Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Canonical P1AC: A Direct Solver for P1P with Affine Correspondences or Field Gradients

Fabrice Mayran de Chamisso

fabrice.mayran-de-chamisso@cea.fr

Université Paris-Saclay, CEA, List

Research Article

Keywords: P1P, P1AC, Perspective-n-Point, Pose estimation, Minimal solvers, Affine correspondences

Posted Date: September 24th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10601162/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Canonical P1AC: A Direct Solver for P1P with Affine Correspondences or Field Gradients

Fabrice Mayran de Chamisso[1*]

[1*]Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France.

Corresponding author(s). E-mail(s): fabrice.mayran-de-chamisso@cea.fr;

Abstract

Solving a PnP problem from a single affine correspondence, or P1AC, has recently gained traction. The best current approach uses an expensive 3Q3 solver. In this paper, we propose a new minimal solver whose core is a single quadratic equation. We decompose the P1AC problem into a canonicalization step followed by a simple quadratic solver. Our method uses the fact that a single affine correspondence is equivalent to two gradient correspondences, forming a sort of infinitesimal P3P. This observation opens up many potential uses with computer vision approaches producing dense descriptor fields over images that are isometry-invariant like DenseObjectNets, SurfEmb or DINO. We provide an in-depth analysis of all degenerate cases where the solver produces less than its nominal two solutions with positive depth and discuss failure cases with an infinite number of solutions. We also analyze the impact of various noise sources on the solution quality.

Keywords: P1P, P1AC, Perspective-n-Point, Pose estimation, Minimal solvers, Affine correspondences

1 Introduction

2D–3D registration or image-based localization consists of finding the pose or *rigid transformation* in SE(3) of an object or camera in a given 2D image. A rigid transformation has six degrees of freedom (DOF), and as such requires at least six equations derived from observations.

A large variety of approaches have been developed over the years to perform 2D–3D registration. Among these, many are based on matching features between a 2D image and a 3D reference and then using an algorithm to recover the transformation from 2D–3D correspondences. Used features are generally points (the PnP problem) but may include lines (PnL), oriented contours or covariance ellipses [1]. The correspondences themselves can be equipped with a covariance matrix, as explored in [2], leading to the P1AC problem, standing for P1P with Affine Correspondences. A single affine correspondence consists of a 3D-to-2D point correspondence (2 DOF) and a 2×2 matrix transformation (4 DOF), which in theory means a single affine correspondence is sufficient to determine all degrees of freedom of the rigid transformation. In practice, as known from the P3P literature, some configurations are indistinguishable in a camera view, so that each non-degenerate affine correspondence leads to at least four solutions, which our derivation will make apparent.

Most existing PnP approaches are designed to process grayscale or RGB images. Approaches like DenseObjectNets [3], SurfEmb [4] and DINO(v2) [5, 6] produce smooth multidimensional descriptor fields even on non-textured areas while blurring the objects' contours and salient features (see

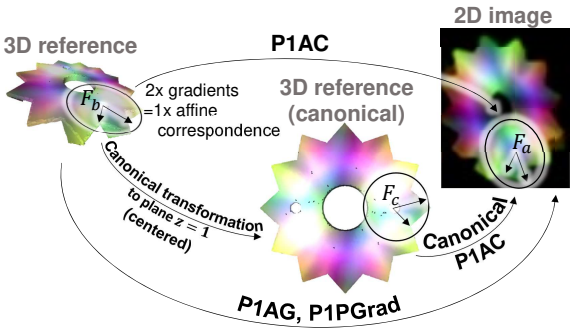


Fig. 1: The P1AC problem is simplified by using an intermediate *canonical* reference frame F_c centered around the interest point, with the local plane aligned to $z = 1$. The simplified canonical P1AC problem has decoupled rotation and translation, allowing for easier solving. Our P1AG and P1PGrad solvers are wrappers around the canonical P1AC solver adapted for one affine correspondence or two gradients.

Figure 2(a)). This hints towards dense gradient-based approaches as opposed to sparse descriptors computed on salient features such as SIFT [7]. Typical registration pipelines such as [4] do not model or exploit the local deformation of the dense descriptor field, notably including field gradients.

One of the key differentiating factors of P1AC compared to P3P is its local nature: by definition of sampling three correspondences, P3P approaches are non-local, meaning they will fail for instance on deformable or articulated objects. In contrast, P1AC only uses information around a single point, and as such will correctly align the individual parts of an articulated assembly, provided of course that affine correspondences are correct.

This paper proposes a new P1AC, which is a drop-in replacement for the solver exposed in [2]. In essence, it produces the same result, but faster, with fewer edge cases, more numerical accuracy, and known degenerate cases. Our contributions are:

1. A **canonical reference frame transformation** that decouples rotation and translation in the P1AC problem, reducing it to a single quadratic equation (Section 3).
2. A **direct quadratic solver** for the canonical P1AC problem, yielding up to two solutions with positive depth. The solver is at least 10× faster and orders of magnitude more numerically stable than the 3Q3 solver of [2] (Section 3, Table 1).
3. A **complete degeneracy analysis** characterizing all configurations where the solver does not produce its nominal amount of solutions, including the single-solution case and the infinite-solution case (colinear gradients, non-invertible affine matrix) (Section 3).
4. A **gradient-based formulation** (P1PGrad) that establishes the equivalence between one affine correspondence and two gradient correspondences, enabling direct use of dense descriptor fields without explicit affine matrix estimation (Section 3).
5. A **sensitivity analysis** quantifying the impact of plane slope, descriptor noise, correspondence accuracy, number of fields, and planarity deviation on the solver's output (Section 4.2).

Computing a spatial gradient on an image gives a 2D gradient vector for each channel. One point-to-point correspondence from a 2D image to a 3D reference gives two equations, so one point correspondence with two 2D gradients leads to six equations. This is exactly enough to fully constrain the six degrees of freedom of an SE(3) transformation, provided that all six equations are independent and do not generate multiple indistinguishable solutions. This problem, which we further refer to as P1PGrad, can be viewed as an infinitesimal P3P. We will briefly discuss how this problem is strictly equivalent to P1AC [2].

The rest of this paper is organized as follows: after a brief review of existing 2D–3D registration approaches, we introduce the canonical P1AC frame. By expressing the problem in this reference frame, the translation, rotation and scale become decoupled enough to use a simple solver, which is the core discovery of this paper. We call the reduced problem *canonical P1AC* and provide a quadratic direct solver, which is our main contribution. We provide two additional solvers for the whole problem: P1AG, a drop-in replacement for the P1AC solver of [2] for processing affine correspondences, and P1PGrad for processing gradients (Figure 1). Both are lightweight wrappers around the canonical P1AC solver and provide

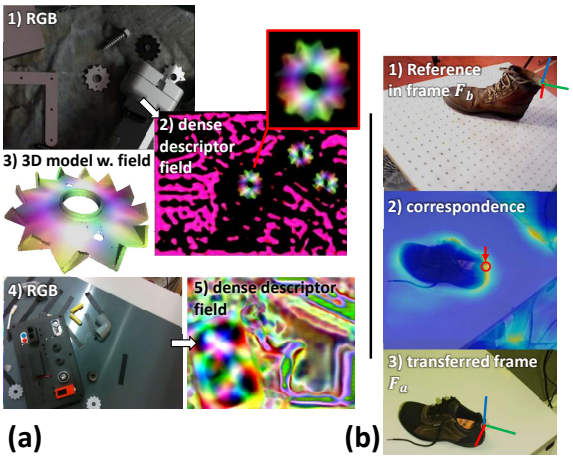


Fig. 2: Exemple use cases. (a) A neural network such as DenseObjectNets [3], SurfEmb [4] or DINO(v2) [5, 6] is trained to produce descriptors that are invariant under affine transformations of the camera or the object. Here, we project n-dimensional (typically, less than 10 dimensions for [4] and more than 128 dimensions for [5, 6]) descriptors into RGB space for visualization. From image a.1), descriptors a.2) are generated. The descriptors are transferred to the 3D model a.3) using either an MLP [4] or a collection of descriptor calculations on renders of the model. a.5) is another descriptor field generated from a.4). Descriptors a.5) do not present sharp features typically observed on RGB images but have distinctive values on the entirety of the objects including textureless areas. (b) Such descriptors allow transferring a point from a reference RGB-D image of an object b.1) to the point with the highest descriptor matching score in image b.3), where b.2) is the heatmap of matching scores for the chosen point in b.1). Being able to also transfer a local reference frame F_b from b.1) to b.3) allows robot grasping. It cannot be done with a P3P and a global reference frame if the objects in b.1) and b.3) have different geometries within the same class.

up to two solutions with positive depth and two with negative depth. We show how to robustify P1PGrad by performing channel-wise RANSAC. We then test P1AG and P1PGrad on synthetic data, with special focus on the quality of gradients/affine matrices and its impact on the final result.

2 Related Work

Minimal solvers in geometric vision. A minimal solver computes a geometric estimate from the smallest possible number of observations. The P3P problem — recovering camera pose from three 2D–3D point correspondences — is the canonical example. Many approaches have been developed to solve P3P using iterative or direct solvers; Ding et al. [8] propose solving it as the intersection of two conics, achieving best-in-class mean and max error. We take this approach as baseline both for its low running time and accuracy.

When more than three correspondences are available, two strategies are typically used: solving using multiple or all correspondences, as in EPnP [9] or UPnP [10], or using multiple P3P solutions in a RANSAC scheme. Using all correspondences lacks robustness to erroneous correspondences, whereas a RANSAC scheme scales as $1/p^3$ with p the probability of picking a non-outlier point. Furthermore, techniques based on multiple correspondences tend to fail for symmetrical objects and scenes with redundant features, as each correspondence found is attached to a random symmetry. This is further amplified when multiple instances of the object are present, as displayed on Figure 3. Finally, approaches using multiple correspondences prefer distant points in order to get precise transformations in the presence of pixel-level correspondence noise. Consequently, sampling correspondences in a small neighborhood is generally not an option, and these approaches may fail for deformable and articulated objects and for objects for which segmentation data is inaccurate or nonexistent.

Affine correspondences and differential methods. A solution to the outlier and symmetry problem is to only consider local information in the vicinity of a single correspondence. The classical take on affine transformations is to try and create descriptors invariant under affine transformations [11]. On the contrary, Köser et al. [1, 12] and more recently [13] leverage affine local descriptions forming *affine correspondences*, that is, correspondences between points equipped with an affine description of the local surface warping. This is motivated by the fact that planar surfaces in images transform by affine transformations when viewed from different viewpoints [11].

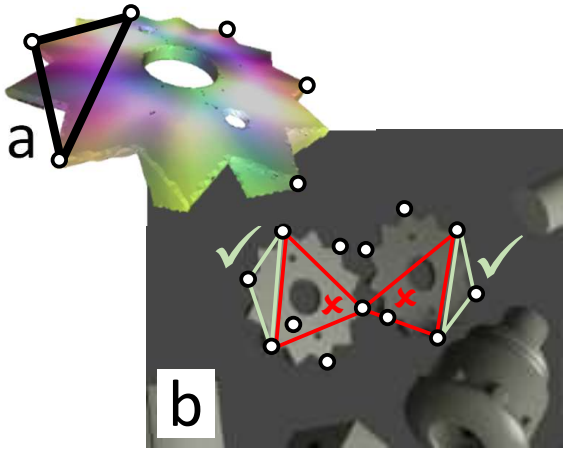


Fig. 3: With symmetrical objects and multiple object instances, each correspondence is attached to one possible object pose. With two objects, each having a perfect 180° rotation invariance (image b), there are only $1/4 \times 1/4 = 1/16$ chances for three random valid correspondences to form a consistent triplet in the 3D reference (image a) allowing P3P to return the correct pose. With a sixfold symmetry and ten objects, which is not uncommon in bin picking applications, the chances of sampling a consistent triplet of correspondences are $(1/60)^2 = 2.8 \cdot 10^{-4}$. This does not even take into account erroneous matches and XY coordinate noise when matching. Anything below $\sim 1 \cdot 10^{-3}$ renders RANSAC ineffective in practice due to time constraints.

The work of Köser et al. [1] is particularly relevant: it shows that a pose can be recovered from a single affine correspondence, framing the problem as a *differential spatial resection* — the limiting case of P3P as the three points converge to a single point. Eichhardt and Baráth [13] extend this to relative pose from a single affine correspondence combined with deep-learned depth. These works establish the theoretical foundations upon which P1AC [2] and our solver build.

P1AC and the 3Q3 solver. P1AC [2] formalizes the problem of computing a pose from a single correspondence and a 2×2 affine matrix. It uses the two equations describing the point correspondence and the four equations describing the affine transformation. The solver in [2] formulates the problem as a system of three quadratic equations in three unknowns (a 3Q3 system), solved using the resultant method. This requires computing the roots of a degree-4 polynomial and performing several matrix inversions, leading to higher computational cost and numerical sensitivity compared to a direct quadratic solver. The pose can then be refined using non-linear optimization, scored using RANSAC, or grouped using 6D pose-space clustering and averaging as in [14]. Compared to [2], our P1AG solver is much simpler and more computationally efficient, although it gives by construction the same result up to numerical accuracy in non-pathological cases. One key difference between our P1AG solver and P1AC is the usage of an intermediary *canonical* reference frame, which is the equivalent for gradients of the technique used by [15] for points in a generalized P3P context. The canonical frame allows factorizing the quartic equation into two quadratic equations.

From affine correspondences to gradients. P1AG internally decomposes a single affine correspondence into two vector correspondences which are homogeneous to 2D gradients. We use the equivalence between one affine correspondence and two gradients expressed below to create our alternate solver P1PGrad, which directly solves from two gradients without ever expressing the affine correspondence. The connection between affine correspondences and gradients is rooted in differential geometry: the local affine warp of an image patch is precisely the Jacobian of the projection mapping, and the gradient of a scalar field transforms covariantly under this mapping. This means that two independent gradient correspondences at a single point carry the same geometric information as the affine matrix, while being easily interpretable first-order quantities for which simple computation methods are known. This is a major advantage compared to deep-learning computations used in [2] to obtain the affine matrix.

First-order vs. second-order local information. The goal of P1PGrad (our one correspondence, two gradients formulation) is to use the maximum amount of information available at order zero (pixel color, correspondences) and

order one (first spatial derivative, aka. the gradient) in a multichannel image. In comparison, affine-transformation-based approaches like [11] and [2] use second-order information on each single channel. Second-order methods only require one image channel compared to the two channels in first-order methods, with the trade-off that reliable second-order information is noisier and harder to obtain, so that neural networks are typically used to obtain affine descriptions [16]. To our knowledge, no such networks are available for robust processing of descriptor fields such as that of Figures 2 and 3. The second-order robustness issue is well known for 3D reference frames like SHOT [17] or BOARD [18] on 3D data, especially when compared to the robustness of first-order features such as surface normals. Also, affine-transformation-based approaches make an underlying assumption that surfaces are locally planar [2]. When reconstructing buildings from multiple views, many surfaces have close to zero curvature, validating the assumption. However, when performing object registration such as on the BOP challenge [19], many objects only have nonzero curvature surfaces. P1PGrad recovers affine transformations from very small areas, typically a disc of radius less than five pixels, so that large-scale planarity is not as critical. We study how deviations from planarity impact P1PGrad in the experiments section of this paper.

3 Mathematical formulation

Notations: We use italic capital letters for matrices and bold for vectors. If $\mathbf{u} = (u_1, u_2)$, we define $\hat{\mathbf{u}} = (u_1, u_2, 0)$ and $\tilde{\mathbf{u}} = (u_1, u_2, 1)$. We use ∇ to denote spatial gradients in either 3D or 2D. $a:b$ denotes the integer range from a to b, inclusive. So, $1:3$ is $1, 2, 3$. We index matrices as $M_{\text{row,col}}$ starting at index 1. The notation $\|\mathbf{v}\|^2$ denotes the squared norm $\mathbf{v}\cdot\mathbf{v}$. We represent rotations as 3x3 matrices.

3.1 Gradient definition, properties and computation

Definition: The spatial gradient $\nabla f_{\mathbf{X}}$ of a scalar field $f(\mathbf{X})$ defined over $\mathbb{R}^n$ is defined by $df_{\mathbf{X}}(\mathbf{v}) = \nabla_{\mathbf{X}} f(\mathbf{X}) \cdot \mathbf{v}$ where $df_{\mathbf{X}}$ is the total differential of f at point $\mathbf{X} = (x, y, z)$. In 3D, $\nabla_{\mathbf{X}} f(\mathbf{X}) = \left(\frac{\partial f}{\partial x}(\mathbf{X}), \frac{\partial f}{\partial y}(\mathbf{X}), \frac{\partial f}{\partial z}(\mathbf{X})\right)$.

Covariance: Let $(R, \mathbf{t}) \in \mathrm{SE}(3)$ be a rigid transformation. Using the change of variables $\mathbf{X}' = R\mathbf{X} + \mathbf{t}$, it is easy to check that $\nabla_{\mathbf{X}'} f = \nabla_{\mathbf{X}} f\, R^T$, meaning that the transposed gradient is a covariant quantity and transforms like an $\mathbb{R}^3$ vector.

Computation: For a locally planar surface, the gradient is obtained in 3D (point cloud, mesh, RGB-D image or implicit image as in NeRFs [20]) and in 2D by quadratic minimization. For 3D gradients, computations are carried out in the local tangent plane. This process is detailed in Appendix C.

3.2 Affine correspondences and the canonical reference frame

A correspondence between two locally planar surfaces in reference frames F_a around point $\mathbf{p_a} = z_a^0(u_a^0, v_a^0, 1)$ and F_b around point $\mathbf{p_b} = z_b^0(u_b^0, v_b^0, 1)$ is expressed by a homography $H_{ab} = R - \mathbf{t}\cdot\mathbf{n}^T/d_b$ where $R, \mathbf{t}$ is the transformation from F_a to F_b, $\mathbf{n}$ is the plane normal in F_b and d_b is the distance from the origin to the plane in F_b. Let $\mathbf{U_a} = (u_a, v_a)^T$ and $\mathbf{U_b} = (u_b, v_b)^T$ be normalized image coordinates of corresponding points in F_a and F_b. The depth of the associated points are respectively z_a and z_b. In the neighborhood N where surfaces are locally planar, we have:

$$\mathbf{U_a} = \frac{z_b}{z_a} \cdot H_{1:2,1:3} \cdot \tilde{\mathbf{U}}_\mathbf{b} \tag{1}$$

$$s(\mathbf{U_b}) = \frac{z_a}{z_b} = \mathbf{H_{3,1:3}} \cdot \tilde{\mathbf{U}}_\mathbf{b} = \mathbf{H_{3,1:2}} \cdot \mathbf{U_b} + H_{3,3} \tag{2}$$

where $s(\mathbf{U_b})$ is the scale (depth ratio) at point $\mathbf{U_b}$. $z_a = 0$ and $z_b = 0$ are degenerate cases where image gradients go to infinity, so from now on we assume $z_a \neq 0, z_b \neq 0$. This hypothesis is common to PnP approaches.

Then, writing a Taylor expansion of coordinates around $\tilde{\mathbf{U}}_\mathbf{a}^\mathbf{0} = (u_a^0, v_a^0, 1)$ which transforms to $\tilde{\mathbf{U}}_\mathbf{b}^\mathbf{0} = (u_b^0, v_b^0, 1)$ yields:

$$\mathbf{U_a} \simeq \mathbf{U_a^0} + \left.\frac{\partial \mathbf{U_a}}{\partial \mathbf{U_b}}\right|_0 \cdot (\mathbf{U_b} - \mathbf{U_b^0}) \tag{3}$$

where the affine matrix $A_f = \frac{\partial \mathbf{U_a}}{\partial \mathbf{U_b}}\big|_0$ can be expressed using a derivative of equation 1:

$$A_f = \frac{H_{1:2,1:2}}{s(\mathbf{U_b^0})} - \frac{H_{1:2,1:3} \cdot \tilde{\mathbf{U}}_\mathbf{b}}{s^2(\mathbf{U_b^0})} \cdot \frac{\partial s}{\partial \mathbf{U_b}}\bigg|_0 \tag{4}$$

$$A_f = \frac{1}{s(\mathbf{U_b^0})} \cdot \begin{pmatrix} H_{11} - H_{31}u_a^0 & H_{12} - H_{32}u_a^0 \\ H_{21} - H_{31}v_a^0 & H_{22} - H_{32}v_a^0 \end{pmatrix} \tag{5}$$

In the P1AC problem, A_f, $\mathbf{n}$, z_b, d_b, $\mathbf{U_a}$ and $\mathbf{U_b}$ are known while R, $\mathbf{t}$ and z_a^0 are unknown and should be determined.

Affine correspondences and gradients

Suppose that a pair of fields f_1, f_2 can be obtained at each point $\mathbf{U_a}$ and $\mathbf{U_b}$. Typically, f_1 and f_2 are individual color channels, whose value is known at each pixel of an RGB, RGB-D or multichannel image. We can express the gradient of the quantities as $\mathbf{g}^{\{\mathbf{a},\mathbf{b}\}}_{\{\mathbf{1},\mathbf{2}\}} = \nabla_{\{a,b\}} f_{\{1,2\}}$.

By definition of A_f and writing gradients as row vectors,

$$\begin{pmatrix} \mathbf{g_1^b} \\ \mathbf{g_2^b} \end{pmatrix} = \begin{pmatrix} \mathbf{g_1^a} \\ \mathbf{g_2^a} \end{pmatrix} A_f \tag{6}$$

Equation 6 allows computing A_f from a pair of corresponding gradients. Conversely, we can create a pair of corresponding gradients from A_f by taking arbitrary non-colinear $\mathbf{g_1^a}$ and $\mathbf{g_2^a}$, for instance $(1,0)$ and $(0,1)$, and computing the corresponding $\mathbf{g_1^b}$ and $\mathbf{g_2^b}$ using the equation. Thus, the affine formulation (one affine correspondence) and the gradient formulation (one correspondence, two gradients) are strictly equivalent. The following explanations use the gradient formulation.

The canonical reference frame Analyzing equation 5 reveals a strong coupling between rotation, translation and scale in A_f through the H matrix and the s component. The coupling between rotation and translation comes from the $\mathbf{t} \cdot \mathbf{n}^T / d_b$ term of the H matrix which causes the translation $\mathbf{t}$ to influence all nine terms of the matrix. Notice how, if the normal $\mathbf{n}$ was parallel to a coordinate axis (e.g., z), only three terms of H would depend on $\mathbf{t}$. So, we define a canonical reference frame F_c (not unique) for the problem as one for which $d_c = 1$ and $\mathbf{n_c}$ is along the z axis, so that $z_c = 1$ for all points in the neighborhood N. Such a frame is obtained by finding the local tangent plane equation in reference frame F_b and creating F_c with the normal as z axis (pointing from the origin of F_c towards U_b^0), arbitrary x and y axes to form an orthonormal basis, and an origin positioned exactly one unit away on z from the computation point U_b^0. In this referential, U_b^0 is at $(0,0,1)$ and H now writes:

$$H_{ac} = \begin{pmatrix} R_{11} & R_{12} & R_{13} - t_1 \\ R_{21} & R_{22} & R_{23} - t_2 \\ R_{31} & R_{32} & R_{33} - t_3 \end{pmatrix} \tag{7}$$

Expressing the affine transformation from F_c instead of F_b has a few key advantages. It removes d_b from calculations and causes the $\mathbf{t}$ term to disappear from everything but H_{13}, H_{23} and H_{33}. Now, only the third component t_3 of the translation is present in A_f, and only through the scale factor s referencing H_{33}. The transformation from F_b to F_c is fully known, so in the following derivations, we suppose $F_b = F_c$ without loss of generality. Instead of dealing with the translation component explicitly in s, we use the fact that when using F_c, $z_b = 1$ for all points of N, so $s = z_a$, the missing depth channel of the image.

A_f now writes:

$$A_f = \frac{1}{s} \begin{pmatrix} R_{11} - R_{31}u_a^0 & R_{12} - R_{32}u_a^0 \\ R_{21} - R_{31}v_a^0 & R_{22} - R_{32}v_a^0 \end{pmatrix} \tag{8}$$

$$= \frac{1}{s}(R_{1:2,1:2} - \mathbf{U_a} \cdot \mathbf{R_{3,1:2}}) \tag{9}$$

Using equation 6, we can write the three two-component equations of canonical P1AC:

$$s.\tilde{\mathbf{U}}_\mathbf{a} = R\tilde{\mathbf{U}}_\mathbf{b} + \mathbf{t} \tag{10}$$

$$s.\mathbf{g_i^b} = \mathbf{g_i^a}(R_{1:2,1:2} - \mathbf{U_a} \cdot \mathbf{R_{3,1:2}}), \quad i \in \{1,2\} \tag{11}$$

We call equation 10 the *translation* equation, as it contains the only occurrence of the translation. We call equations 11 the *gradient* equations. Notice how there are six equations in total, but seven degrees of freedom: three for R, three for $\mathbf{t}$ and one for s. Compared to typical P3P equations, the advantage of our P1AC expression is that only the depth s of one point is unknown, compared to three in P3P.

3.3 Canonical P1AC solution

The process of solving these equations goes as follows: first, we *upgrade* the *gradient* equations to three-component rotation expressions (equation 13), introducing two additional unknowns ϵ_i. This in turn allows us to remove R and the difficulties associated with $SO(3)$. We then compute the depth $s = z_a$ from the *upgraded* equations as well as the additional unknowns ϵ_i. This computation produces two solutions with positive depth in general, and we analyze the rare single-solution case. We then compute the rotation R from the *upgraded* equation and finally the translation from the *translation* equation.

Upgrading to a three-component rotation

Let $\mathbf{G_i} = (\mathbf{g_i^a}, -\mathbf{g_i^a} \cdot \mathbf{U_a})$ and $\mathbf{g_i} = \mathbf{g_i^b}$. Notice how $\mathbf{G_i}$ is constructed to be orthogonal to the camera (or projection) ray at any given point $\tilde{\mathbf{U}}_\mathbf{a}$. Now, equation 11 writes:

$$s.\mathbf{g_i} = \mathbf{G_i} R_{1:3,1:2}, \quad i \in \{1, 2\} \tag{12}$$

Let $\epsilon_i = (\mathbf{G_i}R)|_z, i \in \{1, 2\}$, i.e., the third component of $\mathbf{G_i}R$. Now,

$$\gamma_\mathbf{i} = (s.\mathbf{g_i}, \epsilon_i) = \mathbf{G_i}R. \tag{13}$$

The purpose of introducing the new unknowns ϵ_i is for equation 13 to become a simple rotation, allowing us to remove the nine unknowns (in matrix form, or three in a minimal $SO(3)$ representation such as angle-axis) of rotation R through the property $\forall \mathbf{X} \in \mathbb{R}^3, \|\mathbf{X}^T R\| = \|\mathbf{X}^T\|$.

Finding the depth $s = z_a$: Using 13:

$$\|\mathbf{G_i}\|^2 = \|\mathbf{G_i}R\|^2 = s^2\|\mathbf{g_i}\|^2 + \epsilon_i^2, \quad i \in \{1, 2\} \tag{14}$$

$$\|\mathbf{G_1} + \mathbf{G_2}\|^2 = \|(\mathbf{G_1} + \mathbf{G_2})R\|^2 \tag{15}$$

$$= s^2\|\mathbf{g_1} + \mathbf{g_2}\|^2 + (\epsilon_1 + \epsilon_2)^2 \tag{16}$$

combining the equations:

$$\epsilon_i = \pm\sqrt{\|\mathbf{G_i}\|^2 - s^2\|\mathbf{g_i}\|^2}, \quad i \in \{1, 2\} \tag{17}$$

$$\mathbf{G_1} \cdot \mathbf{G_2} = s^2\mathbf{g_1} \cdot \mathbf{g_2} + \epsilon_1 \cdot \epsilon_2 \tag{18}$$

leading to:

$$\mathbf{G_1} \cdot \mathbf{G_2} - s^2\mathbf{g_1} \cdot \mathbf{g_2} = \tag{19}$$

$$\pm\sqrt{(\|\mathbf{G_1}\|^2 - s^2\|\mathbf{g_1}\|^2)(\|\mathbf{G_2}\|^2 - s^2\|\mathbf{g_2}\|^2)} \tag{20}$$

Squaring to remove the root, reordering and using $D = s^2$, we can write:

$$A \cdot D^2 + B \cdot D + C = 0 \tag{21}$$

$$A = \|\mathbf{g_1}\|^2\|\mathbf{g_2}\|^2 - (\mathbf{g_1} \cdot \mathbf{g_2})^2 \tag{22}$$

$$B = 2(\mathbf{G_1} \cdot \mathbf{G_2})(\mathbf{g_1} \cdot \mathbf{g_2}) \tag{23}$$

$$- \|\mathbf{g_1}\|^2\|\mathbf{G_2}\|^2 - \|\mathbf{g_2}\|^2\|\mathbf{G_1}\|^2 \tag{24}$$

$$C = \|\mathbf{G_1}\|^2\|\mathbf{G_2}\|^2 - (\mathbf{G_1} \cdot \mathbf{G_2})^2 \tag{25}$$

Notice how equation 21 is a quartic equation in s, but only a quadratic equation in $D = s^2$.

Theorem 1 *The point depth s, if nonzero and $A \neq 0$, verifies: $s^2 = (-B - \sqrt{\Delta})/(2A)$ with $\Delta = B^2 - 4AC$. The second root $(-B + \sqrt{\Delta})/(2A)$ is extraneous (introduced by the squaring step).*

Proof First, eliminate the pathological case $s = 0$ for which gradients are not defined. Also eliminate the degenerate case $A = 0$ which corresponds to colinear $\mathbf{g_1}$ and $\mathbf{g_2}$, preventing the affine matrix from being invertible (infinite number of solutions).

Equation 21 required squaring of 20, meaning there is likely an extraneous solution introduced. To find it, we inject equations *before* the squaring operation back into the quadratic equation 21.

Remember that $D\|\mathbf{g_i}\|^2 + \epsilon_i^2 = \|\mathbf{G_i}\|^2$ and $D\mathbf{g_1} \cdot \mathbf{g_2} + \epsilon_1 \cdot \epsilon_2 = \mathbf{G_1} \cdot \mathbf{G_2}$. Next, substitute all instances of $\mathbf{G_i}$ with $\mathbf{g_i}, \epsilon_i$ and D in A, B and C. Notice that $B = -(AD + C/D)$. Since A, C, D are all positive thanks to the properties of the dot product, $B \leq 0$.

Write the discriminant $\Delta = B^2 - 4AC$ and remove B from the expression using the newly found identity $B = -(AD + C/D)$: $\Delta = A^2D^2 + C^2/D^2 - 2AC = (C/D - AD)^2 \geq 0$. $C/D - AD$ can be expressed as $\|\epsilon_2\mathbf{g_1} - \epsilon_1\mathbf{g_2}\|^2 \geq 0$.

The raw solutions to equation 21 are thus $D_1 = (-B - \sqrt{\Delta})/(2A)$ and $D_2 = (-B + \sqrt{\Delta})/(2A)$. $D_1 = D_2$ if and only if $C/D = AD$, which happens when $\epsilon_1\mathbf{g_2} = \epsilon_2\mathbf{g_1}$, a degenerate situation where either gradients are colinear (forbidden, inifinite number of solutions) or $\epsilon_1 = \epsilon_2 = 0$, the latter being discussed in the paragraph *Analysis of the single-solution case*. Since $C/D - AD \geq 0$, $\sqrt{\Delta} = C/D - AD$. Consequently, $D_1 = (-B + AD - C/D)/(2A) = D$ and $D_2 = (-B - AD + C/D)/(2A) = C/(AD)$. The solution we are looking for is the squared point depth $D = s^2$,

which is obtained with D_1. The root $D_2 = C/(AD)$ is extraneous. □

Recovering the third component ϵ_i: Once s has been obtained through Theorem 1, we can extract the missing component ϵ_i of the rotated $\mathbf{G_i}R$ from equations 14 and 18:

$$\epsilon_1 = \alpha\sqrt{\|\mathbf{G_1}\|^2 - s^2\|\mathbf{g_1}\|^2} \tag{26}$$

$$\epsilon_2 = \beta\sqrt{\|\mathbf{G_2}\|^2 - s^2\|\mathbf{g_2}\|^2} \tag{27}$$

with:

$$(\alpha, \beta) \in \{-1, 1\}^2, \tag{28}$$

$$\alpha \cdot \beta = \text{sign}(\mathbf{G_1} \cdot \mathbf{G_2} - s^2\mathbf{g_1} \cdot \mathbf{g_2}) \tag{29}$$

The two solutions (or four including negative depth) arise from the fact that the sign of the gradients in the z direction is lost in the projection operation. The sign constraint $\alpha \cdot \beta = \text{sign}(\mathbf{G_1} \cdot \mathbf{G_2} - s^2\mathbf{g_1} \cdot \mathbf{g_2})$ ensures that $\epsilon_1 \cdot \epsilon_2$ satisfies equation 18. The two valid combinations (α, β) and $(-\alpha, -\beta)$, yield two distinct poses in general unless $\epsilon_1 = \epsilon_2 = 0$.

Analysis of the single-solution case $\epsilon_i = 0$: One interesting case is when $\epsilon_1 = \epsilon_2 = 0$. This also causes $C/D = AD$ in the demonstration of Theorem 1. In this case, there is only one solution (or two including negative depth) to equation 27. $\epsilon_i = 0$ means $\mathbf{G_i}\mathbf{R_{1:3,3}} = 0$. In other words, $\mathbf{G_i}$ is orthogonal to the third axis of the rotation (z vector of F_b expressed in F_a). Now, assume that $\mathbf{g_1^a}$ and $\mathbf{g_2^a}$ are neither zero nor colinear. Then, $\begin{pmatrix}\mathbf{g_1^a}\\ \mathbf{g_2^a}\end{pmatrix}$ is a full rank matrix, meaning

$$\begin{pmatrix}\mathbf{g_1^a}\\ \mathbf{g_2^a}\end{pmatrix}\mathbf{V} = \mathbf{0} \iff \mathbf{V} = \mathbf{0} \tag{30}$$

$\mathbf{G_i}\mathbf{R_{1:3,3}} = 0$ can be written

$$\begin{pmatrix}\mathbf{g_1^a}\\ \mathbf{g_2^a}\end{pmatrix}\left(\mathbf{R_{1:2,3}} - \mathbf{U_a} \cdot R_{3,3}\right) = \mathbf{0} \tag{31}$$

so

$$\left(\mathbf{R_{1:2,3}} - \mathbf{U_a} \cdot R_{3,3}\right) = \mathbf{0}. \tag{32}$$

This expression contains two interesting facts:

1. The single-solution case does not depend on gradients or the affine matrix A_f, only on the surface being observed and on the rotation.
2. The constraint fixes one degree of freedom, meaning we have in general a family of solutions $\mathbf{R_{1:2,3}}$ parametrized by $R_{3,3}$: $\mathbf{R_{1:2,3}} = \mathbf{U_a} \cdot R_{3,3}$ and the constraint $\|\mathbf{R_{1:3,3}}\| = 1$.

If the out-of-plane component $R_{3,3}$ is zero, the single-solution case is impossible. Otherwise, let:

$$\mathbf{W} = \mathbf{U_a}\sqrt{1/(1 + \|\mathbf{U_a}\|^2)} \tag{33}$$

Each value of $\mathbf{U_a}$ generates two fully-determined rotation z vectors:

$$\mathbf{R_{1:3,3}} = \pm(\mathbf{W}, \sqrt{1 - \|\mathbf{W}\|^2})^T \tag{34}$$

$$= \pm\tilde{\mathbf{U}}_\mathbf{a}/\|\tilde{\mathbf{U}}_\mathbf{a}\| \tag{35}$$

The rotation matrix still has a single additional free degree of freedom which can be interpreted as an angle of rotation. Thus, for each projected pixel $\mathbf{U_a}$ in frame F_a, there are exactly two families of rotations parametrized by a single continuous degree of freedom for which P1AC/P1PGrad has a single positive depth solution. For all other rotations, there are either an infinity or exactly two positive depth solutions.

The one-solution case has a simple geometric meaning: it occurs if and only if $\mathbf{R_{1:3,3}}$ is along the *camera (or projection) ray* $\tilde{\mathbf{U}}_\mathbf{a}$, so exactly when R is a rotation around the projection ray at each point . For instance, if $\mathbf{U_a} = \mathbf{0}$, then $\mathbf{R_{1:3,3}} = (0, 0, \pm 1)^T$ (rotation around z). When rotating around the projection ray, equation 13 becomes a true gradient rotation, so $\mathbf{G_i}$ is the true, 3D gradient in the image frame F_a. Since $\mathbf{G_i}$ is (always by construction) orthogonal to $\tilde{U_a}$, then in the single solution case, the component of the gradient along the projection ray is zero, and the gradient is fully observable in the image. Outside of the single solution case, $\mathbf{G_i}$ is not a true gradient and the sign of the (true) gradient along the projection ray is lost in the projection operation.

Finding the rotation and translation: Following [8]:

$$\begin{pmatrix}\gamma_1\\ \gamma_2\\ \gamma_1 \times \gamma_2\end{pmatrix} = \begin{pmatrix}\mathbf{G_1}\\ \mathbf{G_2}\\ \mathbf{G_1} \times \mathbf{G_2}\end{pmatrix} R \tag{36}$$

$$R = \begin{pmatrix} \mathbf{G_1} \\ \mathbf{G_2} \\ \mathbf{G_1} \times \mathbf{G_2} \end{pmatrix}^{-1} \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_1 \times \gamma_2 \end{pmatrix} \tag{37}$$

Finally, $\mathbf{t}$ is obtained from equation 10.

Degenerate cases and accuracy loss: The solver loses accuracy when:

- $A \approx 0$: the two 3D gradients $\mathbf{g_1}, \mathbf{g_2}$ are close to colinear (or zero), meaning the affine matrix is almost singular and the number of solutions is infinite.
- $D \approx 0$: the derivative of a square root goes to infinity near zero, so the uncertainty on depth estimation is higher for very low values of $s = \sqrt{D}$.
- The matrix $\begin{pmatrix} \mathbf{G_1} \\ \mathbf{G_2} \\ \mathbf{G_1} \times \mathbf{G_2} \end{pmatrix}$ is close to singular, i.e., $\mathbf{G_1}$ and $\mathbf{G_2}$ are almost colinear (which is equivalent to $\mathbf{g_1^a}$ and $\mathbf{g_2^a}$ being almost colinear).

These cases all have a simple geometric explanation.

While testing the implementation, we found that terms within square roots may become slightly negative due to numerical precision issues, so we always clamp them to zero.

3.4 Algorithm summary

The complete solver is summarized in Algorithm 1. The P1PGrad and P1AG wrappers are summarized in Algorithms 2 and 3.

Complexity: The canonical P1AC solver requires computing seven dot products, one cross product, one matrix inversion (3×3), one matrix product, four square roots, one matrix–vector product and a few additions and subtractions of vectors and scalars. This is significantly simpler than the 3Q3 solver of [2], which requires solving a system of three quadratic equations in three unknowns, involving computing the roots of a degree-4 polynomial and multiple matrix inversions. Having at most two solutions (compared to four for P1AC) is also a major performance increase on CUDA and similar register-constrained systems where it can allow passing solutions via registers instead of global memory or reducing register pressure to increase occupancy and throughput. We successfully tested P1AG and P1PGrad on CUDA.

Algorithm 1 Canonical P1AC solver

Require: 2D point $\mathbf{p}$, non-colinear 2D gradients $\mathbf{g_1^a}, \mathbf{g_2^a}$, non-colinear canonical-frame gradients $\mathbf{g_1}, \mathbf{g_2}$, de-canonicalization transform M
Ensure: Up to 2 poses $(R, \mathbf{t})$

$\mathbf{G_i} \leftarrow (\mathbf{g_i^a}, -\mathbf{g_i^a} \cdot \mathbf{p})$ for $i \in \{1, 2\}$ ▷ augmented 2D gradients
$A \leftarrow \|\mathbf{g_1}\|^2 \|\mathbf{g_2}\|^2 - (\mathbf{g_1} \cdot \mathbf{g_2})^2$
$B \leftarrow 2(\mathbf{G_1} \cdot \mathbf{G_2})(\mathbf{g_1} \cdot \mathbf{g_2}) - \|\mathbf{g_1}\|^2 \|\mathbf{G_2}\|^2 - \|\mathbf{g_2}\|^2 \|\mathbf{G_1}\|^2$
$C \leftarrow \|\mathbf{G_1}\|^2 \|\mathbf{G_2}\|^2 - (\mathbf{G_1} \cdot \mathbf{G_2})^2$
$\Delta \leftarrow B^2 - 4AC$
$D \leftarrow (-B - \sqrt{\max(\Delta, 0)})/(2A)$ ▷ only valid solution. Also, clamp Δ for numerical precision issues
$s \leftarrow \sqrt{\max(D, 0)}$ ▷ numerical precision issues: clamp
$e_1 \leftarrow \|\mathbf{G_1}\|^2 - s^2\|\mathbf{g_1}\|^2$, $e_2 \leftarrow \|\mathbf{G_2}\|^2 - s^2\|\mathbf{g_2}\|^2$
for each sign pair (α, β) satisfying $\alpha \cdot \beta = \text{sign}(\mathbf{G_1} \cdot \mathbf{G_2} - s^2 \mathbf{g_1} \cdot \mathbf{g_2})$ **do**
    $\epsilon_1 \leftarrow \alpha\sqrt{\max(e_1, 0)}$, $\epsilon_2 \leftarrow \beta\sqrt{\max(e_2, 0)}$
    $\gamma_\mathbf{i} \leftarrow (s \cdot \mathbf{g_i}, \epsilon_i)$ for $i \in \{1, 2\}$
    $R \leftarrow \begin{pmatrix} \mathbf{G_1} \\ \mathbf{G_2} \\ \mathbf{G_1} \times \mathbf{G_2} \end{pmatrix}^{-1} \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_1 \times \gamma_2 \end{pmatrix}$
    $\mathbf{t} \leftarrow s \cdot (\mathbf{p^T}, \mathbf{1})^\mathbf{T} - R \cdot (0, 0, 1)^T$
    Output $(R, \mathbf{t})$ transformed by M
end for

3.5 P1PGrad with RANSAC

For images with more than three channels, it is possible to perform channel-wise RANSAC, selecting pairs of channels randomly and computing a P1PGrad for each pair. The solution $(s, R, \mathbf{t})$ which minimizes the 2D gradient reprojection error $\sum_i \|\mathbf{g_i} \cdot s - \mathbf{G_i} \cdot R_{1:3,1:2}\|^2$ is kept. RANSAC is useful when variations in a specific image channel around a point are unreliable, for instance because the gradient is too small. The number of RANSAC iterations required to consider all gradient pairs scales as N^2_{fields}, but in practice a fixed number of iterations (e.g., 50) provides good results. The translation $\mathbf{t}$ is not used in the reprojection error, since there is, by construction of equation 10, no translation error no matter the (non-colinear) gradient pair selected.

Algorithm 2 P1PGrad: from two gradient correspondences to pose

Require: 2D point $\mathbf{p}$, 3D point $\mathbf{X}$, non-colinear 2D gradients $\mathbf{g_1^a}, \mathbf{g_2^a}$, non-colinear 3D gradients $\mathbf{g_1^b}, \mathbf{g_2^b}$ (3 components each)
Ensure: Up to 2 poses $(R, \mathbf{t})$

▷ Build inverse canonical frame M from 3D gradients:
▷ F_b expressed in F_c, transforms points from F_b to F_c
$\mathbf{e_x} \leftarrow \mathbf{g_1^b}/\|\mathbf{g_1^b}\|$
$\mathbf{e_z} \leftarrow (\mathbf{g_1^b} \times \mathbf{g_2^b})/\|\mathbf{g_1^b} \times \mathbf{g_2^b}\|$
$\mathbf{e_y} \leftarrow \mathbf{e_z} \times \mathbf{e_x}$
$M_{rot} \leftarrow \text{rotation}(\mathbf{e_x}, \mathbf{e_y}, \mathbf{e_z})^T$ ▷ inverse on the fly
$M_{trans} \leftarrow \text{translation}(M_{rot} \cdot (\mathbf{e_z} - \mathbf{X}))$ ▷ on-the-fly affine matrix inversion gives $\mathbf{T}^{-1} = -R^T \cdot \mathbf{T}$
$M \leftarrow \begin{pmatrix} M_{rot} & M_{trans} \\ 0 & 1 \end{pmatrix}$
▷ Project 3D gradients to canonical frame (first 2 components):
$\mathbf{g_i} \leftarrow (\mathbf{g_i^b} \cdot M_{rot}^T)_{1:2}$ for $i \in \{1, 2\}$
▷ Call canonical solver:
return CanonicalP1AC($\mathbf{p}$, $\mathbf{g_1^a}$, $\mathbf{g_2^a}$, $\mathbf{g_1}$, $\mathbf{g_2}$, M)

Algorithm 3 P1AG: from an affine correspondence to pose

Require: 2D point $\mathbf{p}$, 3D point $\mathbf{X}$, normal $\mathbf{n}$, affine matrix A
Ensure: Up to 2 poses $(R, \mathbf{t})$

▷ Build inverse canonical frame M from 3D gradients:
▷ F_b expressed in F_c, transforms points from F_b to F_c
$\mathbf{e_z} \leftarrow \mathbf{n}$
$\mathbf{e_x} \leftarrow$ any vector orthogonal to $\mathbf{e_z}$
$\mathbf{e_y} \leftarrow \mathbf{e_z} \times \mathbf{e_x}$
$M_{rot} \leftarrow \text{rotation}(\mathbf{e_x}, \mathbf{e_y}, \mathbf{e_z})^T$
$M_{trans} \leftarrow \text{translation}(M_{rot} \cdot (\mathbf{e_z} - \mathbf{X}))$ ▷ on-the-fly affine matrix inversion gives $\mathbf{T}^{-1} = -R^T \cdot \mathbf{T}$
$M \leftarrow \begin{pmatrix} M_{rot} & M_{trans} \\ 0 & 1 \end{pmatrix}$
▷ Compute canonicalization affine matrix $J = \frac{\partial \mathbf{U_b}}{\partial \mathbf{U_c}}$:
$J \leftarrow \begin{pmatrix} M_{11} - M_{13}u & M_{21} - M_{23}u \\ M_{12} - M_{13}v & M_{22} - M_{23}v \end{pmatrix} / X_3$ with $(u, v) = \frac{\mathbf{X}_{1:2}}{X_3}$
▷ Giving canonical affine matrix $\frac{\partial \mathbf{U_a}}{\partial \mathbf{U_c}}$ by composition:
$A' \leftarrow A.J$
▷ Convert to two gradient pairs:
$\mathbf{g_1^a} \leftarrow (1, 0)$, $\mathbf{g_2^a} \leftarrow (0, 1)$
$\mathbf{g_1} \leftarrow A'_{1,1:2}$ ▷ $= \mathbf{g_1^a} \cdot A'$, from eq. 6
$\mathbf{g_2} \leftarrow A'_{2,1:2}$ ▷ $= \mathbf{g_2^a} \cdot A'$, from eq. 6
▷ Call canonical solver:
return CanonicalP1AC($\mathbf{p}$, $\mathbf{g_1^a}$, $\mathbf{g_2^a}$, $\mathbf{g_1}$, $\mathbf{g_2}$, M)

4 Experiments

4.1 Synthetic data

We first compare P1AG and P1PGrad with P1AC [2] and the P3P from [8] using the benchmark provided by [2]. This benchmark creates 20000 random test cases with a random rotation and translation, a 3D point X and its projection x. It generates the affine transformation $(H, A_f, R, \mathbf{t}, \mathbf{n}, d_b)$ by sampling a random normal and computing the homography aligning the planes in the 3D reference and query image. For P1PGrad, a pair of random non-colinear 3D gradients $(\mathbf{g_1^b}, \mathbf{g_2^b})$ orthogonal to said normal are generated in the reference image, as well as their 2D projection in the query image through A_f and equation 6. For P3P, two additional 3D points are generated as $(X + \mathbf{g_1^b}, X + \mathbf{g_2^b})$ and projected, allowing P3P to be directly compared to P1PGrad with vectors of similar magnitude. All algorithms use their official C++ Eigen implementation, are compiled with the same $O3$ optimization level and are executed on a single CPU core (intel core i7 7820HQ). The purpose of this test is to compare running time and numerical stability. We do not add noise since all three algorithms do not use exactly the same input data. Table 1 gives average results for all three algorithms and Figure 4 gives a histogram of rotation and translation errors.

We first note that P1AC is at least an order of magnitude slower than other approaches, due to its more complex implementation. P1PGrad/P1AG are marginally faster than the tested P3P, which is one of the fastest, if not the fastest P3P algorithm available. The average rotation and translation errors are orders of magnitude larger for P1AC, but the histograms (Figure 4)

	P3P [8]	P1AC [2]	P1AG (ours)	P1PGrad (ours)
n. sols* ↓	2.0419	3.9996	**2**	**2**
time (ns) ↓	467	5431	419	**392**
R_{err} (avg) ↓	2.4×10^{-6}	3.1×10^{-4}	$\mathbf{1.1\times10^{-12}}$	1.3×10^{-12}
$\mathbf{t}_{\text{err}}$ (avg) ↓	2.8×10^{-6}	2.3×10^{-4}	$\mathbf{1.8\times10^{-12}}$	2.1×10^{-12}
R_{err} (max) ↓	4.7×10^{-2}	3.1	7.0×10^{-9}	$\mathbf{5.8\times10^{-9}}$
$\mathbf{t}_{\text{err}}$ (max) ↓	5.6×10^{-2}	2.9	$\mathbf{9.9\times10^{-9}}$	1.3×10^{-8}

Table 1: Results on synthetic data from [2]. We evaluate P3P [8], P1AC [2] and our two solvers: P1AG and P1PGrad, in terms of computation time (nanoseconds), rotation R_{err} and translation $\mathbf{t}_{\text{err}}$ errors (radians and normalized units respectively) and number of solutions. * When solvers are able to remove negative-depth solutions early without having to compute them, we only count positive solutions and the time to compute these. The distinction is notably important on register-constrained systems such as CUDA where storing and returning a solution is expensive. Note that our solver can theoretically return less than two solutions, but this was never encountered during testing.

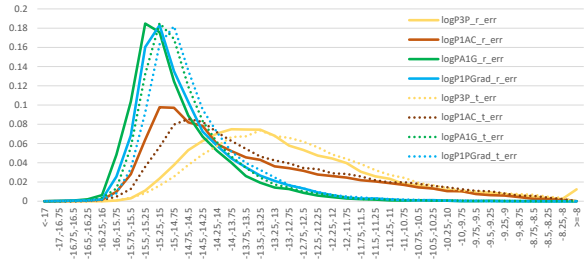


Fig. 4: Numerical stability of P3P [8] (yellow), P1AC [2] (red) and our two solvers: P1AG and P1PGrad (green, blue) on synthetic data from [2], histogram of error distribution. Continuous lines are rotation errors, dotted lines are translation errors. Results on P3P and P1AC are consistent with the findings of [2].

reveal that P1AC is more precise in log-average than P3P, with a few cases where the error is very large. P1AC is thus not as numerically stable as P3P, which may prove problematic in edge cases. P1PGrad/P1AG are many orders of magnitude more precise than both others in terms of average and worst case scenario, notably fixing the stability issue of P1AC. We believe that this is due to P1AC/P1PGrad being simple quadratic solvers while, for instance, P3P is cubic. It can be argued that a large fraction of the numerical accuracy of P1PGrad may have already been lost in the required gradient computations, which P3P does not have to perform. Similarly, computing an affine description introduces noise for P1AC/P1AG. However, P3P has to handle limited correspondence accuracy which none of the other solvers do.

Regarding the number of solutions returned, P1AC almost always returns four solutions and sometimes less, P3P almost always two solutions and sometimes more and P1PGrad/P1AG almost always two solutions with positive depth (and another two with negative depth) with a rare one-solution case (plus one with negative depth) discussed in the previous section. Having fewer solutions is desirable in practice since only one solution corresponds to the ground truth, all others being outliers which reduce the performance of a later robustification stage such as RANSAC or pose grouping [14, 21, 22]. Pose grouping approaches are in general more complex to set up but more robust to outliers than RANSAC.

The synthetic test was able to catch a discrepancy between our solver P1AG and the original P1AC [2] as P1AC missed at least one solution (n. sols = $3.9996 \neq 4$) which both our P1AG solver and our test harness declared valid (as in, when transforming 3D point and gradients to the 2D frame using the solution, the 2D point and 2D gradients are indistinguishable). While exceedingly rare, this event shows that the original P1AC can sometimes fail to compute a valid solution.

4.2 Sensitivity analysis of P1PGrad

P1AG and P1PGrad are direct solvers which always give either the correct solution on a single point, or an indistinguishable solution from the right one, when provided with the right input. As such, errors using one of these solvers come from either earlier or later stages of the computation, including hypotheses violation (non-planar surfaces), noisy gradient computations, incorrect correspondences, bad pose grouping/RANSAC, refinement or scoring. Pose grouping, refinement and scoring are highly approach-specific and cannot be tested simply. However, we can study the impact of the other error sources. For each error source, we carry 100 tests, progressively increasing the error and logging how the output poses of P1PGrad match the ground truth: number of poses with a rotation error less than 10° and average rotation and translation error of these poses. For each test, a 40×40 pixel 70° FOV virtual camera observes a plane, where each point of the plane has $N \geq 2$ field values. This forms the multichannel+depth "image". For the 3D reference, we resample the image uniformly. We do not apply any rotation or translation in 3D as these do not contribute to the errors measured given the late appearance of R and $\mathbf{t}$ in the calculations. Gradients are computed in 2D pixel space and 3D space. Perfect correspondences between 2D and 3D are taken unless specified otherwise. Figure 5 shows the influence of various parameters and inaccuracies on the result of P1PGrad. These include:

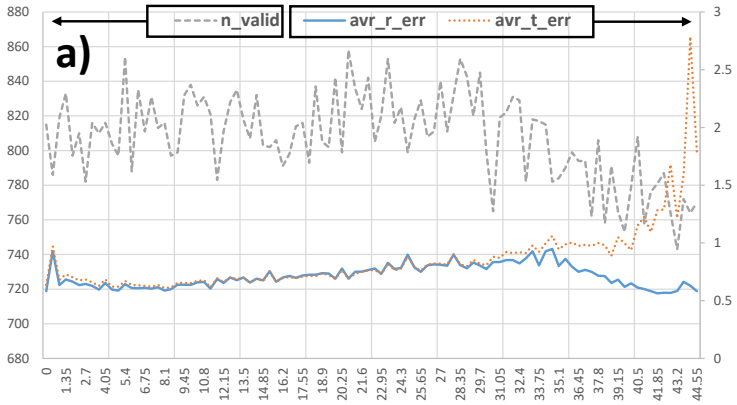
1. Plane slope: how much the normal to the plane differs from the optical axis, in degrees.
2. Descriptor noise: how much random additive noise is applied to each descriptor.
3. Correspondence accuracy: how precise correspondences are (less precise correspondences fall in a larger neighborhood of the ground truth correspondence on average).
4. Number of additional gradients: how many additional fields and corresponding gradients are used through RANSAC in addition to the two mandatory ones.
5. Planarity deviation: how much the plane has been deformed using radial distortion $dist(pt_{x,y}, cam) = (1 + \alpha \cdot (x^2 + y^2)) \cdot dist_{plane}(x, y)$ with α the planarity deviation.
6. Gradient computation support radius 2D: how large the 2D gradient computation neighborhood is.
7. Gradient computation support radius 3D: how large the 3D gradient computation neighborhood is.

Tests are carried on the 40×40 grid of points, generating 1600 pose hypotheses (one per point). When no noise is added, there are always statistically 800 valid rotations, which is half of that. Indeed, P1PGrad gives two solutions, of which only one corresponds to the ground truth. From a single point, there is no way to disambiguate which transformation is the right one.

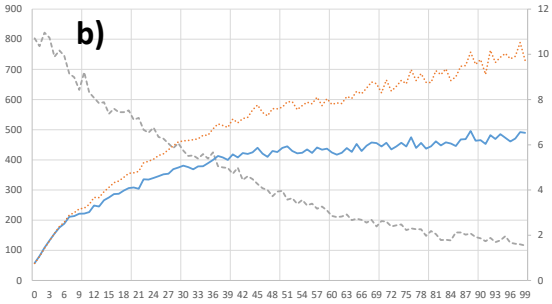
Plane slope: When the plane's normal angle with respect to the optical axis increases, the pitch between points in 3D varies in the image and the discrepancy between $\partial/\partial x_{\text{image}}$ and $\partial/\partial x_{\text{3D}}$ increases. We observe that the number of poses under 10° and average translation and rotation error of those remains constant until a plane angle of approximately 30°, after which the number of valid poses decreases linearly and the translation error increases more than linearly. This is expected, as the closer a surface is to the perspective lines created by the camera, the less reliable gradient computations become. The observed decrease of rotation errors is likely a computation artifact arising from the decrease of the number of valid poses, where remaining poses have better rotation accuracy.

Descriptor noise: When adding a uniform noise of increasing magnitude to the fields but keeping perfect correspondences, all metrics degrade smoothly. Assuming unbiased additive noise, increasing the gradient computation support radius in 2D and 3D would reduce the output error of P1PGrad by averaging the inputs, with an increased risk of violating the local planarity hypothesis in real applications.

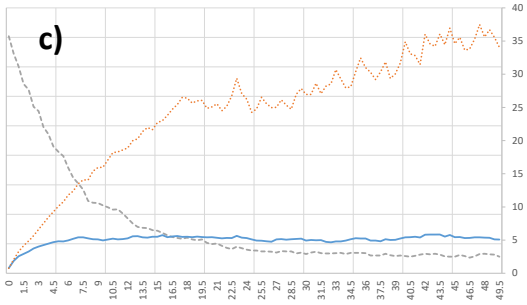
Correspondence noise: When sampling a correspondence for each image point, the corresponding 3D point is sampled randomly around the ground truth. When sampled further away, all metrics degrade. However, when correspondence noise reaches 20× 3D point sampling, the number of rotations under 10° stops decreasing while the translation error of poses under this threshold keeps increasing. This may be the result of the 40×40 pixel virtual sensor, where 20×



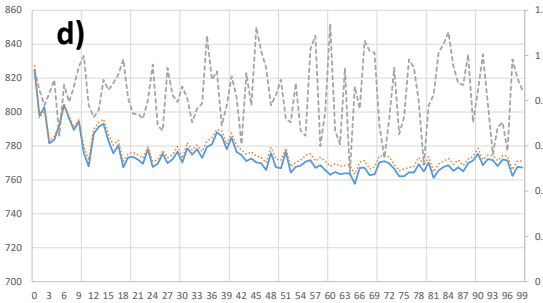
(a) plane slope (°)



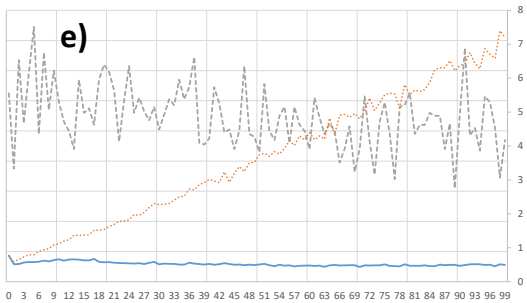
(b) descriptor noise (arbitrary units)



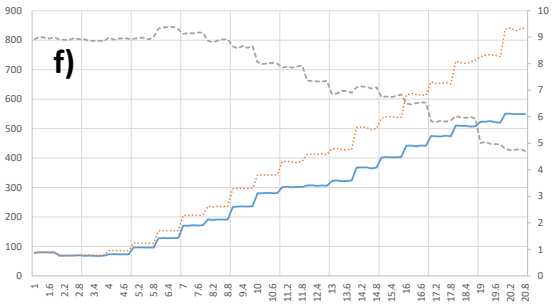
(c) correspondence accuracy (× sampling average)



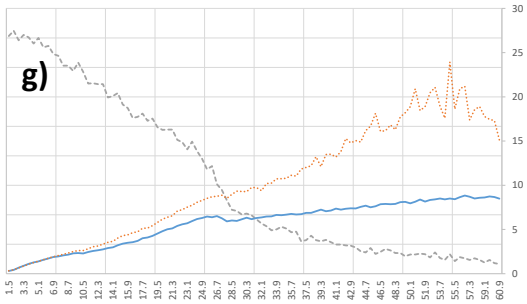
(d) number of additional gradients



(e) planarity deviation (arbitrary units)



(f) gradient computation support radius 2D (pixels)



(g) gradient computation support radius 3D (× sampling)

Fig. 5: Impact of various perturbations, inaccuracies and parameters on P1PGrad result quality. The number of rotations with an error lower than 10° is represented as n_{valid} on the left axis (higher is better). Translation errors of valid poses are scaled arbitrarily on the graphs in order to fit together with rotation errors of valid poses in degrees on the right axis (lower is better).

sampling is 20 pixels, meaning almost random correspondences in a 40×40 grid.

Number of fields: With more fields per point, and thus more gradients, the number of correct poses (under 10°) does not increase, but their precision does. This is due to RANSAC selecting two random gradients and testing the resulting pose against all gradients. With more fields comes the possibility of gradients of higher magnitude or whose direction is further apart, increasing precision. This effect can be observed up to 20 additional fields and becomes marginal after. Note that more fields should require more RANSAC steps to be treated properly, which we did not do in this experiment, where only 50 RANSAC steps are used regardless of the number of fields. The number of RANSAC steps required to consider all gradient pairs scales as N^2_{fields}.

Planarity deviation: Instead of sampling points from a plane $z = z_0$, we sample them from a parabola. Interestingly, the number of rotations under 10° barely changes, with only a slight decrease for extremely large deviations from planarity. The average rotation error remains lower than 1° and stable. Meanwhile, the translation error increases about linearly with the deviation from planarity. This hints at P1PGrad being way more robust in its rotation than its translation prediction. For this reason, when valid depth data is available from a depth sensor, it may be used to compute the pose's translation component instead.

2D gradient computation support radius: When computing the 2D gradient, we can change the image zone used as support. For radii from 1 to 5 pixels, we do not observe significant changes in metrics, which makes choosing a viable support radius easier. For the other experiments, a value of 2 pixels was chosen. This value ideally depends on the speed of variation of the fields in pixels. Computing gradients only works if the Shannon criterion is met, limiting field variation speed. One especially hard case is fields with high-frequency variations on non-planar objects.

3D gradient computation support radius: All metrics degrade smoothly when the 3D gradient calculation radius increases, which was expected since gradients are defined locally using a Taylor development. However, the radius should be large enough to compensate for descriptor noise (set to zero in this experiment). If an estimate of the objects' depth is available, trying to match the 3D and 2D gradient computation support radius and adapting the 2D radius to absorb descriptor noise while taking into account the Shannon criterion seems to be the best approach.

General observations: Even in the worst cases of each experiment, there were still around 6% of poses with an error less than 10°, meaning that a pose grouping/pose hypothesis RANSAC coupled with refinement (via ICP or Levenberg-Marquardt in descriptor-space for instance) and scoring may find correct poses even in noisy situations or when the planarity hypothesis is not met. However, the translation component degrades faster than the rotation component, prompting a use of a depth sensor whenever available. Preliminary experiments on real data, including rigid object registration on the BOP challenge and deformable object reference frame transfer, are reported in Appendices A and B.

5 Conclusion

We introduced a canonical frame transformation that simplifies the P1AC problem, reducing it to a single quadratic equation. The resulting solver, P1AG, is at least ten times faster and orders of magnitude more numerically accurate and stable than the 3Q3-based P1AC solver of [2]. We also showed that the P1AC problem can be formulated using only first-order derivatives (gradients) via the P1PGrad solver, which is strictly equivalent to P1AC and enables direct use of dense descriptor fields. Our degeneracy analysis characterizes all configurations where the solver does not produce its nominal two solutions and gives a geometrical interpretation thereof. In particular, the rare single-solution case is shown to occur for points where the local surface normal is along the camera (or projection) ray.

The key advantage of P1AG/P1PGrad is their *locality*: they use information from a single point and its neighborhood, unlike P3P which requires three spatially separated points. This makes them applicable to articulated, flexible, and symmetrical objects or object classes [3] where long-range geometric consistency is not guaranteed. Future work includes integration into end-to-end learned

pipelines where the solver's differentiability may enable gradient backpropagation.

Acknowledgments

We thank Boris Meden from CEA for advice on formulating early versions of the article.

Funding Statement

This work was financed in full by Université Paris-Saclay, CEA, List, F-91120 Palaiseau, France and received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data Availability Statement

Synthetic datasets were generated using an enhanced version of the data generator introduced by [2], which is publicly available at https://github.com/jonathanventura/P1AC; details on the added functionality are described in the text. Additionally, the evaluations in Appendix A utilize open-source datasets and metrics from the BOP challenge, accessible at https://bop.felk.cvut.cz/home/.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Köser, K., Koch, R.: Differential spatial resection - pose estimation using a single local image feature. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) Computer Vision – ECCV 2008, pp. 312–325. Springer, Berlin, Heidelberg (2008)

[2] Ventura, J., Kukelova, Z., Sattler, T., Baráth, D.: P1ac: Revisiting absolute pose from a single affine correspondence. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 19751–19761 (2023)

[3] Florence, P.R., Manuelli, L., Tedrake, R.: Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. In: 2nd Annual Conference on Robot Learning, CoRL 2018. Proceedings of Machine Learning Research, vol. 87, pp. 373–385. PMLR, ??? (2018)

[4] Haugaard, R.L., Buch, A.G.: Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6739–6748 (2022). https://doi.org/10.1109/CVPR52688.2022.00663

[5] Caron, M., Touvron, H., Misra, I., Jegou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9630–9640 (2021). https://doi.org/10.1109/ICCV48922.2021.00951

[6] Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., HAZIZA, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.-Y., Li, S.-W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning robust visual features without supervision. Transactions on Machine Learning Research (2024)

[7] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. In: International Journal of Computer Vision, vol. 60, pp. 91–110 (2004). https://doi.org/10.1023/B:VISI.0000029664.99615.94

[8] Ding, Y., Yang, J., Larsson, V., Olsson, C., Åström, K.: Revisiting the p3p problem. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4872–4880 (2023)

[9] Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o(n) solution to the pnp problem. International Journal of Computer Vision **81**(2), 155–166 (2009) https://doi.org/10.1007/s11263-008-0152-6

[10] Kneip, L., Li, H., Seo, Y.: Upnp: An optimal o(n) solution to the absolute pose problem with universal applicability. In: Fleet, D.,

Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014, pp. 127–142. Springer, Cham (2014)

[11] Baumberg, A.: Reliable feature matching across widely separated views. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 774–781 (2000). https://doi.org/10.1109/CVPR.2000.855899

[12] Köser, K., Beder, C., Koch, R.: Conjugate rotation: Parameterization and estimation from an affine feature correspondence. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008). https://doi.org/10.1109/CVPR.2008.4587796

[13] Eichhardt, I., Barath, D.: Relative pose from deep learned depth and a single affine correspondence. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020, pp. 627–644. Springer, Cham (2020)

[14] Chamisso, F., Meden, B., Tamaazousti, M.: Hspa: Hough space pattern analysis as an answer to local description ambiguities for 3d pose estimation. In: Proceedings of the 33rd British Machine Vision Conference (BMVC) (2022)

[15] Miraldo, P., Araujo, H.: A simple and robust solution to the minimal general pose estimation. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 2119–2125 (2014). https://doi.org/10.1109/ICRA.2014.6907150

[16] Mishkin, D., Radenovic, F., Matas, J.: Repeatability is not enough: Learning affine regions via discriminability. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)

[17] Tombari, F., Salti, S., Di Stefano, L.: In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) Unique Signatures of Histograms for Local Surface Description, pp. 356–369. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15558-1_26

[18] Petrelli, A., Di Stefano, L.: On the repeatability of the local reference frame for partial shape matching. In: 2011 International Conference on Computer Vision, pp. 2244–2251 (2011). https://doi.org/10.1109/ICCV.2011.6126503

[19] Hodaň, T., Michel, F., Brachmann, E., Kehl, W., Buch, A.G., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., Sahin, C., Manhardt, F., Tombari, F., Kim, T.-K., Matas, J., Rother, C.: Bop: Benchmark for 6d object pose estimation. In: Computer Vision – ECCV 2018, pp. 19–35. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01249-6_2

[20] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: Proceedings of the 16th European Conference on Computer Vision (ECCV) (2020)

[21] Mian, A., Bennamoun, M., Owens, R.: On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. International Journal of Computer Vision **89**(2), 348–361 (2010) https://doi.org/10.1007/s11263-009-0296-z

[22] Guo, Y., Sohel, F., Bennamoun, M., Lu, M., Wan, J.: Rotational projection statistics for 3d local surface description and object recognition. International Journal of Computer Vision **105**(1), 63–86 (2013) https://doi.org/10.1007/s11263-013-0627-y

[23] Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3d object recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2010)

[24] Zhang, R., Huang, Z., Wang, G., Liu, X., Zhang, C., Ji, X.: Gpose2023. In: BOP Challenge 2023 (2023). http://bop.felk.cvut.cz/method_info/410/

[25] Hodaň, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., Zabulis, X.: T-LESS: An

RGB-D dataset for 6D pose estimation of texture-less objects. IEEE Winter Conference on Applications of Computer Vision (WACV), 880–888 (2017) https://doi.org/10.1109/WACV.2017.103

[26] Obrist, J., Zamora, M., Zheng, H., Hinchet, R., Ozdemir, F., Zarate, J., Katzschmann, R.K., Coros, S.: Pokeflex: A real-world dataset of volumetric deformable objects for robotics. IEEE Robotics and Automation Letters **10**(10), 10586–10593 (2025) https://doi.org/10.1109/LRA.2025.3600148

A Application: rigid object registration on BOP

Although this paper is focused on the theory and numerical properties of our solver, we briefly demonstrate P1PGrad on the BOP challenge [19] to show that it is usable in a full pipeline and to highlight its unique capability of providing local reference frames. We use a SurfEmb model [4] to obtain dense descriptors, compute gradients on the descriptor fields, and use P1PGrad for pose hypothesis generation, followed by the pose grouping pipeline of [14]. When valid depth data is available, we use it for the translation component instead of P1PGrad, as our sensitivity analysis (Section 4.2) showed that translation is less robust than rotation under planarity deviation. Table 2 reports BOP AR scores. P1PGrad outperforms the purely geometric PPF approach [23] but underperforms SurfEmb [4] and GPose2023 [24]. This is expected: BOP objects are rigid and well-segmented, so non-local PnP methods benefit from long-range consistency, which is precisely where local methods like P1PGrad are at a disadvantage. The value of P1PGrad lies not in beating specialized BOP pipelines, but in its ability to transfer local reference frames to the image (Figure 2) — a capability that non-local methods cannot offer.

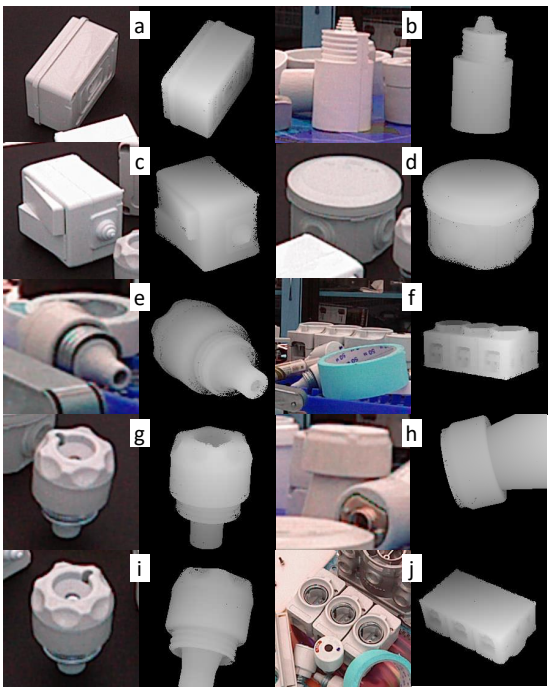


Fig. 6: Selected results from P1PGrad on TLESS [19, 25]. Each pair shows (observation, registration). Flat objects register well (a,c,d) even under occlusion (f). Objects without planar surfaces are more challenging: registration can be perfect (b, e), imprecise (g), or fail completely (h, i). Sometimes, even for flat and non-occluded objects, descriptors are incorrect due to symmetries or pose scoring falls into edge cases, leading to registration failures (j).

B Application: deformable object reference frame transfer

The key advantage of P1PGrad over non-local methods is its ability to provide local reference frames on deformable objects, where long-range geometric consistency is not guaranteed but local surface patches exhibit low deformation. There is currently no large-scale dataset of deformable objects with tracked points. Initiatives such as [26] still lack extensive training data such as BOP BlenderProc renders required to train [4]. We therefore show only a proof of concept on Figure 7: an RGB pattern printed on a sheet of paper simulates a dense descriptor field; the sheet is then deformed, and P1PGrad transfers reference frames from the flat to the deformed configuration using manually established matches. This demonstrates that P1PGrad can handle non-rigid transformations, which is fundamentally impossible for P3P-based approaches.

C Computing gradients

We want to compute the gradient of color channels or equivalent on 2D and 3D images. The following procedure is based on quadratic fitting of the mathematical definition of the gradient.

Let $(x, y, z, c) \to I_3^c(x, y, z) \in \mathbb{R}$ be a 3D surface with $c = 1 \dots C$ color channels. This can be for instance a point cloud, mesh, RGB-D image or implicit image as in NeRFs [20]. Consider the

	AVR	LMO	TLESS	TUDL	ICBIN	ITODD	HB	YCBV
PPF [23]	0.550	0.515	0.500	0.851	0.368	0.570	0.671	0.375
SurfEmb [4]	0.758	0.760	0.828	0.854	0.659	0.538	0.866	0.799
GPose2023 [24]	0.856	0.794	0.914	0.964	0.737	0.704	0.950	0.928
P1PGrad (Ours)	0.713	0.739	0.648	0.888	0.570	0.556	0.845	0.745

Table 2: BOP AR score (higher is better) on the seven core datasets plus overall average [19]. P1PGrad uses the same SurfEmb descriptors as [4].

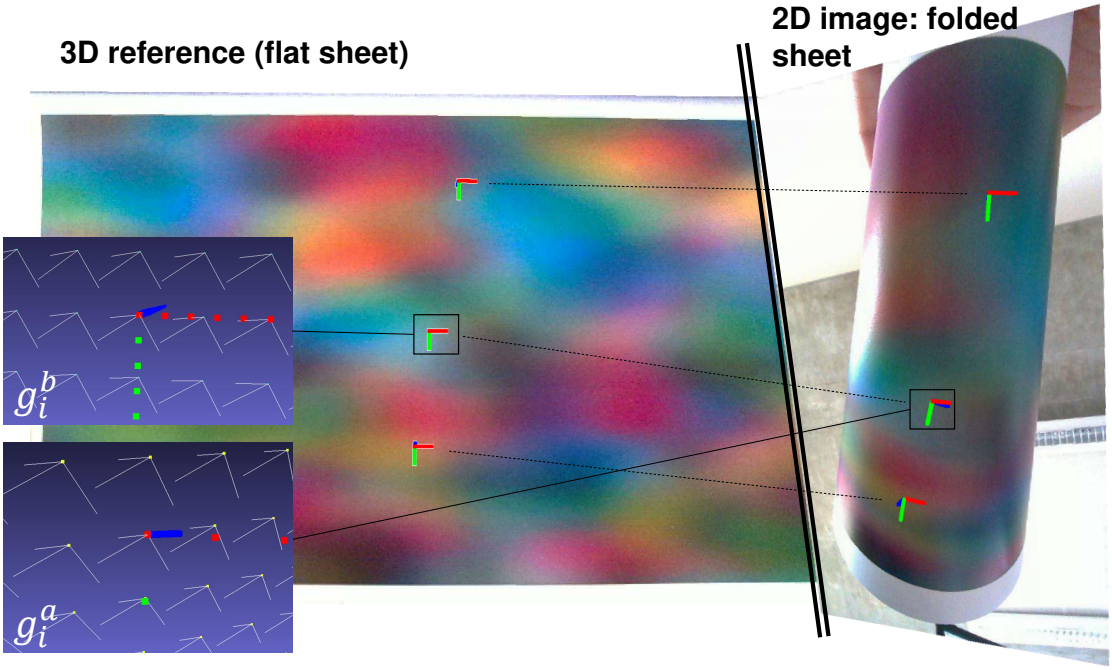


Fig. 7: P1PGrad can transfer reference frames from a 3D object (flat sheet of paper, left) to a deformed version of this object (folded sheet, right). Here, RGB values are used to compute gradients directly, which is not robust to lighting. Gradient fields for all three components (R, G, B) are represented in the bottom left.

neighborhood of point $\mathbf{X_0} = (x_0, y_0, z_0)$ with normal $\mathbf{n_0}$ and a given channel c. Let $Loc = (\mathbf{u}, \mathbf{v}, \mathbf{n_0})$ be an orthonormal local reference frame. We consider that I_3^c is locally planar: $(x, y, z) \in N(\mathbf{X_0}) \Rightarrow \mathbf{n_0} \cdot ((x, y, z) - (x_0, y_0, z_0)) \simeq 0$ where $N(\mathbf{X_0})$ is a 3D local neighborhood of $\mathbf{X_0}$, typically a small ball of radius δ_r.

Let $\mathbf{X} = (x, y, z) \in N$ be a point in the neighborhood of $\mathbf{X_0}$. $\mathbf{X}^* = (x^*, y^*)$ are the coordinates of $\mathbf{X}$ in the local reference frame defined by origin $\mathbf{X_0}$ and axes Loc. The third coordinate is equal to zero thanks to the planar assumption.

The first-order Taylor series writes $I_3^c(\mathbf{X}) = I_3^c(\mathbf{X_0}) + \nabla^*_{\mathbf{X_0}} \cdot (\mathbf{X}^* - \mathbf{X_0^*}) + o(\|\mathbf{X}^* - \mathbf{X_0^*}\|)$. Writing $\mathbf{\Delta X} = \mathbf{X}^* - \mathbf{X_0^*}$ and $\Delta I = I_3^c(\mathbf{X}) - I_3^c(\mathbf{X_0})$, we can find $\nabla^*_{\mathbf{X_0}}$ by quadratic minimization:

$$\nabla^*_{\mathbf{X_0}} = \underset{\nabla}{\operatorname{argmin}} \sum_{\mathbf{X} \in N} w(\|\mathbf{\Delta X}\|)(\Delta I - \nabla \cdot \mathbf{\Delta X})^2 \tag{38}$$

giving

$$\sum_{\mathbf{X}\in\mathbf{N}(\mathbf{X_0})} w(\|\mathbf{\Delta X}\|)\mathbf{\Delta X}\mathbf{\Delta X}^T\nabla^{T*}_{\mathbf{X_0}} \tag{39}$$

$$= \sum_{\mathbf{X}\in\mathbf{N}(\mathbf{X_0})} w(\|\mathbf{\Delta X}\|)\mathbf{\Delta X}\Delta I \tag{40}$$

where w is a sample weighting function, typically $w(\delta) = 1$, $1/\delta^2$ or $e^{-\delta^2/(2\sigma^2)}$. We use $w(\delta) = e^{-\delta^2/(2(\delta_r/3)^2)}$ in our calculations so that the weight has Gaussian decay. The $\sigma = \delta_r/3$ value is commonly used in image filtering, giving $w(\delta = \delta_r) \sim 1\%$.

Equation 40 can be solved directly if $\chi = \sum_{\mathbf{X}\in\mathbf{N}} w(\|\mathbf{\Delta X}\|)\mathbf{\Delta X}\mathbf{\Delta X}^T$ is invertible. We perform a simple determinant check and ignore points with zero determinant in the calculation, setting their descriptor value to *invalid*.

$\nabla^*_{\mathbf{X_0}}$ is the gradient in reference frame *Loc* and can be transformed back into original coordinates thanks to the covariance property of gradient vectors.

Computation of gradients on 2D images $(u, v, c) \rightarrow I^c_2(u, v) \in \mathbb{R}$ are carried out like their 3D counterparts, using a pixel neighborhood instead of a spatial neighborhood.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- canonicalP1AC.cpp