A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation

Laurent Kneip
laurent.kneip@mavt.ethz.ch

Davide Scaramuzza
davide.scaramuzza@mavt.ethz.ch

Roland Siegwart
rsiegwart@ethz.ch

Autonomous Systems Lab, ETH Zurich

Abstract

The Perspective-Three-Point (P3P) problem aims at determining the position and orientation of the camera in the world reference frame from three 2D-3D point correspondences. This problem is known to provide up to four solutions that can then be disambiguated using a fourth point. All existing solutions attempt to first solve for the position of the points in the camera reference frame, and then compute the position and orientation of the camera in the world frame, which alignes the two point sets. In contrast, in this paper we propose a novel closed-form solution to the P3P problem, which computes the aligning transformation directly in a single stage, without the intermediate derivation of the points in the camera frame. This is made possible by introducing intermediate camera and world reference frames, and expressing their relative position and orientation using only two parameters. The projection of a world point into the parametrized camera pose then leads to two conditions and finally a quartic equation for finding up to four solutions for the parameter pair. A subsequent backsubstitution directly leads to the corresponding camera poses with respect to the world reference frame. We show that the proposed algorithm offers accuracy and precision comparable to a popular, standard, state-of-the-art approach but at much lower computational cost (15 times faster). Furthermore, it provides improved numerical stability and is less affected by degenerate configurations of the selected world points. The superior computational efficiency is particularly suitable for any RANSAC-outlier-rejection step, which is always recommended before applying PnP or non-linear optimization of the final solution.

1. Introduction

The Perspective-n-Point (PnP) problem is originated from camera calibration [1, 10, 17, 28]. Also known as pose estimation, it aims at retrieving the position and orientation of the camera with respect to a scene object from n corresponding 3D points. This problem has found many applications in computer animation [30], computer vision [16], augmented reality, automation, image analysis, automated cartography [10], photogrammetry [1, 24], robotics [35], and model-based machine vision systems [34]. In 1981, Fischler and Bolles [10] summarized the problem as follows: *Given the relative spatial locations of n control points, and given the angle to every pair of control points P_i from an additional point called the center of perspective C, find the lengths of the line segments joining C to each of the control points*. The next step then consists of retrieving the orientation and translation of the camera with respect to the object reference frame.

The Direct Linear Transformation was first developed by photogrammetrists [31] as a solution to the PnP problem—when the 3D points are in a general configuration—and then introduced in the computer vision community [7, 16]. When the points are coplanar, the homography transformation can be exploited [16] instead.

In this paper, we address the particular case of PnP for $n = 3$. This problem is also known as Perspective-Three-Point (P3P) problem. The P3P is the smallest subset of control points that yields a finite number of solutions. When the intrinsic camera parameters are known and we have $n \geq 4$ points, the solution is generally unique.

The P3P problem was first investigated in 1841 by Grunert [14] and in 1903 by Finsterwalder [8], who noticed that for a calibrated camera there can be up to four solutions, which can then be disambiguated using a fourth point. In the literature, there exist many solutions to this problem, which can be classified into iterative, non-iterative, linear, and non-linear ones. In 1991, Haralick et al. [15] reviewed the major direct solutions up to 1991, including the six algorithms given by Grunert (1841) [14], Finsterwalder (1903)—as summarized by Finsterwalder and Scheufele in [8]—, Merritt (1949) [25], Fischler and Bolles (1981) [10], Hung et al. (1985) [20], Linnainmaa et al. (1988) [23], and Grafarend et al. (1989) [13], respectively. They also gave the analytical solution for the P3P problem with resultant computation. Different solutions to the P3P problem have been later proposed by Quan and Lan (1999) [28] and Gao et al. (2003) [12]. A different approach—but for non-single-viewpoint cameras—was proposed by Nister and Stewenius in 2006 [27].

It is important to remark here that P3P is the most basic case of the PnP problem. All PnP problems include the P3P problem as a special case. Among those that handle arbitrary values of n are those of Fischler and Bolles (1981) [10], Dhome et al. (1989) [6], Horaud et al. (1989) [17], Haralick et al. (1991) [15], DeMenthon and Davis (1995) [4, 5], Quan and Lan (1999) [28], Triggs (1999) [32], Fiore (2001) [9], Ansar and Daniilidis (2003) [2], and Lepetit et al. (2009) [22]—this last one, in particular, also works for deformable objects.

Applications such as feature-point-based camera tracking [29, 21], structure from motion, and visual odometry [26] require dealing with hundreds or even thousands of noisy feature points and outliers in real-time, which requires computationally efficient methods. The standard approach consists of first using P3P in a RANSAC scheme [10]—in order to remove the outliers—and then PnP on all remaining inliers. If necessary, a further non-linear optimization can also be applied to refine the final solution.

All existing P3P algorithms cited above first estimate the distances $\|CP_i\|$ between the camera center C and the 3D points P_i from constraints given by the triangles CP_iP_j (see Fig. 1). Once the distances are known, the P_i are expressed in the camera frame as P_i^ν. Then, the orientation and translation $[R|t]$ of the camera in the world reference frame is taken to be the transformation that aligns the points P_i on P_i^ν and can be found in closed-form solution using quaternions [18] or singular value decomposition (SVD) [3, 19, 33, 11]. Particularly in RANSAC, the transformation into the world reference frame is a necessary step as it allows us to compute the camera projection matrix, which is then used—in combination with the reprojection error—to validate the RANSAC hypotheses.

In contrast to all previous approaches, in this paper we provide a closed-form solution for the P3P problem, which computes directly the position and orientation (i.e., $[R|t]$) of the camera in the world reference frame as a function of the image coordinates and the coordinates of the reference points in the world frame. To the best of our knowledge, this is the first work in this endeavor. The performance of the proposed algorithm will be evaluated against Gao-et-al.'s [12] implementation, which is one of the most popular and robust P3P solvers. The main advantage of the direct computation of $[R|t]$ is its superior computational efficiency. In the first stage, we avoid determining the points in the camera reference frame, and in the second stage, the aligning transformation—which would require SVD [33, 11]. As we will show in the results section, our algorithm is 15 times faster than Gao's and requires only 2 microseconds on a 2.8Ghz Dual Core laptop, which scales very well for RANSAC implementations. The second advantage is its superior numerical stability and robustness with respect to Gao's solution.

The structure of the paper is as follows. Section 2 presents the derivations that lead to the new solution of the P3P algorithm for retrieving the camera position and orientation directly. Section 3 provides a thorough analysis of the algorithm's performance, including numerical stability, computational cost, accuracy, and precision. The results will be compared to Gao's implementation [12]. Section 4, finally, concludes the work.

2. Theory

We consider the problem illustrated in Fig. 1. The goal is to find the exact position C and orientation matrix R of a camera with respect to the world frame (O, X, Y, Z), under the condition that the absolute spatial coordinates of three observed feature points P_1, P_2, and P_3 are given. We furthermore assume that the intrinsic camera parameters are known. Hence, we can assume that the unitary vectors $\vec{f_1}$, $\vec{f_2}$, and $\vec{f_3}$—pointing towards the three considered feature points from the camera frame—are given.

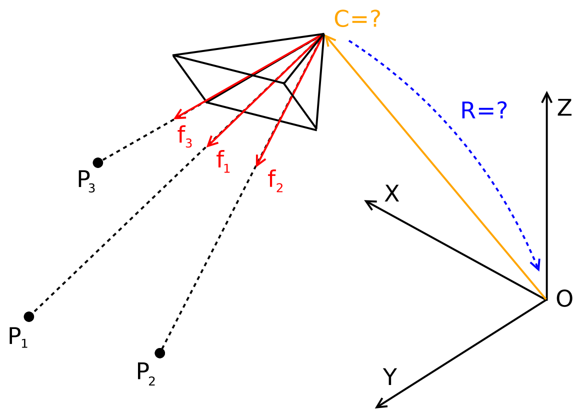

Figure 1. Synopsis of the problem.

Let us denote the original camera frame with ν. The first step involves the definition of a new, intermediate camera frame τ from the feature vectors $\vec{f_1}$ and $\vec{f_2}$ inside ν. As shown in Fig. 2, the new camera frame is defined as $\tau = (C, \vec{t}_x, \vec{t}_y, \vec{t}_z)$, where

$$\begin{aligned} \vec{t}_x &= \vec{f_1} \\ \vec{t}_z &= \frac{\vec{f_1} \times \vec{f_2}}{||\vec{f_1} \times \vec{f_2}||} \\ \vec{t}_y &= \vec{t}_z \times \vec{t}_x. \end{aligned}$$

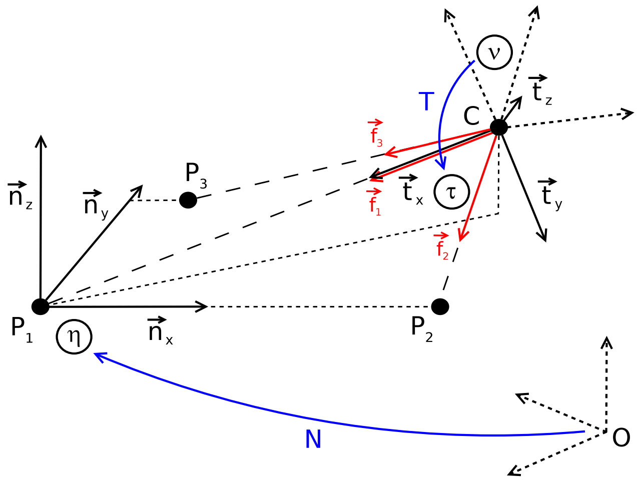

Figure 2. Illustration of the intermediate camera frame $\tau = (C, \vec{t}_x, \vec{t}_y, \vec{t}_z)$ and the intermediate world frame $\eta = (P_1, \vec{n}_x, \vec{n}_y, \vec{n}_z)$.

Via the transformation matrix $T = [\vec{t}_x, \vec{t}_y, \vec{t}_z]^T$, feature vectors can then be transformed into τ using

$$\vec{f}_i^\tau = T \cdot \vec{f}_i. \tag{1}$$

If we are able to define the orientation of τ with respect to the world frame, the orientation of ν is obviously also given using T.

The second step involves the definition of a new world frame η from the world points P_1, P_2, and P_3. The new spatial frame is defined as $\eta = (P_1, \vec{n}_x, \vec{n}_y, \vec{n}_z)$, where

$$\begin{aligned}\vec{n}_x &= \frac{\overrightarrow{P_1P_2}}{\|\overrightarrow{P_1P_2}\|}\\ \vec{n}_z &= \frac{\vec{n}_x \times \overrightarrow{P_1P_3}}{\|\vec{n}_x \times \overrightarrow{P_1P_3}\|}\\ \vec{n}_y &= \vec{n}_z \times \vec{n}_x.\end{aligned}$$

Via the transformation matrix $N = [\vec{n}_x, \vec{n}_y, \vec{n}_z]^T$, world points can finally be transformed into η using

$$P_i^\eta = N \cdot (P_i - P_1). \tag{2}$$

Again, if we are able to define the orientation of τ with respect to η, the orientation of τ is given automatically inside the world frame via N, and thus via T also the orientation of ν. A similar matter accounts for the camera center C that is—if defined inside η—recovered inside the world frame via a straightforward linear transformation. The resulting situation is illustrated in Fig. 2. The condition of existance of η is that P_1, P_2, and P_3 are not colinear. This can be easily avoided by verifying that $\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3}$ is not zero.

In the following, we will focus on the transformation between η and τ. We define the semi-plane Π that contains points P_1, P_2, and C, and hence also the unitary vectors $\vec{n}_x$, $\vec{t}_x$, $\vec{t}_y$, $\vec{f}_1$, and $\vec{f}_2$, as shown in Fig. 3. Points P_1, P_2, and C form a triangle of which two parameters are known, namely the distance d_{12} between P_1 and P_2, and the angle β between $\vec{f}_1$ and $\vec{f}_2$. The latter can be easily obtained via the dot-product $\cos\beta = \vec{f}_1 \cdot \vec{f}_2$. Since the later parametrization will only depend on $\cot\beta$, we define

$$b = \cot\beta = \pm\sqrt{\frac{1}{1-\cos^2\beta} - 1} = \pm\sqrt{\frac{1}{1-(\vec{f}_1 \cdot \vec{f}_2)^2} - 1}. \tag{3}$$

The sign of b is given by the sign of $\cos\beta$. We define the free parameter $\alpha \in [0;\pi]$ as the angle $\angle P_2P_1C$. Using the sine-law, we obtain

$$\frac{\|\overrightarrow{CP_1}\|}{d_{12}} = \frac{\sin(\pi-\alpha-\beta)}{\sin\beta}.$$

The position of the camera center C inside the plane Π is then given by

$$\begin{aligned}C^\Pi(\alpha) &= \begin{pmatrix}\cos\alpha \cdot \|\overrightarrow{CP_1}\|\\ \sin\alpha \cdot \|\overrightarrow{CP_1}\|\\ 0\end{pmatrix}\\ &= \begin{pmatrix}d_{12}\cos\alpha\sin(\alpha+\beta)\sin^{-1}\beta\\ d_{12}\sin\alpha\sin(\alpha+\beta)\sin^{-1}\beta\\ 0\end{pmatrix}\\ &= \begin{pmatrix}d_{12}\cos\alpha(\sin\alpha\cot\beta+\cos\alpha)\\ d_{12}\sin\alpha(\sin\alpha\cot\beta+\cos\alpha)\\ 0\end{pmatrix}\end{aligned}$$

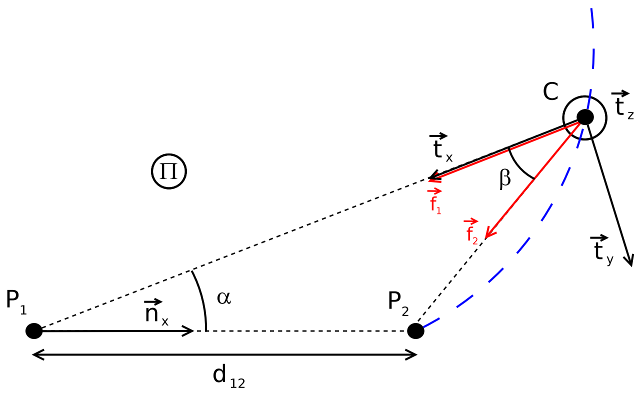

Figure 3. Semi-plane Π containing the triangle (P_1, P_2, C). The blue trajectory indicates the possible locations of the camera centre C depending on the free parameter α, and the fixed parameters d_{12} and β.

$$\Rightarrow C^{\Pi}(\alpha) = \begin{pmatrix} d_{12}\cos\alpha(\sin\alpha\cdot b+\cos\alpha) \\ d_{12}\sin\alpha(\sin\alpha\cdot b+\cos\alpha) \\ 0 \end{pmatrix}. \quad (4)$$

The basis vectors of τ inside Π are easily given with $\vec{t}_x^{\Pi} = (-\cos\alpha, -\sin\alpha, 0)^T$, $\vec{t}_y^{\Pi} = (\sin\alpha, -\cos\alpha, 0)^T$, and $\vec{t}_z^{\Pi} = (0,0,1)^T$.

In order to have C, $\vec{t}_x$, $\vec{t}_y$, and $\vec{t}_z$ expressed inside η, we need to take into account a second free parameter, namely the rotation θ of Π around $\vec{n}_x$, as illustrated in Fig. 4. The corresponding rotation matrix is given by

$$R_\theta = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}.$$

Note that $\theta \in [0;\pi]$ if $f_{3,z}^{\tau} < 0$, and $\theta \in [-\pi;0]$ if $f_{3,z}^{\tau} > 0$, where $\vec{f}_3^{\tau}$ is obtained from $\vec{f}_3$ via (1). This constraint is given very intuitively by the condition that $\vec{f}_3$ and P_3 need to lie on the same side of Π. It follows that the camera center C inside η is given with

$$\begin{aligned} C^{\eta}(\alpha,\theta) &= R_\theta \cdot C^{\Pi} \\ &= \begin{pmatrix} d_{12}\cos\alpha(\sin\alpha\cdot b+\cos\alpha) \\ d_{12}\sin\alpha\cos\theta(\sin\alpha\cdot b+\cos\alpha) \\ d_{12}\sin\alpha\sin\theta(\sin\alpha\cdot b+\cos\alpha) \end{pmatrix}, \end{aligned} \quad (5)$$

and the transformation matrix from η to τ is given by

$$\begin{aligned} Q(\alpha,\theta) &= \left[R_\theta \cdot \left(\vec{t}_x^{\Pi}\vec{t}_y^{\Pi}\vec{t}_z^{\Pi}\right)\right]^T \\ &= \begin{pmatrix} -\cos\alpha & -\sin\alpha\cos\theta & -\sin\alpha\sin\theta \\ \sin\alpha & -\cos\alpha\cos\theta & -\cos\alpha\sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix}. \end{aligned} \quad (6)$$

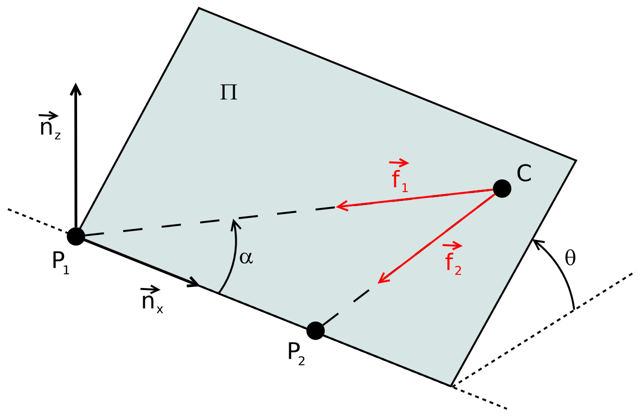

Figure 4. Rotation of the plane Π around $\vec{n}_x$ by the angle θ.

The two conditions for finding the correct values of the parameters α and θ are then established by transforming the third point P_3^{η} into τ, and imposing that the direction of this point is equal to the one of $\vec{f}_3^{\tau}$. Respecting that $P_3^{\eta} = (p_1, p_2, 0)^T$, we obtain

$$P_3^{\tau} = Q(\alpha,\theta)\cdot(P_3^{\eta} - C^{\eta}(\alpha,\theta))$$

$$= \begin{pmatrix} -\cos\alpha\cdot p_1 - \sin\alpha\cos\theta\cdot p_2 + d_{12}(\sin\alpha\cdot b+\cos\alpha) \\ \sin\alpha\cdot p_1 - \cos\alpha\cos\theta\cdot p_2 \\ -\sin\theta\cdot p_2 \end{pmatrix}. \quad (7)$$

After defining

$$\phi_1 = \frac{f_{3,x}^{\tau}}{f_{3,z}^{\tau}} \text{ and } \phi_2 = \frac{f_{3,y}^{\tau}}{f_{3,z}^{\tau}}, \quad (8)$$

the two conditions finally result in

$$\begin{aligned} & \begin{cases} \phi_1 = \frac{P_{3,x}^{\tau}}{P_{3,z}^{\tau}} \\ \phi_2 = \frac{P_{3,y}^{\tau}}{P_{3,z}^{\tau}} \end{cases} \\ \Leftrightarrow \quad & \begin{cases} \phi_1 = \frac{-\cos\alpha\cdot p_1 - \sin\alpha\cos\theta\cdot p_2 + d_{12}(\sin\alpha\cdot b+\cos\alpha)}{-\sin\theta\cdot p_2} \\ \phi_2 = \frac{\sin\alpha\cdot p_1 - \cos\alpha\cos\theta\cdot p_2}{-\sin\theta\cdot p_2} \end{cases} \\ \Leftrightarrow \quad & \begin{cases} \frac{\sin\theta}{\sin\alpha}p_2 = \frac{-\cot\alpha\cdot p_1 - \cos\theta\cdot p_2 + d_{12}(b+\cot\alpha)}{-\phi_1} \\ \frac{\sin\theta}{\sin\alpha}p_2 = \frac{p_1 - \cot\alpha\cos\theta\cdot p_2}{-\phi_2} \end{cases} \\ \Rightarrow \quad & \cot\alpha = \frac{\frac{\phi_1}{\phi_2}p_1 + \cos\theta\cdot p_2 - d_{12}\cdot b}{\frac{\phi_1}{\phi_2}\cos\theta\cdot p_2 - p_1 + d_{12}}. \end{aligned} \quad (9)$$

Furthermore, we have

$$\begin{aligned} & \phi_2 = \frac{P_{3,y}^{\tau}}{P_{3,z}^{\tau}} \\ \Leftrightarrow \quad & \sin^2\theta\cdot f_2^2 p_2^2 = \sin^2\alpha(p_1 - \cot\alpha\cos\theta\cdot p_2)^2 \\ \Leftrightarrow \quad & (1-\cos^2\theta)(1+\cot^2\alpha)f_2^2 p_2^2 \\ & \quad = p_1^2 - 2\cot\alpha\cos\theta\cdot p_1 p_2 + \cot^2\alpha\cos^2\theta\cdot p_2^2. \end{aligned} \quad (10)$$

Replacing (9) in (10), expanding, and collecting then easily leads to a fourth order polynomial of the form

$$a_4\cdot\cos^4\theta + a_3\cdot\cos^3\theta + a_2\cdot\cos^2\theta + a_1\cdot\cos\theta + a_0 = 0, \quad (11)$$

where,

$$
\begin{aligned}
a_4 &= -\phi_2^2 p_2^4 - \phi_1^2 p_2^4 - p_2^4 \\
a_3 &= 2p_2^3 d_{12} b + 2\phi_2^2 p_2^3 d_{12} b - 2\phi_1 \phi_2 p_2^3 d_{12} \\
a_2 &= -\phi_2^2 p_1^2 p_2^2 - \phi_2^2 p_2^2 d_{12}^2 b^2 - \phi_2^2 p_2^2 d_{12}^2 + \phi_2^2 p_2^4 \\
&\quad +\phi_1^2 p_2^4 + 2p_1 p_2^2 d_{12} + 2\phi_1 \phi_2 p_1 p_2^2 d_{12} b \\
&\quad -\phi_1^2 p_1^2 p_2^2 + 2\phi_2^2 p_1 p_2^2 d_{12} - p_2^2 d_{12}^2 b^2 - 2p_1^2 p_2^2 \\
a_1 &= 2p_1^2 p_2 d_{12} b + 2\phi_1 \phi_2 p_2^3 d_{12} \\
&\quad -2\phi_2^2 p_2^3 d_{12} b - 2p_1 p_2 d_{12}^2 b \\
a_0 &= -2\phi_1 \phi_2 p_1 p_2^2 d_{12} b + \phi_2^2 p_2^2 d_{12}^2 + 2p_1^3 d_{12} \\
&\quad -p_1^2 d_{12}^2 + \phi_2^2 p_1^2 p_2^2 - p_1^4 - 2\phi_2^2 p_1 p_2^2 d_{12} \\
&\quad +\phi_1^2 p_1^2 p_2^2 + \phi_2^2 p_2^2 d_{12}^2 b^2.
\end{aligned}
$$

Up to four real solutions for $\cos\theta$ are then obtained by simply applying Ferrari's closed form solution for finding the roots of a fourth order polynomial. Via replacement in (9), each value for $\cos\theta$ will then also lead to exactly one value for $\cot\alpha$. Each real (α, θ)-pair is then backsubstituted into (5) and (6), and the camera center and orientation with respect to the world reference frame are finally given as

$$C = P_1 + N^T \cdot C^\eta \tag{12}$$

and

$$R = N^T \cdot Q^T \cdot T. \tag{13}$$

Note that a proper implementation of the algorithm excludes the use of any computationally expensive trigonometric functions. Using the restricted domains of parameters α and θ, all appearing trigonometric forms of the parameters can be directly derived from $\cot\alpha$ and $\cos\theta$ using simple trigonometric relationships. Furthermore, during the tests we observed that, due to noise, we sometimes get complex solutions with small imaginary parts instead of real ones. In this case, it is better to retain the real part of these solutions instead of ignoring them completely.

The full procedure may be summarized as follows:

- compute the transformation matrix T and the feature vector $\vec{f}_3^T$ using (1)
- compute the transformation matrix N and the world point P_3^η using (2)
- extract p_1 and p_2 from P_3^η
- compute d_{12} and b using (3)
- compute ϕ_1 and ϕ_2 using (8)
- compute the factors a_4, a_3, a_2, a_1, and a_0 of polynomial (11)
- find the real roots of the polynomial (values for $\cos\theta$)
- for each solution, find the values for $\cot\alpha$ using (9)
- compute all necessary trigonometric forms of α and θ using trigonometric relationships and the restricted parameter domains
- for each solution, compute C^η and Q using (5) and (6), respectively
- for each solution, compute the absolute camera center C and orientation R using (12) and (13), respectively
- backproject a fourth point for disambiguation

Please note that the final version of the Matlab- and C++-implementations used during the experiments can be downloaded at

- *http://www.laurentkneip.de*

3. Results

The algorithm presented in Section 2 has been thoroughly tested by means of synthetic data, and compared to Gao's [12] solution to the P3P-problem. The code for the comparison algorithm is available online. In order to have a fair comparison, Gao's solution for finding the three distances between the camera center C and the world points P_i has been extended by Arun's method [3] to find the aligning transformation between the two point sets. This is needed in order to derive the absolute position and orientation of the camera frame from the relative position of the three points, and thus obtain comparable entities. Gao's method, obviously, also returns up to four possible solutions. For both algorithms, the disambiguation of the four possible solutions has been done using the same fourth point, and exactly the same method.

The synthetic data consists of 1'000 3D points that are uniformly distributed in a volume of 4×4×4, centered around the origin of the world frame. The position of the camera is fixed at $C = \begin{pmatrix} 0 & 0 & 6 \end{pmatrix}^T$, and the orientation is kept at $R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$, thus perfectly downlooking.

For each experimental run, synthetic 2D-3D correspondences are created by randomly selecting three points from the entire point set, and projecting them into image space using a virtual calibrated camera with resolution 640×480, principal point $(u_c, v_c) = (320, 240)$, and effective focal lengths $f_u = f_v = 800$. Depending on the experiment, a different level of white Gaussian noise ranging from 0 to 5 pixels is then added to the 2D coordinates before finally reprojecting the features on the unit sphere.

3.1. Numerical stability

To analyse the numerical stability of the proposed algorithm, we perform 50'000 runs without any noise added to the 2D coordinates. The results are shown in Fig. 5. It can be observed that the distribution of the numerically-caused imprecisions provide a significantly higher concentration around the smallest errors for our new solution in comparison to Gao's standard solution. Note that the range of the y-scale had to be cropped for a proper visualization. The actual peak value for our new solution lies around 8'000. The range of the x-scale is also cropped, meaning that Gao's solution is quite uniform and the number of votes decreases only slowly toward higher error values.

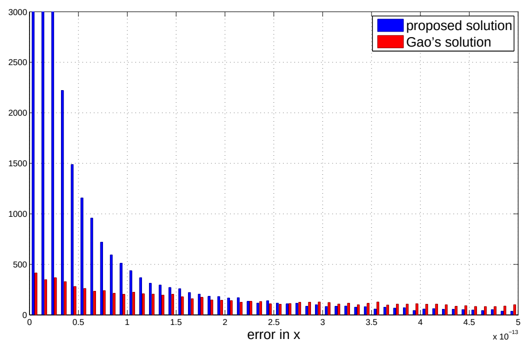

Figure 5. Numerical stability of the proposed algorithm and Gao's solution.

3.2. Computational complexity

This experiment targets the evaluation of the computational cost of both approaches. In order to have a fair comparison, both algorithms have been implemented in C++. The experiment consists of 1'000 runs, where one run comprises 1'000 evaluations of the same set of 3D world points. Measuring the time of 1'000 executions instead of only one allows us to obtain a relatively precise value for the execution time not suffering from any quantization errors, or errors caused by any other computational overhead. The results are shown in Fig. 6. The difference in the execution time is significant, which is mainly due to the fact that the proposed solution directly computes the absolute position and orientation instead of having to rely on a least squares solution for the point set alignment in a postprocessing step. The proposed solution is 15 times faster than Gao's solution, and takes only 2 microseconds to be executed on a 2.8GHz Dual Core laptop.

3.3. Noise sensitivity

This last experiment evaluates the effects of noise on the accuracy and precision of the computation, that is, mean and standard deviation of the absolute value of the translational and rotational errors. 1'000 runs have been performed for different Gaussian noise levels ranging from 0 to 5 pixels, and the results are shown in Fig. 7. This figure shows that mean and standard deviation for both methods behave in a very similar way, and increase almost linearly with the noise level. One exception is given with the translational error of Gao's method, where several peaks appear in the mean and standard deviation for increased noise levels. This behavior is caused by single outlier results with very high error. The fact that our algorithm does not show these peaks shows that it is more stable in degenerate configurations than Gao's standard solution.

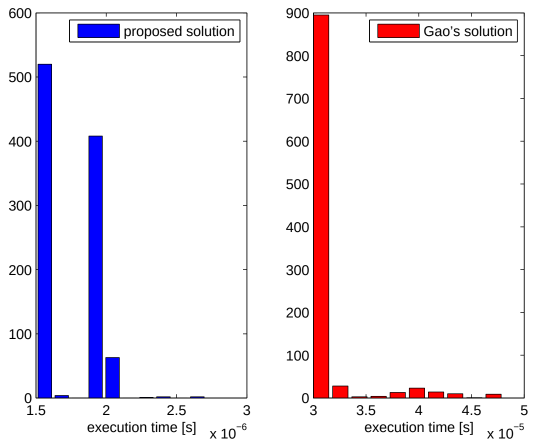

Figure 6. Average execution time of the proposed algorithm and Gao's standard solution.

4. Conclusion

In this paper, we proposed a new method for solving the P3P-problem as a direct computation of the absolute camera position and orientation, which is novel in the domain. The derivations are easy to understand, and the final algorithm is more lightweight than existing P3P-solutions since it does not depend on any postprocessing step for the alignment of two point sets. It leads to a comparable accuracy and precision at a substantially lower computational cost. Furthermore, the algorithm has improved numerical stability, and is affected less by degenerate configurations of the selected world points. It represents a very compact algorithm, particularly suitable for any RANSAC-outlier-rejection step, which is always recommended before applying PnP or non-linear optimization of the final solution.

References

[1] M. 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. 1

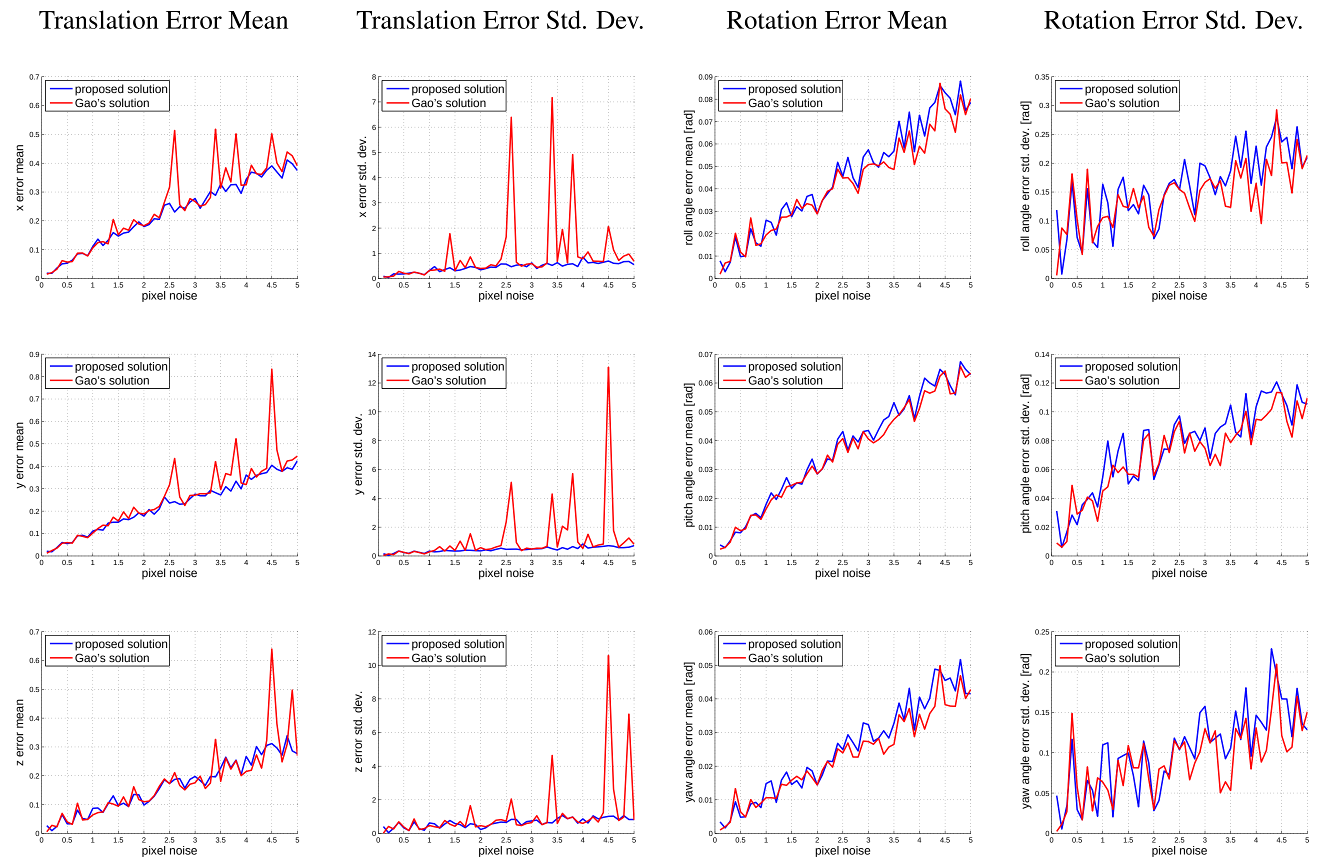

Figure 7. Mean error and standard deviation of position and orientation of the camera for the proposed solution and Gao's standard solution. Results are obtained over 1000 runs and in function of the pixel noise.

[2] A. Ansar and K. Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003. 2

[3] K. Arun, T. Huang, and S. Blostein. Least-squares fitting of two 3-d points sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987. 2, 5

[4] D. De Menthon and L. Davis. Exact and approximate solutions of the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(11):1100–1105, 1992. 2

[5] D. De Menthon and L. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141, 1995. 2

[6] M. Dhome, M. Richetin, and J. Lapreste. Determination of the attitude of 3D objects from a single perspective view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, 1989. 2

[7] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, Cambridge, MA, USA, 1993. 1

[8] S. Finsterwalder and W. Scheufele. *Das Rückwärtseinschneiden im Raum*. Verlag Herbert Wichmann, Berlin, Germany, 1937. 1

[9] P. Fiore. Efficient linear solution of exterior orientation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):140–148, 2001. 2

[10] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 1, 2

[11] W. Gander. Least squares fit of point clouds. *Solving Problems in Scientific Computing using MAPLE and MATLAB*, pages 339–349, 2004. 2

[12] X. Gao, X. Hou, J. Tang, and H. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003. 2, 5

[13] E. Grafarend, P. Lohse, and B. Schaffrin. Dreidimensionaler Rückwärtsschnitt, Teil 1: Die projektiven Gleichungen. *Zeitschrift für Vermessungswesen, Geodätisches Institut, Universität Stuttgart*, pages 1–37, 1989. 1

[14] J. A. Grunert. Das pothenotische Problem in erweiterter Gestalt nebst über seine Anwendungen in Geodäsie. In *Grunerts Archiv für Mathematik und Physik*, 1841. 1

[15] R. Haralick, C. Lee, K. Ottenberg, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. In *IEEE International Conference on Computer Vision and Pattern Recognition*, Maui, USA, 1991. 1, 2

[16] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, second edition, 2004. 1

[17] R. Horaud, B. Conio, and O. Leboulleux. An analytic solution for the perspective 4-point problem. *Computer Vision, Graphics, and Image Processing*, 47:32–44, 1989. 1, 2

[18] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4:629–642, 1987. 2

[19] B. Horn, H. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America*, 5(7):1127–1135, 1988. 2

[20] Y. Hung, P. Yeh, and D. Harwood. Passive ranging to known planar point sets. In *IEEE International Conference on Robotics and Automation*, San Francisco, CA, USA, 1986. 1

[21] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, 2006. 2

[22] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate O(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):578–589, 2009. 2

[23] S. Linnainmaa, D. Harwood, and L. Davis. Pose estimation of a three-dimensional object using triangle pairs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):634–647, 1988. 1

[24] C. McGlove, E. Mikhail, and J. Bethel. *Manual of photogrametry*. American society for photogrammetry and remote sensing, Bethesda, Maryland, USA, fifth edition, 2004. 1

[25] E. Merritt. Explicit three-point resection in space. *Photogrammetric Engineering*, 15(4):649–655, 1949. 1

[26] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1), 2006. inaugural issue. 2

[27] D. Nistér and H. Stewenius. A minimal solution to the generalized 3-point pose problem. *Journal of Mathematical Imaging and Vision*, 27(1):67–79, 2006. 2

[28] L. Quan and Z. Lan. Linear n-point camera pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):774–780, 1999. 1, 2

[29] I. Skrypnyk and D. Lowe. Scene modelling, recognition and tracking with invariant image features. In *International symposium on mixed and augmented reality*, Washington, DC, USA, 2004. 2

[30] C. Su, Y. Xu, H. Li, and S. Liu. Application of Wu's method in computer animation. In *Fifth International Conference on CAD/CG*, Shenzhen, China, 1997. 1

[31] I. Sutherland. Sketchpad: A man machine graphical communications system, 1963. Technical Report 296, MIT Lincoln Laboratories. 1

[32] B. Triggs. Camera pose and calibration from 4 or 5 known 3D points. In *IEEE International Conference on Computer Vision*, Kerkyra, Greece, 1999. 2

[33] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. 2

[34] W. Wolfe, D. Mathis, C. Weber, and M. Magee. The perspective view of three points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):66–73, 1991. 1

[35] J. Yuan. A general photogrammetric method for determining object position and orientation. *IEEE Transactions on Robotics and Automation*, 5(2):129–142, 1989. 1