A Simple Direct Solution to the Perspective-Three-Point Problem

Gaku Nakano
g-nakano@cq.jp.nec.com

Central Research Labs
NEC Corporation
Kawasaki, Japan

Abstract

This paper proposes a new direct solution to the perspective-three-point (P3P) problem based on an algebraic approach. The proposed method represents the rotation matrix as a function of distances from the camera center to three 3D points, then, finds the distances by utilizing the orthogonal constraints of the rotation matrix. The formulation can be simply written because it relies only on some simple concepts of linear algebra. According to synthetic data evaluations, the proposed method gives the second-best performance against the state-of-the-art methods on both numerical accuracy and computational efficiency. In particular, the proposed method is the fastest among the quartic-equation based solvers. Moreover, the experimental results imply that the P3P problem still has an arguable issue on numerical stability regarding a point distribution and a camera pose.

1 Introduction

The perspective-three-point (P3P) problem, also known as the absolute camera pose estimation problem, is one of the most classical and fundamental problems in computer vision that determines the pose of a calibrated camera, *i.e*. the rotation and the translation, from three pairs of 3D point and its projection on the image plane. Since Grunert [7] gave the first solution in 1841, the P3P problem has been widely investigated [5, 6, 8] and extended to more complex camera pose estimation problems, *e.g*. for least squares case with n points (the PnP problem [8, 12, 14, 19]), for uncalibrated cameras with unknown internal parameters such as focal length or lens distortion (the P3.5P [17], P4P[1, 3], P5P [4], PnPf [15, 18], and PnPfr [15] problems).

Classical methods for the P3P problem [6, 8] consist of two steps: first, find the distances between the camera center and the given three 3D points; then, estimate the camera pose by solving an alignment problem of two triangles. The first step formulates a quartic equation with respect to one of the three distances by eliminating the other two based on the law of cosines. After finding the roots of the quartic equation, the second step solves the alignment problem, which is a rigid transformation between two triangles, by using a 4×4 eigenvalue decomposition or 3×3 singular value decomposition. Due to operations of the matrix decomposition, the numerical accuracy of the final solution becomes low despite its time-consuming processing.

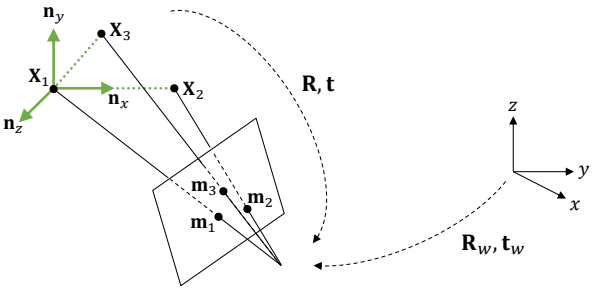


Figure 1: The perspective-three-point problem.

To overcome the above issue, direct methods, which do not require to solve the alignment problem, have been proposed. We can classify those direct P3P solvers into two categories depending on their mathematical derivation: geometric approaches and algebraic approaches. Based on geometric constrains, Kneip *et al*. [11] gave the first direct method that is faster and more numerically stable than a classical method by Gao *et al*. [6]. Inspired by the success of Kneip *et al*., Masselli and Zell [13] and Ke and Roumeliotis [10] also proposed a direct method by utilizing different geometric constraints, respectively. On the other hand, Banno [2] derived an algebraic approach that represents the camera pose as a linear combination of three nullspace vectors, whose coefficients coincide with the three distances. Persson and Nordberg [16] recently showed that the classical distance based equation can be solved by a cubic polynomial, which is an elliptic equation of rank 2.

Those direct methods achieve around 10^{-13} numerical error in a few microseconds of computation time on C++ implementation. One may think that the P3P problem has been improved enough on accuracy and efficiency for practical use. However, we believe that exploring more compact algorithms can contribute to not only the P3P problem itself but also other geometric problems in both theoretical and practical aspects because the P3P problem is one of the most basic problems in computer vision.

The new direct method proposed in this paper belongs to the second category, which is based on an algebraic approach. Similarly to Banno [2], we directly formulate the rotation matrix as a linear representation of the distances. However, our derivation is simpler that can be written in a closed form without using a computer algebra system, unlike Banno's method. Due to less computational operations, the proposed method outperforms the state-of-the-art methods solving a quartic equation in terms of computational efficiency by 12% to 50% while maintaining comparable numerical stability with the cubic-equation method.

2 Perspective-Three-Point (P3P) Problem

This section briefly describes the P3P problem, which is illustrated in Fig. 1. Let $\mathbf{X}_i$ be an i-th 3D point and $\mathbf{m}_i$ be the corresponding 2D point on an image projected by a calibrated camera with a rotation matrix $\mathtt{R}_w$ and a translation $\mathbf{t}_w$ with respect to the world coordinate system. A 2D point is represented as a 3×1 homogeneous vector such that $\|\mathbf{m}_i\| = 1$. The projective transformation between $\mathbf{X}_i$ and $\mathbf{m}_i$ can be written by

$$d_i \mathbf{m}_i = \mathtt{R}_w \mathbf{X}_i + \mathbf{t}_w, \tag{1}$$

where d_i represents the distance between the 3D point and the camera center. Note that d_i, also called the projective depth, is not always equivalent to the real distance because $\mathbf{m}_i$ is normalized and Eq. (1) holds with an arbitrarily scaling, *i.e.* $d_i\mathbf{m}_i = (d_i/s)(s\mathbf{m}_i)$. The P3P problem is to determine the rotation matrix $\mathtt{R}_w$ and the translation $\mathbf{t}_w$ if three point pairs, $\{\mathbf{X}_1, \mathbf{m}_1\}$, $\{\mathbf{X}_2, \mathbf{m}_2\}$, and $\{\mathbf{X}_3, \mathbf{m}_3\}$, are given.

3 Proposed Solution

3.1 P3P Problem in intermediate coordinate system

Without loss of generality, we can consider the P3P problem in an intermediate coordinate system centered on the given 3D points. For the new coordinate system, we assume a similar condition used in the existing methods of Kneip *et al.* [1] and Banno [2], where the first 3D point is set to be the new origin, the second 3D point is on the new x-axis, and all points lie on the same plane, $z = 0$. New 3D points $\mathbf{X}'_i$, $i \in \{1,2,3\}$, satisfying the above condition can be given by the following transformation:

$$\mathbf{X}'_i = \mathtt{N}^\mathsf{T}(\mathbf{X}_i - \mathbf{X}_1) \quad \rightarrow \quad \mathbf{X}'_1 = \begin{bmatrix}0\\0\\0\end{bmatrix}, \quad \mathbf{X}'_2 = \begin{bmatrix}a\\0\\0\end{bmatrix}, \quad \mathbf{X}'_3 = \begin{bmatrix}b\\c\\0\end{bmatrix}, \tag{2}$$

where

$$\begin{gathered}\mathtt{N} = [\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z], \quad \mathbf{n}_x = \frac{\mathbf{X}_2 - \mathbf{X}_1}{\|\mathbf{X}_2 - \mathbf{X}_1\|}, \quad \mathbf{n}_z = \frac{\mathbf{n}_x \times (\mathbf{X}_3 - \mathbf{X}_1)}{\|\mathbf{n}_x \times (\mathbf{X}_3 - \mathbf{X}_1)\|}, \quad \mathbf{n}_y = \mathbf{n}_z \times \mathbf{n}_x, \\ a = \|\mathbf{X}_2 - \mathbf{X}_1\|, \quad c > 0, \quad \sqrt{b^2 + c^2} = \|\mathbf{X}_3 - \mathbf{X}_1\|.\end{gathered} \tag{3}$$

Since $\mathtt{N}$ is an orthogonal matrix, *i.e.* $\mathtt{N}\mathtt{N}^\mathsf{T} = \mathtt{N}^\mathsf{T}\mathtt{N} = \mathtt{I}$, the rigid transformation between the new and the original coordinate systems can be written by

$$\begin{aligned}\mathtt{R}_w\mathbf{X}_i + \mathbf{t}_w &= (\mathtt{R}_w\mathtt{N})\left(\mathtt{N}^\mathsf{T}(\mathbf{X}_i - \mathbf{X}_1)\right) + (\mathbf{t}_w + \mathtt{R}_w\mathbf{X}_1) \\ &= \mathtt{R}\mathbf{X}'_i + \mathbf{t}.\end{aligned} \tag{4}$$

Hence, the new rotation matrix $\mathtt{R}$ and the translation $\mathbf{t}$ in the intermediate coordinate system are given by

$$\begin{aligned}\mathtt{R} &= \mathtt{R}_w\mathtt{N}, \\ \mathbf{t} &= \mathbf{t}_w + \mathtt{R}_w\mathbf{X}_1.\end{aligned} \tag{5}$$

Finally, using Eqs. (2) and (5), we can rewrite the P3P problem in a simple form:

$$d_1\mathbf{m}_1 = \mathbf{t}, \tag{6}$$
$$d_2\mathbf{m}_2 = a\mathbf{r}_1 + \mathbf{t}, \tag{7}$$
$$d_3\mathbf{m}_3 = b\mathbf{r}_1 + c\mathbf{r}_2 + \mathbf{t}, \tag{8}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ denote the first and the second column of $\mathtt{R}$, respectively.

3.2 Finding distances

Subtracting Eq. (6) from Eq. (7), we can eliminate $\mathbf{t}$ and obtain the following equation with respect to $\mathbf{r}_1$:

$$\begin{aligned}\mathbf{r}_1 &= \frac{1}{a}\left(d_2\,\mathbf{m}_2 - d_1\,\mathbf{m}_1\right)\\ &= \frac{1}{a}\mathtt{A}\,\mathbf{d},\end{aligned} \tag{9}$$

where $\mathtt{A} = [-\mathbf{m}_1, \mathbf{m}_2, \mathbf{0}]$ and $\mathbf{d} = [d_1, d_2, d_3]^\mathsf{T}$. Similarly, subtracting Eq. (6) from Eq. (8) and inserting Eq. (9) into that, we can also express $\mathbf{r}_2$ in a linear form:

$$\begin{aligned}\mathbf{r}_2 &= \frac{1}{c}\left(d_3\,\mathbf{m}_3 - d_1\,\mathbf{m}_1 - b\,\mathbf{r}_1\right)\\ &= \frac{1}{c}\left(\mathtt{B} - p\,\mathtt{A}\right)\mathbf{d}\\ &= \frac{1}{c}\mathtt{C}\,\mathbf{d},\end{aligned} \tag{10}$$

where $p = b/a$, $\mathtt{B} = [-\mathbf{m}_1, \mathbf{0}, \mathbf{m}_3]$, and $\mathtt{C} = \mathtt{B} - p\,\mathtt{A}$.

Let us consider to build a system of equations with respect to the distance $\mathbf{d}$ by using the orthogonality constrains of $\mathbf{r}_1$ and $\mathbf{r}_2$, *i.e.*

$$\mathbf{r}_1^\mathsf{T}\mathbf{r}_2 = 0, \tag{11}$$

$$\mathbf{r}_1^\mathsf{T}\mathbf{r}_1 - \mathbf{r}_2^\mathsf{T}\mathbf{r}_2 = 0. \tag{12}$$

Since the above Eqs. (11) and (12) have both a scale ambiguity, we then define two new variables, $x = d_2/d_1$ and $y = d_3/d_1$, to reduce one of the unknown parameters. Replacing $\mathbf{d}$ by $\boldsymbol{\lambda} = \mathbf{d}/d_1 = [1, x, y]^\mathsf{T}$, we can rewrite Eqs. (11) and (12) by

$$\mathbf{r}_1^\mathsf{T}\mathbf{r}_2 = \frac{1}{ac}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{C}\,\mathbf{d} \qquad \propto \boldsymbol{\lambda}^\mathsf{T}\left(p\,\mathtt{A}^\mathsf{T}\mathtt{A} - \mathtt{A}^\mathsf{T}\mathtt{B}\right)\boldsymbol{\lambda}, \tag{13}$$

$$\mathbf{r}_1^\mathsf{T}\mathbf{r}_1 - \mathbf{r}_2^\mathsf{T}\mathbf{r}_2 = \frac{1}{a^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\,\mathbf{d} - \frac{1}{c^2}\mathbf{d}^\mathsf{T}\mathtt{C}^\mathsf{T}\mathtt{C}\,\mathbf{d} \propto \boldsymbol{\lambda}^\mathsf{T}\left(q\mathtt{A}^\mathsf{T}\mathtt{A} - \mathtt{B}^\mathsf{T}\mathtt{B}\right)\boldsymbol{\lambda}, \tag{14}$$

where $\propto$ denotes equality up to scale and $q = (b^2 + c^2)/a^2$. Expanding Eqs. (13) and (14), we have two bivariate quadratic equations in x and y:

$$\begin{aligned} f(x,y) &= f_1 x^2 + f_2 xy \qquad\quad + f_4 x + f_5 y + f_6 = 0,\\ g(x,y) &= g_1 x^2 \qquad + g_3 y^2 + g_4 x + g_5 y + g_6 = 0,\end{aligned} \tag{15}$$

where

$$\begin{aligned} &f_1 = p, \quad f_2 = \mathbf{m}_2^\mathsf{T}\mathbf{m}_3, \quad f_4 = (1-2p)(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2), \quad f_5 = \mathbf{m}_1^\mathsf{T}\mathbf{m}_3, \quad f_6 = p - 1,\\ &g_1 = q, \quad g_3 = -1, \quad\;\; g_4 = -2q(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2), \quad\;\; g_5 = 2\,f_5, \quad\;\; g_6 = q - 1.\end{aligned} \tag{16}$$

Readers may refer to the Appendix for the details of Eqs. (13) – (16).

The coefficient of y^2 in $f(x,y)$ is zero, therefore, we can express y as a function of x in a rational form:

$$y = -\frac{f_1 x^2 + f_4 x + f_6}{f_2 x + f_5}. \tag{17}$$

Substituting Eq. (17) into $g(x, y)$ leads to a univariate quartic equation in x:

$$h(x) = h_1 x^4 + h_2 x^3 + h_3 x^2 + h_4 x + h_5, \tag{18}$$

where

$$\begin{aligned}
h_1 &= {f_2}^2 g_1 - {f_1}^2, \\
h_2 &= {f_2}^2 g_4 + 2 f_2 f_5 (g_1 - f_1) - 2 f_1 f_4, \\
h_3 &= {f_5}^2 (g_1 - 2 f_1) + 2 f_2 f_5 (g_4 - f_4) - 2 f_1 f_6 + {f_2}^2 g_6 - {f_4}^2, \\
h_4 &= {f_5}^2 (g_4 - 2 f_4) + 2 f_2 f_5 (g_6 - f_6) - 2 f_4 f_6, \\
h_5 &= {f_5}^2 (g_6 - 2 f_6) - {f_6}^2.
\end{aligned} \tag{19}$$

For finding roots of a quartic equation, we can employ a closed form solution, *e.g.* Ferrari's method or Descartes' method. As a result, up to four real roots for x are obtained from Eq. (18). Then, substituting x into Eq. (17) yields a corresponding value of y for each x. Optionally, we can perform root polishing by applying Newton's method to increase numerical accuracy of the roots obtained by the closed form solution. In this paper, we apply only one Newton's iteration on $f(x,y)$ and $g(x,y)$ in Eq. (15) for each pair of x and y. Extra computational cost by the root polishing will be discussed in Sec. 4.3.

3.3 Recovering pose in world coordinate system

Once x and y are obtained, the camera pose are easily computed. First, based on the unit vector constraint of the rotation matrix, *i.e.* $\|\mathbf{r}_1\| = \|\mathtt{A}\mathbf{d}/a\| = 1$, we recover the distance by scaling

$$\mathbf{d} = \frac{1}{s}\lambda, \quad s = \frac{1}{a}\|\mathtt{A}\lambda\|. \tag{20}$$

Then, we obtain the first and the second columns of the rotation matrix, $\mathbf{r}_1$ and $\mathbf{r}_2$, by substituting $\mathbf{d}$ into Eqs. (9) and (10), respectively. The third column of the rotation matrix can be computed as a cross product of $\mathbf{r}_1$ and $\mathbf{r}_2$. The translation $\mathbf{t}$ can be given from Eq. (6). Thus, we have the rotation matrix $\mathtt{R}$ and the translation $\mathbf{t}$ in the intermediate coordinate system by

$$\begin{aligned}
\mathtt{R} &= [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_1 \times \mathbf{r}_2], \\
\mathbf{t} &= d_1 \mathbf{m}_1.
\end{aligned} \tag{21}$$

Finally, by back-substituting $\mathtt{R}$ and $\mathbf{t}$ into Eq. (5), we can then recover the absolute camera pose with respect to the world coordiate system by

$$\begin{aligned}
\mathtt{R}_w &= \mathtt{R}\mathtt{N}^\mathsf{T}, \\
\mathbf{t}_w &= \mathbf{t} - \mathtt{R}_w \mathbf{X}_1.
\end{aligned} \tag{22}$$

The entire procedure of the proposed solution is summarized in Algorithm 1.

Algorithm 1 A simple direct P3P solver

Input: $\mathbf{X}_i$: 3D points, $\mathbf{m}_i$: normalized 2D points ($\|\mathbf{m}_i\| = 1$)
Output: $\mathtt{R}_w$: rotation matrix, $\mathbf{t}_w$: translation vector

compute $\mathtt{N}$ and $\mathbf{X}'_i \leftarrow \mathtt{N}^\mathsf{T}(\mathbf{X}_i - \mathbf{X}_1)$ ▷ Eq.(3)
compute p, q, $\mathtt{A}$, $\mathtt{B}$, $\mathtt{C}$ ▷ Eqs.(9) and (10)
compute coefficients of $f(x,y)$, $g(x,y)$, $h(x)$ ▷ Eqs.(16) and (19)
find roots of $h(x)$
for each $x \in$ four roots **do**
    $y \leftarrow -(f_1 x^2 + f_4 x + f_6)/(f_2 x + f_5)$
    apply Newton's method on $f(x,y)$ and $g(x,y)$
    $\boldsymbol{\lambda} \leftarrow [1, x, y]^\mathsf{T}$
    $s \leftarrow \|\mathtt{A}\boldsymbol{\lambda}\|/a, \quad \mathbf{d} \leftarrow \boldsymbol{\lambda}/s$
    $\mathbf{r}_1 \leftarrow \mathtt{A}\mathbf{d}/a, \quad \mathbf{r}_2 \leftarrow \mathtt{C}\mathbf{d}/c$
    $\mathtt{R} \leftarrow [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_1 \times \mathbf{r}_2], \quad \mathbf{t} \leftarrow d_1 \mathbf{m}_1$
    $\mathtt{R}_w \leftarrow \mathtt{R}\mathtt{N}^\mathsf{T}, \quad \mathbf{t}_w \leftarrow \mathbf{t} - \mathtt{R}\mathbf{X}_1$
end

3.4 Difference from Banno's method

Banno [2] also proposes a direct P3P method estimating a distance in the intermediate coordinate system. Although the formulations by this paper and Banno are not same each other, actually, they are mathematically equivalent.

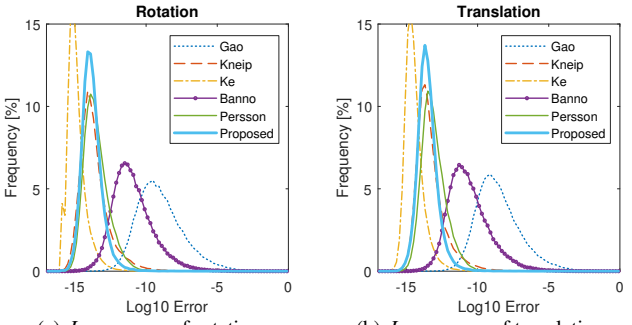
The essential difference is just to choose which distance is assumed to be 1; d_1 in this paper[1] and d_3 in Banno's method. This seems to be an insignificant matter, but it is not true. Since the third column of $\mathtt{A}$ and the second column of $\mathtt{B}$ are both zero vectors, we can reduce the number of terms in $f(x,y)$ and $g(x,y)$. For example, y^2 in $f(x,y)$ is eliminated in our formulation, Eq. (15). On the other hand, as shown in Eq. (24) in the Appendix, the coefficient of y^2 is not zero if $d_3 = 1$ is assumed. As a result, we can compute the coefficients of $h(x)$ by less numerical operations than Banno's method. For example, h_1 is given by $f_2^2 g_1 - f_1^2$ in the proposed method as shown in Eq. (19), but $f_4^2 g_1^2 + f_1^2 g_4^2 - f_2 f_4 g_1 g_2 + f_2^2 g_1 g_4 + \cdots$ in Banno's method (see Eq. (18) in [2]). This small difference greatly affects their performance on accuracy and speed, which will be demonstrated by experiments in Secs. 4.1 and 4.3, respectively.

4 Experiments

This section reports experimental results on synthetic data by comparing the proposed solution with the existing methods in terms of numerical stability, noise sensitivity, and computational efficiency.

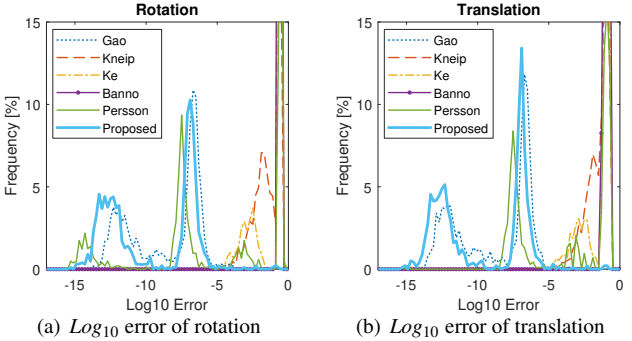
We implemented our proposed P3P solver on MATLAB. For the existing methods, we used a MATLAB code by Kneip [11][2], and faithfully ported C++ implementations to MAT-

[1] Choosing d_2 results in the same conclusion.
[2] https://dl.dropboxusercontent.com/u/23966023/home_page_files/p3p_code_final.zip



(a) Log_{10} error of rotation (b) Log_{10} error of translation

Figure 2: Histogram of numerical stability in a general configuration.



(a) Log_{10} error of rotation (b) Log_{10} error of translation

Figure 3: Histogram of numerical stability in a degenerate configuration.

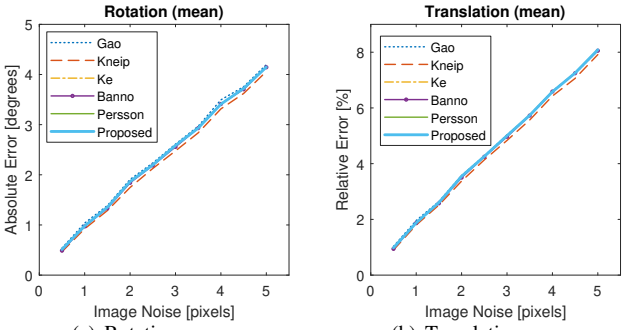
LAB for methods of Gao [8][3], Ke [10][3], Banno [2][4], and Persson [16][5]. Those original code except for Persson's method use different root finding methods for solving quartic equations; however, for a fair comparison, we employed the same solver, *i.e.* Ferrari's method included in Kneip's code.

For synthetic data generation, the camera properties were set to be the same condition as in Kneip [11]; $\mathtt{R}_w = diag([1,-1,-1])$, $\mathbf{t}_w = [0,0,6]^\mathsf{T}$, focal length $f = 800$, and principal point $(u_c, v_c) = (320, 240)$. Then, we calculated errors between the ground truth and estimated values by measuring the norm of the difference of quaternions for rotation and the relative difference for translation, respectively.

4.1 Numerical stability

We have tested the methods on two different configurations of a 3D point distribution to analyze numerical stability. For the first scene, we generated 1000 of 3D points distributed uniformly in the range of $[-2,2] \times [-2,2] \times [-2,2]$. Then, we randomly picked three points among them. For the second scene, we assumed a numerically degenerate case for Kneip's method[6], where three 3D points in the intermediate coordinate system were randomly gener-

[3] solveP3P function, https://docs.opencv.org/4.1.0/d9/d0c/group__calib3d.html
[4] https://github.com/atsuhikobanno/p3p
[5] https://github.com/midjji/lambdatwist-p3p
[6] The denominator of $\cot\alpha$ becomes zero. See Eq. (10) in [11].



(a) Rotation error (b) Translation error

Figure 4: Mean pose estimation error with respect to various image noise.

ated so that x coordinates of the second and the third points, $\mathbf{X}'_2$ and $\mathbf{X}'_3$, have both the same value, namely, $a = b$. In both scenes, selected 3D points were projected onto the image plane without adding any noise.

Figures 2 and 3 show a histogram of numerical error distribution over 10^5 trials for the first and the second scenes, respectively. In the first scene, Ke's method gave the highest stability with the peak value at 10^{-15} error, followed by the proposed method, Kneip's method, and Persson's method with the peak at 10^{-14} error. Although Ke's method is the most numerically accurate, the four methods are practically comparable because of the slight differences in the last few digits. By contrast, the state-of-the-art methods failed to compute true values in the second scene, where only the proposed and the classical Gao's methods can work with. We designed the second configuration mainly for Kneip's method, but Fig. 3 implies that Ke's, Banno's, and Persson's methods also have a similar numerical degeneracy in their algorithms.

In the second configuration, the camera direction is perpendicular to the plane on which $\mathbf{X}_{i\in\{1,2,3\}}$ lie under the experimental setting using $\mathtt{R}_w$ and $\mathbf{t}_w$. In other words, the z-axis of the camera and the normal vector of the plane are parallel. This configuration is not a rare case in practical situations, *e.g.* frontal observation of a square AR marker, which is known as one of severe conditions for camera pose estimation that causes to numerical instability. From this, we observed that the P3P problem still has an arguable issue on numerical stability regarding a point distribution and a camera pose.

4.2 Noise sensitivity

In this experiment, we set the first point configuration used in the previous section, and added zero-mean Gaussian noise onto projected image points. We evaluated the rotation and the translation errors by varying standard deviation of the noise from 0.5 to 5 pixels.

Figure 4 shows the mean error of 10^4 independent trials for each noise level. Kneip's method appears to be slightly better than the other methods, however, the difference can be negligibly small for practical use.

This result indicates that, whichever method is chosen, we can obtain almost similar performance in practical situations except for numerically degenerate configurations.

Table 1: Average computational time over 10^5 trials.

Method	Time [μsec]
Gao [6]	126.6
Kneip [11]	88.8
Ke [10]	72.1
Banno [2]	79.2
Persson [16]	51.6
Proposed	63.3

Table 2: Runtime profile of the proposed method. *Step*: processing step. *Line No.*: line numbers in Algorithm 1. *Time*: processing time in microseconds. *Rate*: percentage of the elapsed time for each step.

Step	Line No.	Time	Rate
Preprocess	1–3	14.5	23%
Root finding	4, 6	9.5	15%
Root polishing	7	8.9	14%
Recover pose	8–12	30.4	48%
Total		63.3	100%

4.3 Computational efficiency

We measured computational time by performing 10^5 independent trials on a Core i9-7920X PC. Table 1 shows the average value of the computational time for each method. The fastest is Persson's method that does not solve a quartic equation but a cubic equation. Among the methods solving a quartic equation, the proposed method outperforms the others: faster than Banno's method by 20%, Ke's method by 12%, Kneip's method by 28%, and Gao's method by 50%.

Table 2 shows a runtime profile of the proposed method. The most time consuming part is the final procedure for recovering the camera pose, which accounts for 48% of the total elapsed time. Since the time consumption by root polishing is relatively small, whether to perform root polishing is not a trade-off problem in our method.

5 Conclusion

In this paper, we have proposed a simple and efficient direct solution to the P3P problem based on an algebraic approach. The proposed method is mathematically similar to Banno's method, and can be interpreted as an optimization of Banno's method for reducing numerical operations. The actual difference between the proposed method and Banno's method seems to be small. However, unless the simplicity of our mathematical derivation, we cannot achieve the improvement on numerical accuracy and computational time as shown by the synthetic data evaluation. Through the experiments, we reported an interesting result that the state-of-the-art methods have a numerical degeneracy in their algorithm for a specific point-camera configuration. This result implies that the P3P problem still has an open issue on numerical stability.

Appendix

Derivation of $f(x,y)$:

$$\begin{aligned}\mathbf{r}_1^\mathsf{T}\mathbf{r}_2 &= \frac{1}{ac}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{C}\mathbf{d}\\ &= \frac{1}{ac}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}(\mathtt{B}-p\mathtt{A})\mathbf{d}\\ &\propto \boldsymbol{\lambda}^\mathsf{T}\left(p\mathtt{A}^\mathsf{T}\mathtt{A}-\mathtt{A}^\mathsf{T}\mathtt{B}\right)\boldsymbol{\lambda}\end{aligned} \tag{23}$$

$$\begin{aligned}f(x,y) &= \boldsymbol{\lambda}^\mathsf{T}\left(p\mathtt{A}^\mathsf{T}\mathtt{A}-\mathtt{A}^\mathsf{T}\mathtt{B}\right)\boldsymbol{\lambda}\\ &= [1,x,y]^\mathsf{T}\left(p\begin{bmatrix}1 & -\mathbf{m}_1^\mathsf{T}\mathbf{m}_2 & 0\\ -\mathbf{m}_1^\mathsf{T}\mathbf{m}_2 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}-\begin{bmatrix}1 & 0 & -\mathbf{m}_1^\mathsf{T}\mathbf{m}_3\\ -\mathbf{m}_1^\mathsf{T}\mathbf{m}_2 & 0 & \mathbf{m}_2^\mathsf{T}\mathbf{m}_3\\ 0 & 0 & 0\end{bmatrix}\right)\begin{bmatrix}1\\x\\y\end{bmatrix}\\ &= [1,x,y]^\mathsf{T}\begin{bmatrix}p-1 & -p(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2) & \mathbf{m}_1^\mathsf{T}\mathbf{m}_3\\ (1-p)(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2) & p & -\mathbf{m}_2^\mathsf{T}\mathbf{m}_3\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}1\\x\\y\end{bmatrix}\\ &= px^2+(\mathbf{m}_2^\mathsf{T}\mathbf{m}_3)xy+(1-2p)(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2)x+(\mathbf{m}_1^\mathsf{T}\mathbf{m}_3)y+p-1\end{aligned} \tag{24}$$

Derivation of $g(x,y)$:
From Eq. (23), we use $\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{C}\mathbf{d}=0 \leftrightarrow \mathbf{d}^\mathsf{T}\mathtt{B}^\mathsf{T}\mathtt{A}\mathbf{d}=p\,\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{d}$.

$$\begin{aligned}\mathbf{r}_1^\mathsf{T}\mathbf{r}_1-\mathbf{r}_2^\mathsf{T}\mathbf{r}_2 &= \frac{1}{a^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{d}-\frac{1}{c^2}\mathbf{d}^\mathsf{T}\mathtt{C}^\mathsf{T}\mathtt{C}\mathbf{d}\\ &= \frac{1}{a^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{d}-\frac{1}{c^2}\mathbf{d}^\mathsf{T}(\mathtt{B}-p\mathtt{A})^\mathsf{T}(\mathtt{B}-p\mathtt{A})\mathbf{d}\\ &= \frac{1}{a^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{d}-\frac{1}{c^2}\mathbf{d}^\mathsf{T}\mathtt{B}^\mathsf{T}\mathtt{B}\mathbf{d}+\frac{p}{c^2}\mathbf{d}^\mathsf{T}\mathtt{B}^\mathsf{T}\mathtt{A}\mathbf{d}+\frac{p}{c^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{C}\mathbf{d}\\ &= \frac{1}{a^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{d}-\frac{1}{c^2}\mathbf{d}^\mathsf{T}\mathtt{B}^\mathsf{T}\mathtt{B}\mathbf{d}+\frac{p^2}{c^2}\mathbf{d}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{d}\\ &= \frac{1}{c^2}\mathbf{d}^\mathsf{T}\left(\frac{b^2+c^2}{a^2}\mathtt{A}^\mathsf{T}\mathtt{A}-\mathtt{B}^\mathsf{T}\mathtt{B}\right)\mathbf{d}\\ &\propto \boldsymbol{\lambda}^\mathsf{T}\left(q\mathtt{A}^\mathsf{T}\mathtt{A}-\mathtt{B}^\mathsf{T}\mathtt{B}\right)\boldsymbol{\lambda}\end{aligned} \tag{25}$$

$$\begin{aligned}g(x,y) &= \boldsymbol{\lambda}^\mathsf{T}\left(q\mathtt{A}^\mathsf{T}\mathtt{A}-\mathtt{B}^\mathsf{T}\mathtt{B}\right)\boldsymbol{\lambda}\\ &= [1,x,y]^\mathsf{T}\left(q\begin{bmatrix}1 & -\mathbf{m}_1^\mathsf{T}\mathbf{m}_2 & 0\\ -\mathbf{m}_1^\mathsf{T}\mathbf{m}_2 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}-\begin{bmatrix}1 & 0 & -\mathbf{m}_1^\mathsf{T}\mathbf{m}_3\\ 0 & 0 & 0\\ -\mathbf{m}_1^\mathsf{T}\mathbf{m}_3 & 0 & 1\end{bmatrix}\right)\begin{bmatrix}1\\x\\y\end{bmatrix}\\ &= [1,x,y]^\mathsf{T}\begin{bmatrix}q-1 & -q(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2) & \mathbf{m}_1^\mathsf{T}\mathbf{m}_3\\ -q(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2) & q & 0\\ \mathbf{m}_1^\mathsf{T}\mathbf{m}_3 & 0 & -1\end{bmatrix}\begin{bmatrix}1\\x\\y\end{bmatrix}\\ &= qx^2-y^2-2q(\mathbf{m}_1^\mathsf{T}\mathbf{m}_2)x+2(\mathbf{m}_1^\mathsf{T}\mathbf{m}_3)y+q-1\end{aligned} \tag{26}$$

References

[1] Mongi A. Abidi and T Chandra. A new efficient and direct solution for pose estimation using quadrangular targets: Algorithm and evaluation. *IEEE transactions on pattern analysis and machine intelligence*, 17(5):534–538, 1995.

[2] Atsuhiko Banno. A p3p problem solver representing all parameters as a linear combination. *Image and Vision Computing*, 70:55–62, 2018.

[3] Martin Bujnak, Zuzana Kukelova, and Tomas Pajdla. A general solution to the p4p problem for camera with unknown focal length. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[4] Martin Bujnak, Zuzana Kukelova, and Tomas Pajdla. New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. In *Asian Conference on Computer Vision*, pages 11–24. Springer, 2010.

[5] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[6] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003.

[7] Johann August Grunert. Das pothenotische problem in erweiterter gestalt nebst bber seine anwendungen in der geodasie. *Grunerts Archiv fur Mathematik und Physik*, pages 238–248, 1841.

[8] Bert M Haralick, Chung-Nan Lee, Karsten Ottenberg, and Michael Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *International journal of computer vision*, 13(3):331–356, 1994.

[9] Joel A Hesch and Stergios I Roumeliotis. A direct least-squares (dls) method for pnp. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 383–390. IEEE, 2011.

[10] Tong Ke and Stergios I Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7225–7233, 2017.

[11] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2969–2976. IEEE, 2011.

[12] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155–166, 2009.

[13] Andreas Masselli and Andreas Zell. A new geometric approach for faster solving the perspective-three-point problem. In *2014 22nd International Conference on Pattern Recognition*, pages 2119–2124. IEEE, 2014.

[14] Gaku Nakano. Globally optimal dls method for pnp problem with cayley parameterization. In *BMVC*, pages 78–1, 2015.

[15] Gaku Nakano. A versatile approach for solving pnp, pnpf, and pnpfr problems. In *European Conference on Computer Vision*, pages 338–352. Springer, 2016.

[16] Mikael Persson and Klas Nordberg. Lambda twist: an accurate fast robust perspective three point (p3p) solver. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–332, 2018.

[17] Changchang Wu. P3. 5p: Pose estimation with unknown focal length. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2440–2448, 2015.

[18] Yinqiang Zheng and Laurent Kneip. A direct least-squares solution to the pnp problem with unknown focal length. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1790–1798, 2016.

[19] Yinqiang Zheng, Yubin Kuang, Shigeki Sugimoto, Kalle Astrom, and Masatoshi Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2344–2351, 2013.