N.; and Welling, M., eds., *Computer Vision – ECCV 2016*, 583–599. Cham: Springer International Publishing.

Von Gioi, R. G.; Jakubowicz, J.; Morel, J.-M.; and Randall, G. 2010. Lsd: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence* 32(4):722–732.

Xu, C.; Zhang, L.; Cheng, L.; and Koch, R. 2017. 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.

Zhang, L., and Koch, R. 2014. Structure and motion from line correspondences: Representation, projection, initialization and sparse bundle adjustment. *Journal of Visual Communication and Image Representation* 25(5):904 – 915.

Zhang, Y.; Li, X.; Liu, H.; and Shang, Y. 2016. Probabilistic approach for maximum likelihood estimation of pose using lines. *IET Computer Vision* 10(6):475–482.

Zheng, Y.; Kuang, Y.; Sugimoto, S.; Åström, K.; and Okutomi, M. 2013. Revisiting the pnp problem: A fast, general and optimal solution. In *Proceedings of 2013 IEEE International Conference on Computer Vision*, 2344–2351.

Zhou, F.; Duh, H. B.-L.; and Billinghurst, M. 2008. Trends in augmented reality tracking, interaction and display: A review of ten years of ismar. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '08, 193–202. Washington, DC, USA: IEEE Computer Society.