— Supplementary Material —
P1AC: Revisiting Absolute Pose From a Single Affine Correspondence

Jonathan Ventura[1] Zuzana Kukelova[2] Torsten Sattler[3] Dániel Baráth[4]
[1] Department of Computer Science & Software Engineering, Cal Poly
[2] Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague
[3] Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague
[4] Department of Computer Science, ETH Zürich

In this supplementary material, we present additional experiments to support the claims in the main paper, explain and analyze alternative solver formulations, and present more detailed explanations of some aspects of the main paper. Section 1 presents extra experimental results on noise sensitivity. Section 2 evaluates an alternative application of the IPPE method. Section 3 analyzes degenerate cases for the P1AC solver. Section 4 introduces a fast P1AC solution using an approximate rotation representation which is only valid for small rotations. Section 5 details an alternate P1AC solution using the 3×3 rotation matrix representation.

1. Noise sensitivity tests

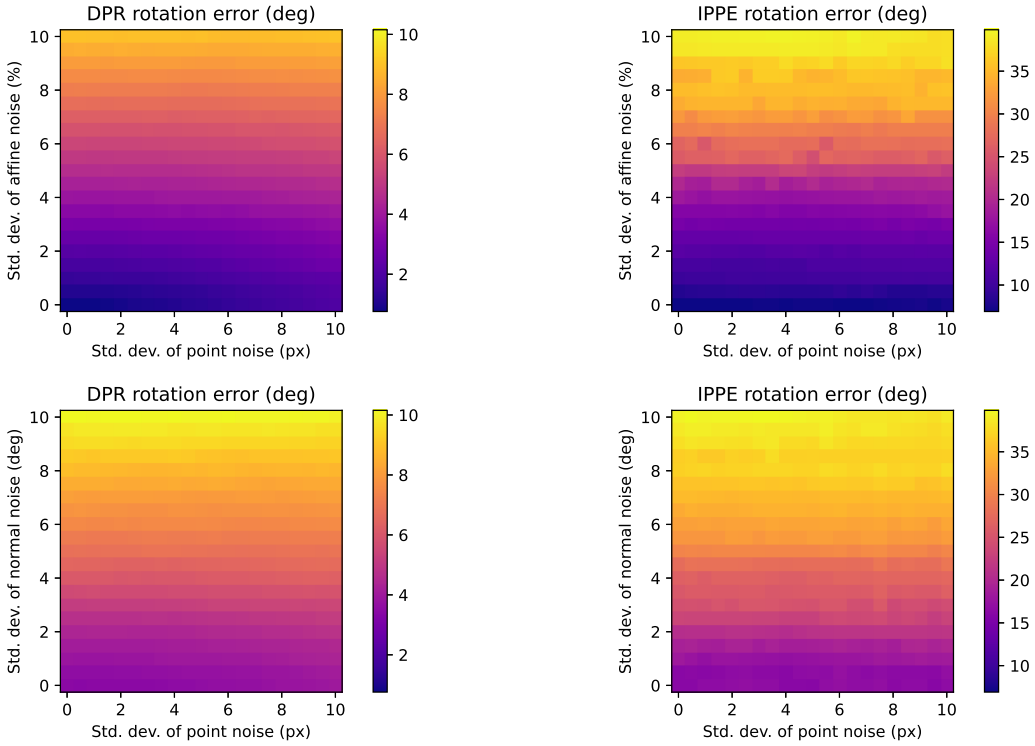

Figure 1: Analysis of error from DPR and IPPE methods with respect to various types and levels of noise.

Figure 1 plots the rotation error versus noise for IPPE and DPR. The results for DPR are similar to P1AC, whereas IPPE's

Dataset	Recall (0.1m/1°) ↑ P1AC	IPPE (4PC)	Rec. (0.2m/1°) ↑ P1AC	IPPE (4PC)
Cambridge Landmarks	**65.08**	46.51	**80.65**	58.92

Table 1: Comparison of P1AC and IPPE (4PC) results on **Cambridge Landmarks**.

Dataset	Recall (0.25m/2°) ↑ P1AC	IPPE (4PC)	Recall (0.5m/5°) ↑ P1AC	IPPE (4PC)	Recall (5m/10°) ↑ P1AC	IPPE (4PC)
Aachen Day	**62.0**	60.1	**84.6**	81.3	**95.9**	92.6
Aachen Night	**51.3**	38.2	**66.0**	49.2	**82.2**	60.2

Table 2: Comparison of P1AC and IPPE (4PC) results on **Aachen Day-Night**.

error is far higher (note the color bar range).

2. IPPE with four point correspondences

We conducted extra tests of the IPPE solver on real data using four PCs (rather than a single AC with added virtual correspondences) within the GC-RANSAC framework, in a manner consistent with our other experiments. Given the requirement of IPPE for coplanar points, we implemented the NAPSAC sampler [5] within GC-RANSAC to select proximate points in the samples. As shown in Tables 1 and 2, the proposed P1AC substantially outperforms IPPE (4PC) on all tested datasets.

3. Degeneracies

As mentioned in Section 4.1 of the main paper, we investigated several specific problem configurations to search for situations where our P1AC solver might produce inaccurate results.

3.1. Weak perspective

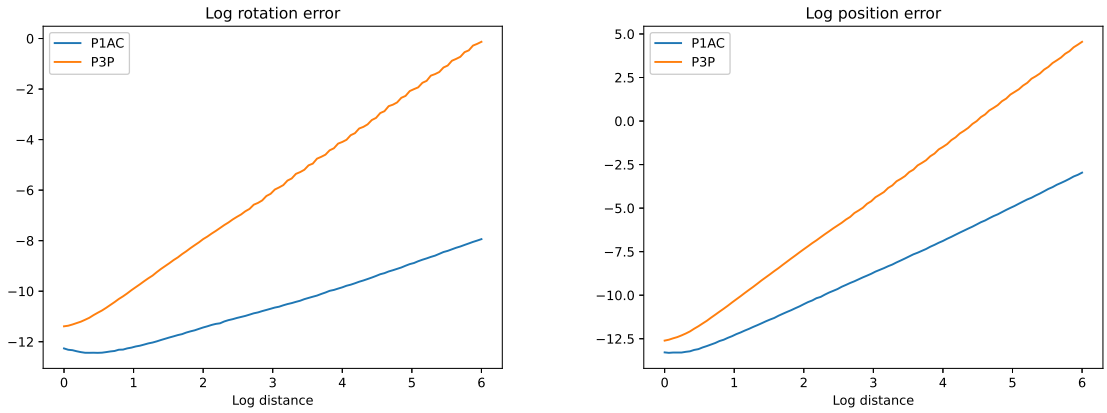

Figure 2: Plot of rotation and position error of P1AC and P3P over increasing distance of the query camera from the point. The error of both solvers increases with distance, although the error of P3P increases at a faster rate than P1AC.

We investigated the change in performance as the configuration approaches weak perspective, where the differences in depths in the object / scene are much smaller than the distance to the object / scene. To test this scenario, we generated random problems in the following manner. We select a random rotation for the object plane and a center point for the plane selected from a 3D normal distribution. Two co-planar points are placed at a distance of 0.1 from the center point along the principal axes of the plane. A random reference camera is selected with unit distance from the origin, and a query camera

at the chosen distance from the origin. Both the reference and query camera look at a random target point sampled from $[-0.5\ 0.5]^3$. No noise was added to the observations in this experiment.

As the distance of the query camera increases, the projected size of the planar object decreases, and the configuration approaches the weak perspective case. Figure 2 plots the rotation and position error of P1AC and P3P as the distance of the query camera increases. The error of both P1AC and P3P increases with query camera distance, with the error of P3P increasing more quickly than P1AC.

3.2. Near-identity rotation

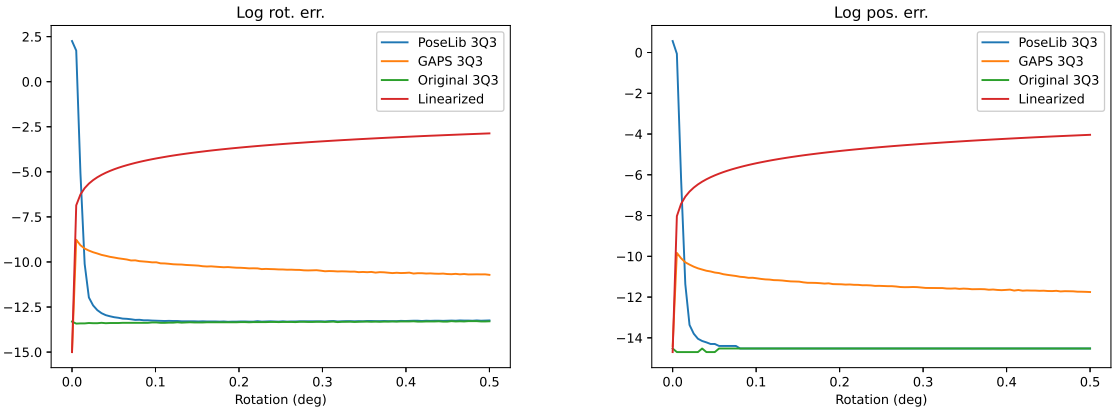

Figure 3: Plot of rotation and position error of various solvers when the ground truth rotation solution has small magnitude ($\leq 0.5°$). The PoseLib 3Q3 implementation is numerically unstable for very small rotations ($< 0.05°$).

When the rotation is equal to or close to identity, we found that the performance of our P1AC solver depends on the implementation of 3Q3 solver used. When using the 3Q3 implementation from PoseLib [2], the solver often produces zero solutions or inaccurate solutions. This is likely because the last column of the coefficient matrix, after Gaussian elimination, contains extremely small numbers. When using the original 3Q3 implementation provided by the authors [1], the accuracy remains stable.

In the specific case where the rotation is identity and the translation is zero, both the PoseLib 3Q3 implementation and the original 3Q3 implementation often fail to produce a solution. However, we found that a 3Q3 solution produced by the GAPS automatic generator [4] avoids all instability with near-identity rotations, even in the case of zero translation. Using the 3Q3 solver produced by GAPS, the P1AC solver has an average timing of 15 μs.

We see two possible solutions to the issue encountered with the PoseLib 3Q3 and near-identity rotations, if the P1AC solver is to be used in a scenario where the rotation is expected to be close to identity. One option is to use the GAPS 3Q3 solver, at the cost of a slightly slower minimal solver. A faster option is to append the solution from a linearized rotation solver (described below) to the list of PoseLib 3Q3 solutions.

Figure 3 plots the average rotation error for the various solvers over a range of rotation magnitude settings. For each rotation magnitude setting we generated 10,000 random problems.

4. Linearized rotation solution

When the rotation is close to identity, we can use a small-angle approximation to arrive at a linear solution to the P1AC problem. Let $\mathbf{r}$ be the $SO(3)$ representation of the rotation R. We linearize R using the first order Taylor expansion $\mathsf{R} \approx \mathsf{I}_{3\times3} + [\mathbf{r}]_\times$. Now the P1AC equations become six linear equation in six unknowns $\mathbf{r}, \mathbf{t}$ and are easily solved.

The solver is extremely fast, with an average timing of 0.366 μs.

5. Rotation matrix solution

The P1AC formulation presented in the main paper is based on the Cayley parameterization of the rotation matrix. As mentioned in the main paper, this parameterization introduces a degeneracy for rotations of exactly 180°. Even though this

degeneracy is not a problem in practice, we will present here a formulation of the P1AC problem and a solution that does not suffer from this degeneracy.

If we parameterize the rotation directly as a 3×3 matrix, then we have 12 parameters in $\mathsf{P} = [\ \mathsf{R} \mid \mathbf{t}\ ]$. We can write the six P1AC equations as a matrix-vector multiply:

$$\mathsf{M}\bar{\mathsf{P}} = \mathbf{0}. \tag{1}$$

where $\bar{\mathsf{P}}$) denotes the matrix P rearranged into a vector.

Let B be the 12×6 nullspace of M. We compute the nullspace of M using singular value decomposition (SVD). Any solution, up to scale, for $\bar{\mathsf{P}}$ has the form

$$\bar{\mathsf{P}} = \mathsf{B}\mathbf{b}, \tag{2}$$

where $\mathbf{b}$ is a vector of six coefficients for the basis vectors in B. Our goal is to find solutions for $\mathbf{b}$ that make the rotation matrix orthogonal.

We follow the solution procedure described by Ventura et al. [6]. Assuming $b_6 \neq 0$, we remove one parameter and simplify the solution by fixing $b_6 = 1$. Let $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ be the rows of R and $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$ be the columns. The following constraints ensure that R is orthogonal, up to scale:

$$\begin{aligned}
||\mathbf{r}_1||^2 - ||\mathbf{r}_2||^2 = 0, \quad ||\mathbf{r}_1||^2 - ||\mathbf{r}_3||^2 = 0, \\
||\mathbf{c}_1||^2 - ||\mathbf{c}_2||^2 = 0, \quad ||\mathbf{c}_1||^2 - ||\mathbf{c}_3||^2 = 0, \\
\mathbf{r}_1 \cdot \mathbf{r}_2 = 0, \quad \mathbf{r}_1 \cdot \mathbf{r}_3 = 0, \quad \mathbf{r}_2 \cdot \mathbf{r}_3 = 0, \\
\mathbf{c}_1 \cdot \mathbf{c}_2 = 0, \quad \mathbf{c}_1 \cdot \mathbf{c}_3 = 0, \quad \mathbf{c}_2 \cdot \mathbf{c}_3 = 0.
\end{aligned} \tag{3}$$

Plugging in equation 2 to these constraints results in a system of ten quadratic equations in twenty-one monomials with variables $b_1, \ldots, b_5$. After extracting the roots of this system of equations, for each solution we divide $\tilde{\mathbf{p}}$ by $||\mathbf{c_1}||$ and negate $\tilde{\mathbf{p}}$ if necessary to ensure that $\det(\mathsf{R}) = 1$.

This system has eight solutions and can be solved using the action matrix method and an automatic solver generator [3]. The resulting solver involves elimination of a 47×55 template matrix and eigendecomposition of an 8×8 matrix.

The solver is much slower than the 3Q3 solver, having an average timing of 27 μs. It does not exhibit any instability with near-identity rotations.

In contrast to [6], we discovered that the system of equations 3 can be further simplified by eliminating one unknown, e.g., b_1. This can be done by rewriting the ten equations (3) in a matrix form $\mathsf{C}\mathbf{v} = \mathbf{0}$, where C is a 10×21 coefficient matrix, and $\mathbf{v}$ is a vector of 21 monomials ordered using the lexicographic ordering. After eliminating the matrix C, six monomials containing b_1 can be expressed as quadratic polynomials in $b_2, \ldots, b_5$. In this way, b_1 can be eliminated from the original equations. Moreover, new equations that express relationships between different monomials can be added to the original equations, e.g. if $b_1 = p_1(b_2, \ldots, b_5)$ and $b_1 b_2 = p_2(b_2, \ldots, b_5)$, where p_1 and p_2 are polynomials in $b_2, \ldots, b_5$, extracted from the eliminated matrix C, then a new equation that can be added to the original equations has the form $p_1 b_2 = p_2$. In this way, a new system of polynomials equations in four unknowns can be generated. This system can be solved using the automatic generator [3] and results in a solver that performs elimination of a 29×37 template matrix and eigendecomposition of an 8×8 matrix. Although this solution path is faster than the 47×55 path, we found that it is less numerically stable.

References

[1] Zuzana Kukelova, Jan Heller, and Andrew Fitzgibbon. Efficient intersection of three quadrics and applications in computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1799–1808, 2016. 3

[2] Viktor Larsson. PoseLib - Minimal Solvers for Camera Pose Estimation. https://github.com/vlarsson/PoseLib, 2020. 3

[3] Viktor Larsson, Kalle Astrom, and Magnus Oskarsson. Efficient solvers for minimal problems by syzygy-based reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 820–829, 2017. 4

[4] Bo Li and Viktor Larsson. GAPS: Generator for automatic polynomial solvers. *arXiv preprint arXiv:2004.11765*, 2020. 3

[5] Philip Hilaire Torr, Slawomir J Nasuto, and John Mark Bishop. NAPSAC: High noise, high dimensional robust estimation-it's in the bag. In *British Machine Vision Conference (BMVC)*, volume 2, page 3, 2002. 2

[6] Jonathan Ventura, Clemens Arth, Gerhard Reitmayr, and Dieter Schmalstieg. A minimal solution to the generalized pose-and-scale problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 422–429, 2014. 4