Pattern Recognition Letters 108 (2018) 31–37

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A simple, robust and fast method for the perspective-n-point Problem

Ping Wang[a], Guili Xu[a,*], Yuehua Cheng[b], Qida Yu[a]

[a] College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China
[b] College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China

ARTICLE INFO

Article history:
Received 18 March 2017
Available online 5 March 2018

MSC:
41A05
41A10
65D05
65D17

Keywords:
Perspective-n-point problem (PnP)
Absolute position and orientation
Camera pose estimation
Vision-based navigation
Computer vision

ABSTRACT

In this work, we present a simple, robust and fast method to the perspective-n-point (PnP) problem for determining the position and orientation of a calibrated camera from known reference points. Our method transfers the pose estimation problem into an optimal problem, and only needs to solve a seventh-order and a fourth-order univariate polynomial, respectively, which makes the processes more easily understood and significantly improves the performance. Additionally, the number of solutions of the proposed method is substantially smaller than existing methods. Experiment results show that the proposed method can stably handle all 3D point configurations, including the ordinary 3D case, the quasi-singular case, and the planar case, and it offers accuracy comparable or better than that of the state-of-art methods, but at much lower computational cost.

1. Introduction

Determining the position and orientation of a calibrated camera from n 3D points and their 2D projections, which is also known as the perspective-n-point (PnP) problem [5], has numerous applications in computer vision and robotics. Examples include robot localization [6,8], augmented reality [17], structure-from-motion (SfM) [4], spacecraft pose estimation during descent and landing [18,23]. Considering the importance of the PnP problem, a large amount of research effort has been devoted to solving this problem over the past few decades. The existing PnP methods can be classified as iterative and non-iterative methods.

Classical iterative methods formulate the PnP problem into a non-linear least-squares problem [15], and then solve it using iterative optimization methods, i. e., Gauss–Newton and Levenberg–Marquardt [9] method. However, iterative methods are sensitive to the initialization, and are easily trapped into a local minimum, which will leads to poor accuracy, especially when no redundant points ($n \leq 6$) are available.

For non-iterative methods, the traditional methods apply linear operations to obtain solutions, i. e., the DLT [1] and HOMO method [16]. Non-iterative methods have an advantage of less computing costs, but are sensitive to noise. Quan and Lan [21] and Ansar and Daniilidis [2] presented two linear solutions for the PnP problem, with respective computational complexity $O(n^5)$ and $O(n^8)$. However, they are still inaccurate when n is small. On the contrary, they are very time-consuming when n is large. To overcome these problems, Lepetit et al. [12] introduced four virtual control points to represent the 3D reference points, and proposed the first linear complexity method, named EPnP, with respect to the number of the points. EPnP is computationally efficient, but is inaccurate for $n = 4$ or 5, due to its underlying linearization scheme. To improve accuracy, Li et al. [14] proposed another non-iterative $O(n)$ solution, named RPnP, which transfers the PnP problem into a sub-optimal problem by solving a seventh-order polynomial. RPnP is very efficient and works well for both non-redundant ($n \leq 6$) and redundant points cases. Hesch and Roumeliotis [7] developed the first globally optimal method (called DLS) with complexity $O(n)$, which formulates the PnP problem into a multivariate polynomial system using the camera measurement equations, and employs the multiplication matrix to determine all roots of the system. Unfortunately, the accuracy of DLS is unstable because of the Cayley parameterization, which has a singularity for any 180-degree rotations. To resolve these drawbacks, Zheng et al. [26] proposed the OPnP method, which adopts the non-unit quaternion parameterization to replace the Cayley parameterization, and uses the Gröbner basis [11] to solve the PnP problem. To our knowledge, OPnP is one of the most accurate non-iterative methods until now. To ex-

[*] Corresponding author.
E-mail addresses: pingwangsky@gmail.com (P. Wang), guilixu2002@163.com (G. Xu).

https://doi.org/10.1016/j.patrec.2018.02.028
0167-8655/© 2018 Elsevier B.V. All rights reserved.

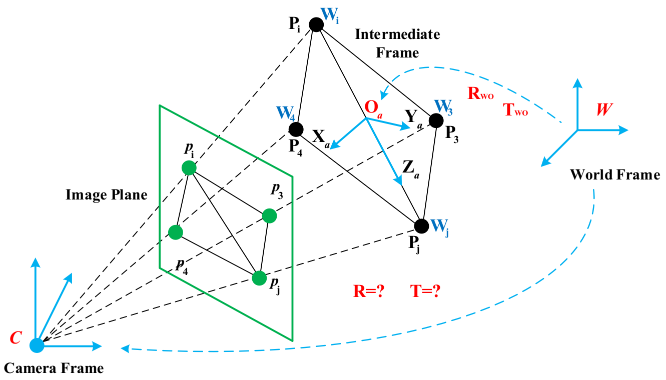


Fig. 1. Illustration of the P*n*P problem.

tend the scope of application, Kneip et al. [10] presented the UP*n*P method, which is applicable to both central and non-central camera systems [3]. However, its accuracy is worse than OP*n*P in some configurations. Recently, the P*n*P*f* [20,25] and P*n*P*fr* [19] methods were proposed to solve the pose estimation problem in the case of an uncalibrated camera. However, the accuracy of these methods is usually lower than the classical P*n*P method.

The most recent works, i. e., DLS, OP*n*P, and UP*n*P method, formulate the P*n*P problem as a minimization problem, and then solve it using the Gröbner basis technique. However, the Gröbner basis technique needs to construct a large elimination template, because of the large number of unknowns and the high maximum degree of the P*n*P problem. This process takes significant time and is difficult to assure reliability. All of these disadvantages will reduce overall performance and limit general understanding.

In contrast to previous methods, in this paper we propose a simple, robust, and fast solution to the classical P*n*P problem. Our method transfers the P*n*P problem into an optimal one, that only needs to solve a seventh-order and fourth-order univariate polynomial without using the Gröbner basis technique. The number of the solutions for our method is substantially smaller than existing globally optimal methods, i. e., the DLS and OPnP method. All of these make the processes more easily applicable and significantly improve the performance. The experiment results show that our method can stably address all 3D point configurations, including the ordinary 3D case, quasi-singular case, and planar case. It also offers accuracy comparable to the leading methods, but at much lower computational cost.

The rest of the paper is organized as follows. Section 2 presents the derivations of our method. Section 3 provides a thorough analysis of the proposed method by simulated experiments. Section 4 shows the real tests. Section 5, finally, concludes the work.

2. Proposed method

As shown in Fig. 1, suppose a reference point W_i whose coordinates in the world frame and the normalized image plane are $W_i = [X_i^w, Y_i^w, Z_i^w]^T$ and $f_i = [u_i, v_i, 1]^T$, respectively. Note that the superscript indicates the different coordinate frame, i. e., w indicates the world frame. Our goal is to retrieve the rotation matrix R and the translation vector t between the world frame and the camera frame using n ($n \geq 4$) reference points when the camera is calibrated.

2.1. Building an object frame

The first step involves the definition of a new, intermediate object frame from the 3D reference points. As shown in Fig. 1, we choose the center of $\overrightarrow{W_iW_j}$ as the origint O_a, and create an intermediate frame $[O_a - \vec{X}_a, \vec{Y}_a, \vec{Z}_a]$, where

$$\begin{aligned} \vec{X}_a &= \frac{W_j - O_a}{\|W_j - O_a\|} \\ \vec{Z}_a &= \frac{\vec{X}_a \times [0, 1, 0]^T}{\|\vec{X}_a \times [0, 1, 0]^T\|} \\ \vec{Y}_a &= \frac{\vec{Z}_a \times \vec{X}_a}{\|\vec{Z}_a \times \vec{X}_a\|}, \end{aligned} \tag{1}$$

if $|[0, 1, 0]^T \vec{X}_a| \leq |[0, 0, 1]^T \vec{X}_a|$, and

$$\begin{aligned} \vec{X}_a &= \frac{W_j - O_a}{\|W_j - O_a\|} \\ \vec{Y}_a &= \frac{[0, 0, 1]^T \times \vec{X}_a}{\|[0, 0, 1]^T \times \vec{X}_a\|} \\ \vec{Z}_a &= \frac{\vec{X}_a \times \vec{Y}_a}{\|\vec{X}_a \times \vec{Y}_a\|}, \end{aligned} \tag{2}$$

if $|[0, 1, 0]^T \vec{X}_a| > |[0, 0, 1]^T \vec{X}_a|$.

Via the transformation matrix $T_{wo} = [\vec{X}_a, \vec{Y}_a, \vec{Z}_a]^T$, the reference point $W_i = [X_i^w, Y_i^w, Z_i^w]^T$ can be easily transformed into the intermediate frame using

$$P_i = T_{wo}(W_i - O_a) \qquad i = 1, 2, \ldots, n, \tag{3}$$

where $P_i = [X_i^p, Y_i^p, Z_i^p]^T$, and the superscript p indicates the intermediate object frame.

2.2. Determining a rotation axis using least-square residual

Every remaining point together with the P_i and P_j forms a 3-point subsets. By using the P3P (perspective-three-point) constraint [13], each subset can build a fourth-order polynomial as follows:

$$\begin{cases} f_1(x) = a_1x^4 + b_1x^3 + c_1x^2 + d_1x + e_1 = 0 \\ f_2(x) = a_2x^4 + b_2x^3 + c_2x^2 + d_2x + e_2 = 0 \\ \ldots \\ f_{n-2}(x) = a_{n-2}x^4 + b_{n-2}x^3 + c_{n-2}x^2 + \\ \qquad d_{n-2}x + e_{n-2} = 0 \end{cases}. \tag{4}$$

Instead of directly solving a series of fourth-order polynomials, a cost function $F = \sum_{i=1}^{n-2} f_i^2(x)$ is defined as the square sum of these polynomials. The minima of F can then 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 seventh-order polynomial, which has at most 4 minima, and can be easily solved by the eigenvalue method [22]. Once the minimal of F is determined, the depths of P_i and P_j can be calculated according to the P3P constraint [13], and then the rotation axis Z_a can be calculated as $Z_a = \overrightarrow{P_iP_j}/\|P_iP_j\|$.

2.3. Retrieving the pose by solving an optimal problem

When the Z_a-axis of $[O_a - \vec{X}_a, \vec{Y}_a, \vec{Z}_a]$ is determined, the transformation from the intermediate object frame $[O_a - \vec{X}_a, \vec{Y}_a, \vec{Z}_a]$ to the camera frame $[O_c - \vec{X}_c, \vec{Y}_c, \vec{Z}_c]$ can be expressed as

$$\lambda_i f_i = R_c P_i + t_c \qquad i = 1, 2, \ldots, n, \tag{5}$$

where

$$R_c = R_1 R_2 = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & x & -y \\ 0 & y & x \end{bmatrix}$$

and $t_c = [t_1, t_2, t_3]^T$. R_1 is an arbitrary rotation matrix whose third column $[r_3, r_6, r_9]^T$ equals the rotation axis Z_a, and R_1 should meet the orthogonal constraint of the rotation matrix. R_2 denotes a rotation of α degrees around the $Z - axis$, with $x = cos\alpha$ and $y = sin\alpha$.

R_c can be further expressed as

$$R_c = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} = \begin{bmatrix} r_1 & r_2x + r_3y & -r_2y + r_3x \\ r_4 & r_5x + r_6y & -r_5y + r_6x \\ r_7 & r_8x + r_9y & -r_8y + r_9x \end{bmatrix}. \tag{6}$$

By combining Eqs. (5) and (6), we have

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} P_i + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}. \tag{7}$$

After eliminating the λ_i, Eq. (7) becomes

$$\begin{aligned} t_1 - t_3u_i &= m_3^T P_i u_i - m_1^T P_i \\ t_2 - t_3v_i &= m_3^T P_i v_i - m_2^T P_i \qquad i = 1, 2, \ldots n. \end{aligned} \tag{8}$$

By denoting a new unknown $s = [x, y, 1]^T$, Eq. (8) can be represented as

$$A_i t_c = B_i s \qquad i = 1, 2, \ldots n, \tag{9}$$

where

$$A_i = \begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \end{bmatrix}$$

and

$$B_i = \begin{bmatrix} r_8Y_i^p u_i + r_9Z_i^p u_i - r_2Y_i^p - r_3Z_i^p & r_9Y_i^p u_i - r_8Z_i^p u_i - r_3Y_i^p + r_2Z_i^p & r_7X_i^p u_i - r_1X_i^p \\ r_8Y_i^p v_i + r_9Z_i^p v_i - r_5Y_i^p - r_6Z_i^p & r_9Y_i^p v_i - r_8Z_i^p v_i - r_6Y_i^p + r_5Z_i^p & r_7X_i^p v_i - r_4X_i^p \end{bmatrix}.$$

Eq. (9) is satisfied for every points, hence

$$\begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_n \end{bmatrix} t_c = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} s \Longleftrightarrow At_c = Bs \Longleftrightarrow t_c = Cs, \tag{10}$$

where $C = (A^TA)^{-1}A^TB$.

Similarly, R_cP_i can also be expressed as

$$R_cP_i = Q(P_i)s, \tag{11}$$

where

$$Q(P_i) = \begin{bmatrix} r_2Y_i^p + r_3Z_i^p & r_3Y_i^p - r_2Z_i^p & r_1X_i^p \\ r_5Y_i^p + r_6Z_i^p & r_6Y_i^p - r_5Z_i^p & r_4X_i^p \\ r_8Y_i^p + r_9Z_i^p & r_9Y_i^p - r_8Z_i^p & r_7X_i^p \end{bmatrix}.$$

Replacing (10) and (11) in (5), expanding, and rearranging the terms, we have

$$\lambda_i e_i = M_i s \qquad i = 1, 2, \ldots, n, \tag{12}$$

where $e_i = f_i / \|f_i\|$ is the normalized direction vector and $M_i = Q(P_i) + C$.

From Eq. (12), λ_i can be expressed as

$$\lambda_i = e_i^T M_i s \qquad i = 1, 2, \ldots, n. \tag{13}$$

After plugging Eq. (13) back into (12), we have

$$e_i e_i^T M_i s = M_i s \qquad i = 1, 2, \ldots, n \tag{14}$$

Note that Eq. (14) is not perfectly satisfied due to the noise. The residual of Eq. (14) can be written as

$$\eta_i = (e_i e_i^T M_i - M_i)s = E_i s \qquad i = 1, 2, \ldots, n. \tag{15}$$

In addition, there is a constraint that $x^2 + y^2 = 1$. Hence, we directly minimize the sum of the squared residuals to build a cost function with a constraint. The simplified cost function is

$$\varepsilon = s^T G s + \lambda(1 - x^2 - y^2) \qquad i = 1, 2, \ldots, n, \tag{16}$$

in which

$$G = \sum_{i=1}^{n} E_i^T E_i = \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{12} & G_{22} & G_{23} \\ G_{13} & G_{23} & G_{33} \end{bmatrix}$$

is a know 3×3 symmetric matrix, and λ is a Lagrange multiplier. The minima of Eq. (16) can be determined by solving the polynomial system of its first-order optimality condition. By calculating the derivative of ε with respect to x, y and λ, the first-order optimality condition reads

$$\begin{aligned} \frac{\partial \varepsilon}{\partial x} &= G_{11}x + G_{12}y - \lambda x + G_{13} = 0 \\ \frac{\partial \varepsilon}{\partial y} &= G_{12}x + G_{22}y - \lambda y + G_{23} = 0 \\ \frac{\partial \varepsilon}{\partial \lambda} &= 1 - x^2 - y^2 = 0. \end{aligned} \tag{17}$$

By solving Eq. (17) and eliminating λ, we can express y via

$$y = \frac{2G_{12}x^2 + G_{23}x - G_{12}}{(G_{11} - G_{22})x + G_{13}}. \tag{18}$$

By squaring both sides of Eq. (18), and substituting $y^2 = 1 - x^2$ into it, we finally obtain a fourth-order polynomial of the form

$$x^4F_4 + x^3F_3 + x^2F_2 + xF_1 + F_0 = 0, \tag{19}$$

where

$$\begin{aligned} F_4 &= 4G_{12}^2 + G_{22}^2 + G_{11}^2 - 2G_{11}G_{22} \\ F_3 &= 4G_{12}G_{23} + 2G_{11}G_{13} - 2G_{13}G_{22} \\ F_2 &= G_{23}^2 + 2G_{11}G_{22} + G_{13}^2 - 4G_{12}^2 - G_{11}^2 - G_{22}^2 \\ F_1 &= 2G_{13}G_{22} - 2G_{11}G_{13} - 2G_{12}G_{23} \\ F_0 &= G_{12}^2 - G_{13}^2. \end{aligned}$$

x can be easily solved from Eq. (19) by using the eigenvalue method [22]. After plugging x back into Eq. (18), y can also be determined. Eq. (19) has at most 2 minima, and up to eight minima can be obtained in our method (the seventh-order polynomial has 4 minima), which is significantly less than the DLS and OP*n*P method, which are 27 and 81 solutions, respectively. For each minimum, the coordinates of all points in the intermediate frame can be calculated via Eq. (5). Hence, the R and t of the camera with respect to the world frame are given by a standard 3D alignment scheme [24]. When $n \geq 6$, the P*n*P problem has a unique solution. Therefore, we choose the minimum with the least re-projection residual as the optimum of the solution. When $4 \leq n < 6$, the P*n*P problem has multiple solutions in general [26]. Therefore, we return all minima to the end solution.

2.4. Refining results using single Gauss–Newton iteration

In order to further improve the accuracy of the estimated results, we reformulate the pose estimation problem into a least-squares problem with three variables, then solve the least-squares problem via a single Gauss–Newton step.

The perspective model from the world frame to the normalized image plane can be expressed as

$$\lambda_i f_i = RW_i + t \qquad i = 1, 2, \ldots, n, \tag{20}$$

in which $t = [t_x, t_y, t_z]^T$. We adopt the Cayley parameterization to express the rotation R, which is given by

$$R = \frac{1}{H} \begin{bmatrix} \hat{m}_1^T \\ \hat{m}_2^T \\ \hat{m}_3^T \end{bmatrix} = \frac{1}{H} \begin{bmatrix} 1 + b^2 - c^2 - d^2 & 2bc - 2d & 2bd + 2c \\ 2bc + 2d & 1 - b^2 + c^2 - d^2 & 2cd - 2b \\ 2bd - 2c & 2cd + 2b & 1 - b^2 - c^2 + d^2 \end{bmatrix},$$

where $H = 1 + b^2 + c^2 + d^2$.

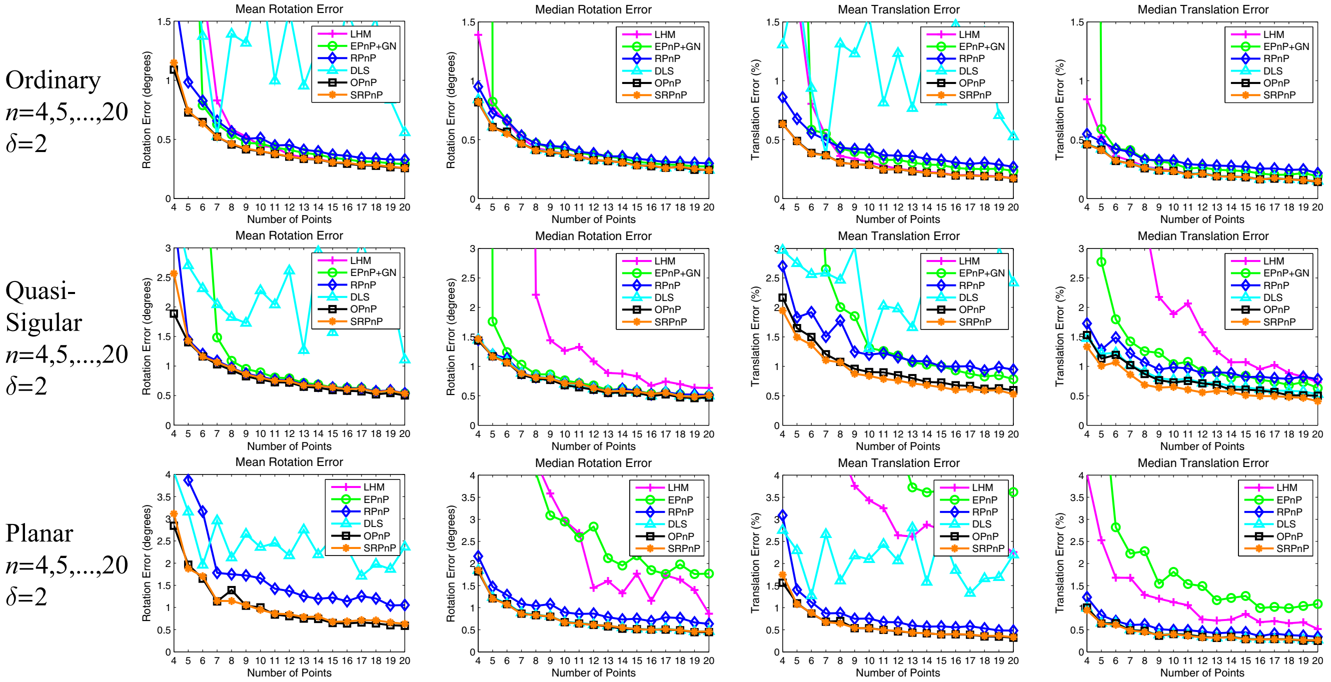


Fig. 2. The mean and median rotation and translation errors with the varying of point numbers.

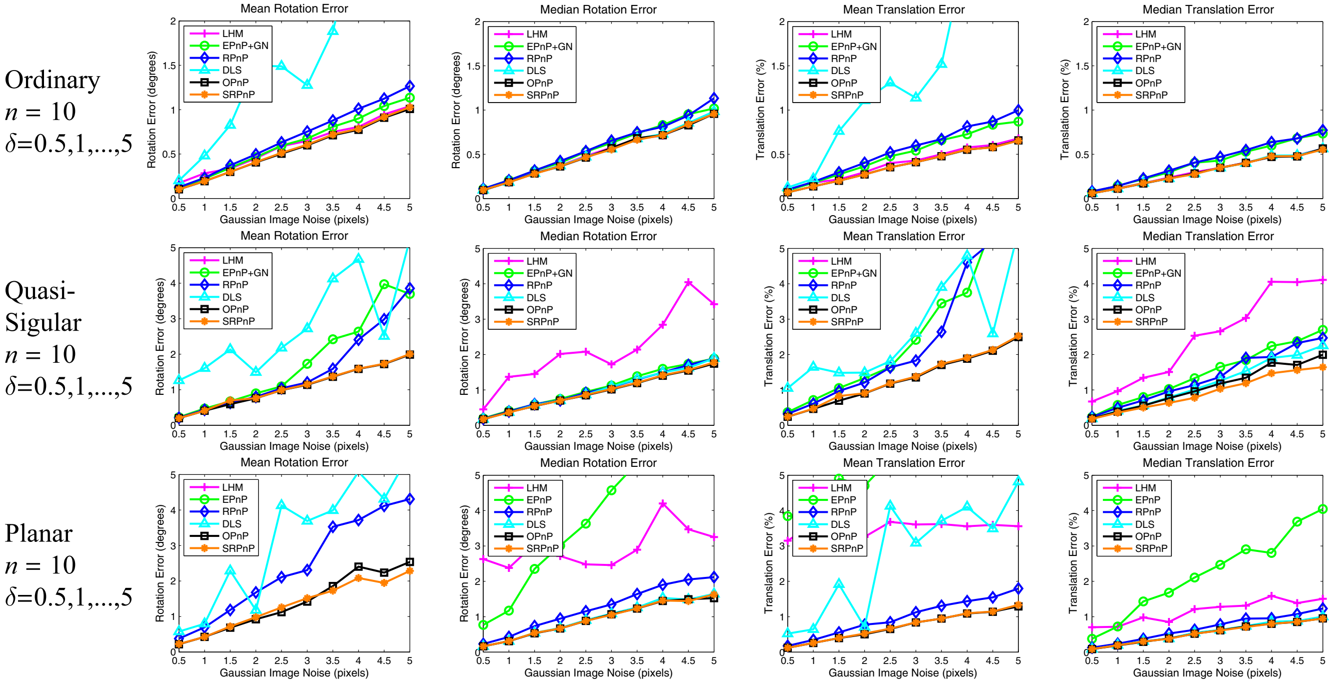


Fig. 3. The mean and median rotation and translation errors with the varying of noise levels.

We multiply H at both sides of Eq. (20), and obtain

$$\hat{\lambda}_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} \hat{m}_1^T \\ \hat{m}_2^T \\ \hat{m}_3^T \end{bmatrix} W_i + \begin{bmatrix} \hat{t}_x \\ \hat{t}_y \\ \hat{t}_z \end{bmatrix}, \qquad (21)$$

where $\hat{\lambda}_i = H\lambda_i$ and $[\hat{t}_x, \hat{t}_y, \hat{t}_z]^T = \hat{t} = Ht$.

Now letting $\hat{s} = [1, b, c, d, b^2, bc, bd, c^2, cd, d^2]^T$, and Eq. (21) can be further expressed as

$$\hat{t}_x - \hat{t}_z u_i = \hat{m}_3^T W_i u_i - \hat{m}_1^T W_i$$
$$\hat{t}_y - \hat{t}_z v_i = \hat{m}_3^T W_i v_i - \hat{m}_2^T W_i \qquad i = 1, 2, \ldots n$$

$$\iff A_i \hat{t} = \hat{B}_i \hat{s} \qquad i = 1, 2, \ldots, n,$$

where

$$A_i = \begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \end{bmatrix} \qquad (22)$$

and

$$\hat{B}_i = \begin{bmatrix} -X_i^w + Z_i^w u_i, & 2Y_i^w u_i, & -2Z_i^w - 2X_i^w u_i, & 2Y_i^w, & -X_i^w - Z_i^w u_i, & -2Y_i^w, & -2Z_i^w + 2X_i^w u_i, & X_i^w - Z_i^w u_i, & 2Y_i^w u_i, & Z_i^w u_i + X_i^w \\ -Y_i^w + Z_i^w v_i, & 2Y_i^w v_i + 2Z_i^w, & -2X_i^w v_i, & -2X_i^w, & Y_i^w - Z_i^w v_i, & -2X_i^w, & 2X_i^w v_i, & -Y_i^w - Z_i^w v_i, & -2Z_i^w + 2Y_i^w v_i, & Z_i^w v_i + Y_i^w \end{bmatrix}.$$

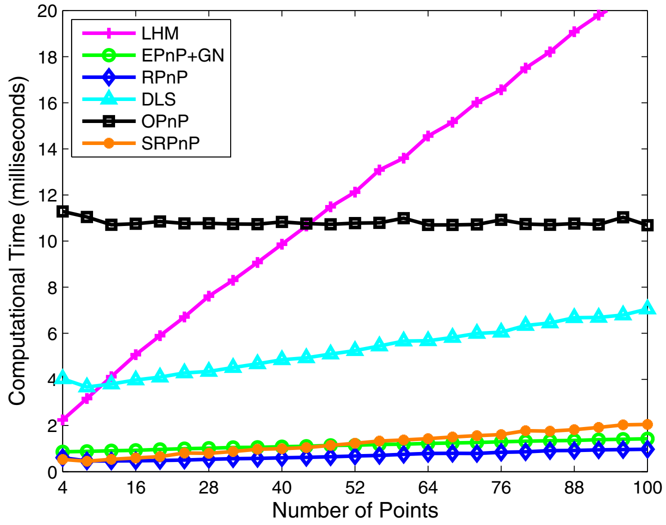


Fig. 4. The distribution of average running time.

Eq. (22) is also satisfied for every points, hence

$$\begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_n \end{bmatrix} \hat{t} = \begin{bmatrix} \hat{B}_1 \\ \hat{B}_2 \\ \vdots \\ \hat{B}_n \end{bmatrix} \hat{s} \Longleftrightarrow A\hat{t} = \hat{B}\hat{s} \Longleftrightarrow \hat{t} = \hat{C}\hat{s}, \tag{23}$$

where $\hat{C} = (A^T A)^{-1} A^T \hat{B}$.

Additionally, RW_i can be rewritten as

$$RW_i = \hat{Q}(W_i)\hat{s}, \tag{24}$$

where

$$\hat{Q}(W_i) = \begin{bmatrix} X_i^w, & 0, & 2Z_i^w, & -2Y_i^w, & X_i^w, & 2Y_i^w, & 2Z_i^w, & -X_i^w, & 0, & -X_i^w \\ Y_i^w, & -2Z_i^w, & 0, & 2X_i^w, & -Y_i^w, & 2X_i^w, & 0, & Y_i^w, & 2Z_i^w, & -Y_i^w \\ Z_i^w, & 2Y_i^w, & -2X_i^w, & 0, & -Z_i^w, & 0, & 2X_i^w, & -Z_i^w, & 2Y_i^w, & Z_i^w \end{bmatrix}.$$

Using Eqs. (22) and (24) in Eq. (20) results in

$$\hat{\lambda}_i e_i = \hat{M}_i \hat{s} \qquad i = 1, 2, \dots, n, \tag{25}$$

where $\hat{M}_i = \hat{Q}(W_i) + \hat{C}$.

After plugging $\hat{\lambda}_i = e_i^T \hat{M}_i \hat{s}$ back into Eq. (25), we finally obtain the least-squares problem as follows

$$\hat{\varepsilon} = \sum_{i=1}^{n} \| (e_i e_i^T \hat{M}_i - \hat{M}_i)\hat{s} \|^2 = \sum_{i=1}^{n} \| \hat{E}_i \hat{s} \|^2, \tag{26}$$

where $\hat{E}_i$ is a 3 × 10 matrix that can be computed ahead. We then use the typical Gauss–Newton method to solve the least-squares problem. Since the initialization is accurate enough, only one-step iteration is used.

3. Experiments with synthetic data

In this section, we investigated the performance of the proposed method, referred to as SRPnP,[1] by means of synthetic data, and compared the accuracy with the leading PnP methods:

- **LHM:** One of the best iterative methods, which is globally convergent in the ordinary case [15].
- **EP*n*P** (**EP*n*P+GN**)**:** The efficient non-iterative method, which achieves excellent results when $n \geq 6$ [12].
- **RP*n*P:** A robust non-iterative method, which works well for both non-redundant ($n \leq 5$) and redundant point sets [14].
- **DLS:** The direct least-squares method, which is the first method that computes all solutions of P*n*P in the general case [7].
- **OP*n*P:** The first non-iterative methods that is fast, generally applicable, and globally optimal, which represents the state-of-the-art solution [26].

All methods were implemented via MATLAB, and all codes are executed on a quad-core notebook with 2.5 GHz CPU and 4GB RAM. The source codes can be downloaded from http://pingwang.sxl.cn/.

3.1. Synthetic data

We synthesized a virtual perspective camera with an image size of 640 × 480 pixels, focal length of 800 pixels, and principle point at the image center. Next, we generated *n* 3D reference points in the camera frame, and transformed these 3D points into the world frame using the ground-truth of rotation R_{true} and translation T_{true}. Finally, we projected these 3D points into the 2D image plane using the virtual calibrated camera. Depending on the experiment, a different level of white Gaussian noise was added to the 2D image plane.

The accuracy of the PnP problem is closely related to the configuration of the 3D reference points, hence we test the proposed method at different configurations. Let a matrix $M = [W_1, W_2, \dots, W_n]^T$, where W_i is the 3D coordinate of the reference point and *n* is the size of the point set. According to the 3 × 3 matrix $M^T M$, the configuration of the reference points can be categorized into three groups:

(1) **Ordinary Case.** $rank(M^T M) = 3$ and the smallest eigenvalue of $M^T M$ is not close to zero. The reference points were distributed in the range [-2,2] × [-2,2] × [4,8].
(2) **Planar Case.** $rank(M^T M) = 2$. In this case, the reference points lay on a plane, and were distributed in the range [−2,2] × [−2,2] × [0,0].
(3) **Quasi-Singular Case.** $rank(M^T M) = 3$ and the ratio of the smallest eigenvalue to the largest one is very small (< 0.05). The reference points were distributed in the range [1,2] × [1,2] × [4,8].

The estimated rotation and translation were defined as *R* and *T*, respectively, and the errors of each were calculated as

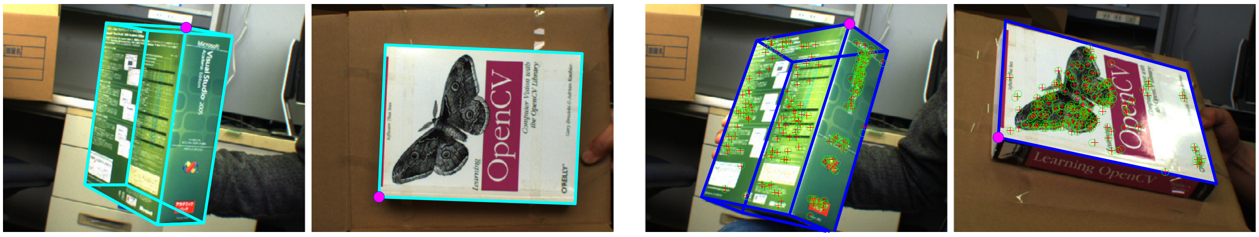
$$\begin{aligned} e_{rot}(degrees) &= \max_{k \in \{1,2,3\}} \cos^{-1}(r_{k,true}^T r_k) \times \frac{180}{\pi} \\ e_{trans}(\%) &= \frac{\| t_{true} - t \|}{\| t \|} \times 100 \end{aligned} \tag{27}$$

where $r_{k,\,true}$ and r_k are the *k*th column of R_{true} and *R*, respectively.

3.2. The effect with the varying number of points

The first simulated experiment investigated the performance of all methods with the varying number of points. We varied the point number *n* from 4 to 20, and add zero-mean Gaussian noise with fixed deviation $\delta = 2$ pixels onto the image projections. The results are shown in Fig. 2, and show that EP*n*P (EP*n*P+GN in the ordinary case) is not accurate enough, especially when *n* is small, due to its underlying linearization scheme. RP*n*P is also not accurate enough in most cases, because it is a suboptimal method. LHM is much less accurate in the quasi-singular and planar case. Besides, LHM tends to be inaccurate when no redundant points are available ($n \leq 5$), because of possible local optimum. The DLS method is not stable for the quasi-singular and planar case, due to the singularities of the Cayley parameterization. On the contrary,

[1] The source code of the proposed method (SRPnP) can be downloaded from http://pingwang.sxl.cn/.



(a) The reference images

(b) The input images

Fig. 5. Images that is augmented with the projected contour by using the estimated pose. The red marks "+" in the input images are the feature points matched with the reference image, and the green marks "o" are the re-projection of the features points using the estimated camera pose. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the SRP*n*P method offers accuracy comparable to the OP*n*P method for both planar and non-planar configurations, and its translation is slightly better than OP*n*P in the quasi-singular case.

3.3. The effect with the varying noise

The second simulated experiment tested the effects of noise on the accuracy of all methods. We fixed $n = 10$ and varied the noise deviation level δ from 0.5 to 5 pixels. The results are shown in Fig. 3. Similar results show that SRP*n*P and OP*n*P are comparable in all cases, which is still much better than other methods in terms of accuracy.

3.4. Computational efficiency

Fig. 4. shows the computational time with varying $4 \leq n \leq 100$ and fixed $\delta = 2$. As evident, our method has high efficiency, due to the fact that our method only needs to solve a seventh-order and fourth-order univariate polynomial. EP*n*P+GN and RP*n*P are faster than our method. However, our method is still very competitive, especially considering its high accuracy and easy implementation. All of these advantages indicate that our method is suitable for real tasks, especially when n is not extremely large. Interestingly, the running time of OP*n*P remains almost unchanged with increasing point numbers. This is because the Gröbner basis technique takes a lot of time. However, the remaining processes take very little time, due to using the vectorization technique, which can also be used in further work to improve the computational speed of our method. Faster performance can also be acquired for the SRP*n*P by using a C++ implementation, which will be published upon completion.

4. Experiments with real images

We repeat the experiment from [26] to test the proposed method. We first establish tentative correspondences by matching SIFT points between the input and reference image. After removing outliers by RANSAC, we recalculate the pose of the camera and augment the input image by using the projection of the model contour. As shown in Fig. 5, our method shows the similar and visually pleasing results.

5. Conclusions

In this paper, we developed a new method, which has high accuracy and efficiency, to determine the position and orientation of a calibrated camera by using n known 3D points and their image projections. The key process of our method is to solve univariate polynomials. The derivations of our method are easy to understand, and the final method is more efficient than existing direct minimization methods. The experiment results also demonstrated its superiority in accuracy, when compared with existing leading methods.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (No. 61473148). Ping Wang furthermore wants to thank Xinyu Kou for her encouragement, care and support over the passed years.

References

[1] Y.I. Abdel-Aziz, H.M. Karara, M. Hauck, Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry, Photogramm. Eng. Remote Sensing 81 (2) (2015) 103–107.

[2] A. Ansar, K. Daniilidis, Linear pose estimation from points or lines, Pattern Anal. Mach. Intell. IEEE Trans. 25 (5) (2003) 578–589.

[3] C.S. Chen, W.Y. Chang, On pose recovery for generalized visual sensors, IEEE Trans. Pattern Anal. Mach. Intell. 26 (7) (2004) 848.

[4] A.P. Dani, N.R. Fischer, W.E. Dixon, Single camera structure and motion, Autom. Control IEEE Trans. 57 (1) (2012) 238–243.

[5] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.

[6] S. Gai, E.J. Jung, B.J. Yi, Multi-group localization problem of service robots based on hybrid external localization algorithm with application to shopping mall environment, Intell. Service Robot. 9 (3) (2016) 257–275.

[7] J.A. Hesch, S.I. Roumeliotis, A direct least-squares (dls) method for pnp, in: IEEE International Conference on Computer Vision, 2011, pp. 383–390.

[8] S. Hijikata, K. Terabayashi, K. Umeda, A simple indoor self-localization system using infrared leds, in: Sixth International Conference on Networked Sensing Systems, 2009, pp. 1–7.

[9] D.G. Hull, Introduction to Optimization, A John Wiley and Sons, Inc., Publication, 2003.

[10] L. Kneip, H. Li, Y. Seo, UPnP: An Optimal O (n) Solution to the Absolute Pose Problem with Universal Applicability, 2014.

[11] Z. Kukelova, M. Bujnak, T. Pajdla, Automatic generator of minimal problem solvers, in: European Conference on Computer Vision, 2008, pp. 302–315.

[12] V. Lepetit, F. Moreno-Noguer, P. Fua, Epnp: an accurate o(n) solution to the pnp problem, Int J Comput Vis 81 (2) (2009) 155–166.

[13] S. Li, C. Xu, A stable direct solution of perspective-three-point problem, Int. J. Pattern Recognit Artif Intell. 25 (5) (2011) 627–642.

[14] S. Li, C. Xu, M. Xie, A robust o(n) solution to the perspective-n-point problem, IEEE Trans. Pattern Anal. Mach. Intell. 34 (7) (2012) 1444–1450.

[15] C.P. Lu, G.D. Hager, E. Mjolsness, Fast and globally convergent pose estimation from video images, IEEE Trans. Pattern Anal. Mach. Intell. 22 (6) (2000) 610–622.

[16] S. Malik, G. Roth, Robust 2d tracking for real-time augmented reality, in: Proceedings of the Vision Interface Conference, 2002.

[17] E. Marchand, H. Uchiyama, F. Spindler, Pose estimation for augmented reality: a hands-on survey, IEEE Trans. Vis. Comput. Graph. 22 (2016). 1–1

[18] A.I. Mourikis, N. Trawny, S.I. Roumeliotis, A.E. Johnson, A. Ansar, L. Matthies, Vision-aided inertial navigation for spacecraft entry, descent, and landing, Robot. IEEE Trans. 25 (2) (2009) 264–280.

[19] G. Nakano, A Versatile Approach for Solving PnP, PnPf, and PnPfr Problems, Springer International Publishing, 2016.

[20] A. Penate-Sanchez, J. Andrade-Cetto, F. Moreno-Noguer, Exhaustive linearization for robust camera pose and focal length estimation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (10) (2013) 2387–2400.

[21] L. Quan, Z. Lan, Linear n-point camera pose determination, IEEE Trans. Pattern Anal. Mach. Intell. 21 (8) (1999) 774–780.

[22] L.D. Roper, Numerical recipes: the art of scientific computing, Bull. Math. Biol. 49 (6) (1987) 761–762.

[23] J. Ryan, A. Hubbard, J. Box, J. Todd, P. Christoffersen, J. Carr, T. Holt, N. Snooke, Uav photogrammetry and structure from motion to assess calving dynamics at store glacier, a large outlet draining the greenland ice sheet, Cryosphere 9 (1) (2015) 1–11.

[24] S. Umeyama, Least-squares estimation of transformation parameters between two point patterns, IEEE Trans. Pattern Anal. Mach. Intell. 13 (4) (1991) 376–380.

[25] Y. Zheng, L. Kneip, A direct least-squares solution to the pnp problem with unknown focal length, in: Computer Vision and Pattern Recognition, 2016, pp. 1790–1798.

[26] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, M. Okutomi, Revisiting the pnp problem: a fast, general and optimal solution, in: IEEE International Conference on Computer Vision, 2013, pp. 2344–2351.