Jingnan Shi

Perspective-n-Point: P3P

Introduction

The Perspective-n-Point (PnP) problem is the problem of estimating the relative pose (six degrees of freedom) between an object and the camera, given a set of correspondences between 3D points and their projections on the image plane. It is a fundamental problem that was first studied in the photogrammetry literature, and later on studied in the context of computer vision.

In this post, I will present a few solvers (among many), discuss their proofs and also show some concise implementations. I will focus on the minimal solvers — solutions to the PnP problem that requires the minimal amount of information. In this case, we need at least three pairs of correspondences, and the minimal solvers that only require three pairs of correspondences are called P3P solvers.

Preliminaries

So, what is a PnP problem exactly? To understand this, let's first look at the P3P problem. Fig. 2 shows the typical geometry of a P3P problem instance. P_1, P_2 and P_3 are three known 3D points in the fixed world frame, and P_1^I, P_2^I and P_3^I are their projections on the image plane respectively. O is the camera's optical center, also known as the center of projection. Note that we assume a calibrated pinhole camera: that is we know the camera's focal lengths and distortion coefficients, and the image points P_1^I, P_2^I and P_3^I are calibrated rays with two degrees of freedom each.

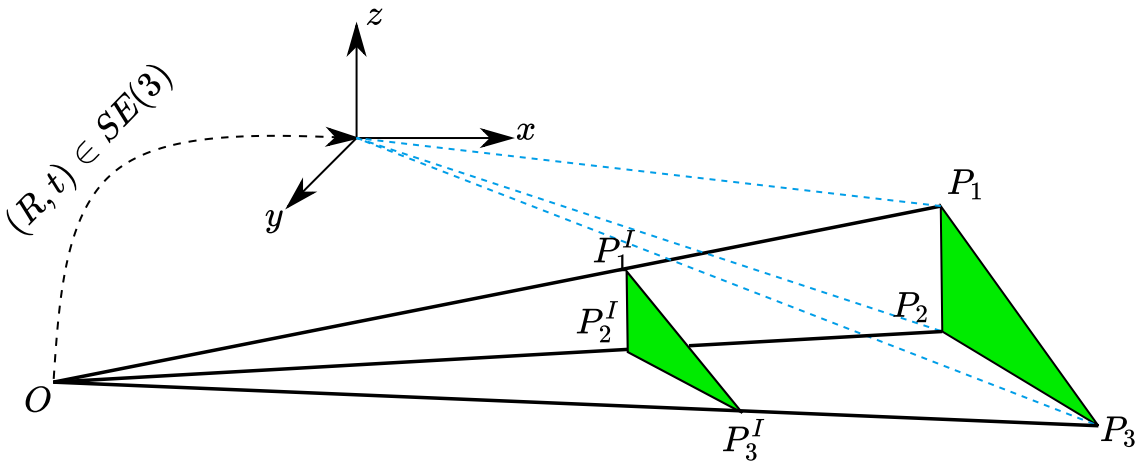

Figure 1: The geometry of the P3P problem.

The goal of the P3P problem is: find the P_1, P_2 and P_3's coordinates in the camera frame. And PnP problems are the extended version: find P_i, $i = 1 \dots n$'s coordinates in the camera frame. To put it in a more algebraic form, we have the following equation:

$$P_i^C = R_W^C P_i + t^C \qquad i = 1 \dots 3$$

where R^C and t^C are the relative transformation we are trying to solve (so that we can recover P_i^C, the positions of the points in the camera's frame). We know the projections of P_i on the camera's pixel space.

Interestingly, the early solutions to this problem solve for P_i^C directly, then find the R^C and t^C using Arun's Method. For them, the problem of P3P boils down to solving the geometry of a tetrahedron. The first three methods introduced in this pose belong to this category.

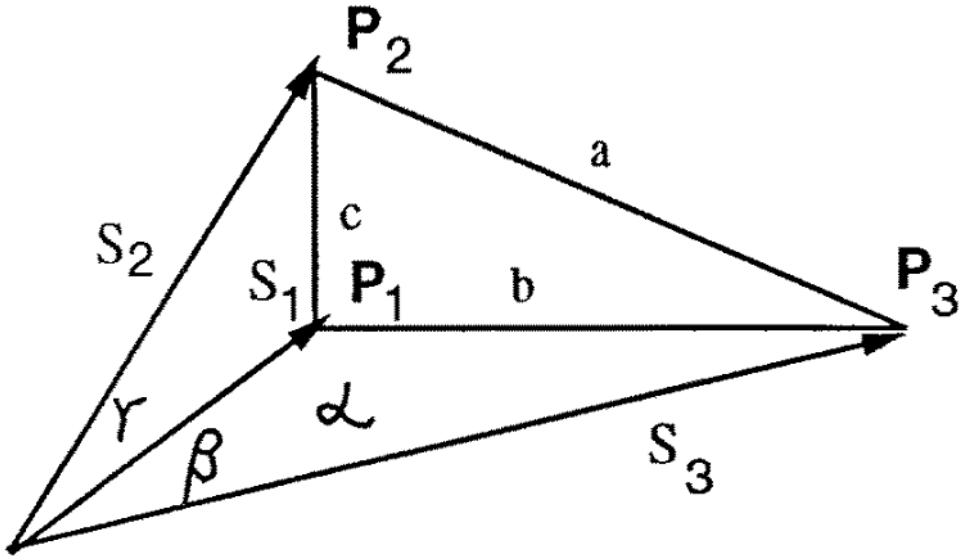

Figure 2: The P3P tetrahedron; P_1, P_2 and P_3 are the unknown 3D points; a, b and c are the known triangle sides, and the angles α, β and γ are also known. We need to solve for the lengths s_1, s_2, and s_3. (credit: Haralick, Bert M., et al. "Review and analysis of solutions of the three point perspective pose estimation problem." International journal of computer vision 13.3 (1994): 331-356.).

We know the lengths of the three bases (a, b and c), where

$$a = ||P_2 - P_3||$$
$$b = ||P_1 - P_3||$$
$$c = ||P_1 - P_2||$$

Let the (calibrated) projections of P_1, P_2 and P_3 be

$$q_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \quad \text{where } i = 1, 2, 3$$

and the vectors pointing to the vertices of the bases from the optical center are therefore

$$P_i = s_i j_i = \frac{s_i}{\sqrt{u_i^2 + v_i^2 + 1}} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \quad \text{where } i = 1, 2, 3$$

where s_i are the lengths of these vectors, and j_i are unit vectors.

We also define the following three angles (for convenience):

$$\cos\alpha = j_2 \cdot j_3$$
$$\cos\beta = j_1 \cdot j_3$$
$$\cos\gamma = j_1 \cdot j_2$$

With these information, we need to calculate the lengths of these vectors (and the coordinates of the vertices P_1, P_2 and P_3 of the bases come for free).

It is also possible to find R^C and t^C directly using linear algebra and geometry. The last method introduced in this post does this, and is perhaps the more widely used solver for P3P out there.

Grunert's P3P

Grunert appears to present the first solution to the P3P problem in his 1841 work, *Das Pothenotische Problem in erweiterter Gestalt nebst Über seine Anwendungen in der Geodäsie*.

Using law of cosines, we have

$$\begin{aligned} s_1^2 + s_2^2 - 2s_1s_2\cos\gamma = c^2 \\ s_1^2 + s_3^2 - 2s_1s_3\cos\beta = b^2 \\ s_2^2 + s_3^2 - 2s_2s_3\cos\alpha = a^2 \end{aligned} \tag{1}$$

And the problem becomes how to solve for s_1, s_2 and s_3 given the three equations above. Grunert employed a simple strategy of variable substitution: replacing s_2 and s_3 with $s_2 = us_1$ and $s_3 = us_2$. Using this substitution, the three equations above become:

$$\begin{aligned} s_1^2 &= \frac{c^2}{1 + u^2 - 2u\cos\gamma} \\ s_1^2 &= \frac{b^2}{1 + v^2 - 2v\cos\beta} \\ s_1^2 &= \frac{a^2}{u^2 + v^2 - 2uv\cos\alpha} \end{aligned}$$

We then combine the equations containing γ and β, and β and α to get:

$$u^2 - \frac{c^2}{b^2}v^2 + 2v\frac{c^2}{b^2}\cos\beta - 2u\cos\gamma + \frac{b^2 - c^2}{b^2} = 0 \tag{2}$$

and

$$u^2 + \frac{b^2 - a^2}{b^2}v^2 - 2uv\cos\alpha + \frac{2a^2}{b^2}v\cos\beta - \frac{a^2}{b^2} = 0 \tag{3}$$

From the Eq. 3, we can get an expression form u^2:

$$u^2 = -\frac{b^2 - a^2}{b^2}v^2 + 2uv\cos\alpha - \frac{2a^2}{b^2}v\cos\beta + \frac{a^2}{b^2} \tag{4}$$

And substituting it back to Eq. 2, we have an expression for u in terms of v:

$$u = \frac{\left(-1 + \frac{a^2-c^2}{b^2}\right)v^2 - 2\left(\frac{a^2-c^2}{b^2}\right)\cos\beta v + 1 + \frac{a^2-c^2}{b^2}}{2(\cos\gamma - v\cos\alpha)}$$

This expression for u can then be substitute back to Eq. 3 to get a polynomial in v:

$$A_4v^4 + A_3v^3 + A_2v^2 + A_1v + A_0 = 0 \tag{5}$$

where

$$
\begin{aligned}
A_4 =& \left(\frac{a^2-c^2}{b^2}-1\right)^2 - \frac{4c^2}{b^2}\cos^2\alpha \\
A_3 =& 4\left[\frac{a^2-c^2}{b^2}\left(1-\frac{a^2-c^2}{b^2}\right)\cos\beta - \left(1-\frac{a^2+c^2}{b^2}\right)\cos\alpha\cos\gamma + 2\frac{c^2}{b^2}\cos^2\alpha\cos\beta\right] \\
A_2 =& 2\left[\left(\frac{a^2-c^2}{b^2}\right)^2 - 1 + 2\left(\frac{a^2-c^2}{b^2}\right)^2\cos^2\beta + 2\left(\frac{b^2-c^2}{b^2}\right)\cos^2\alpha - 4\left(\frac{a^2+c^2}{b^2}\right)\cos\alpha\cos\beta\cos\gamma + 2\left(\frac{b^2-a^2}{b^2}\right)\cos \right. \\
A_1 =& 4\left[-\left(\frac{a^2-c^2}{b^2}\right)\left(1+\frac{a^2-c^2}{b^2}\right)\cos\beta + \frac{2a^2}{b^2}\cos^2\gamma\cos\beta - \left(1-\left(\frac{a^2+c^2}{b^2}\right)\right)\cos\alpha\cos\gamma\right] \\
A_0 =& \left(1+\frac{a^2-c^2}{b^2}\right)^2 - \frac{4a^2}{b^2}\cos^2\gamma
\end{aligned}
$$

Eq. 5 can have as many as four real solutions for v. With each value of v, we can easily recover the value of u, which will then give us the values for s_1, s_2 and s_3.

Note that there exists an algebraic singularity in this solution: when the denominator in Eq. 4 equals to zero. This will happen if $\alpha = \beta = \gamma = 60\,\text{deg}$, and $s_1 = s_2 = s_3$.

Fischler and Bolles' P3P

Fischler and Bolles also presented a P3P solution in their seminal paper *Random Sample Consensus: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. Essentially, their derivation is similar to Grunert's, except that they used a different set of algebraic manipulations to arrive at the final fourth order polynomial. In addition, their solution **does not** have an algebraic singularity.

Starting from the same P3P tetrahedron,

$$
\begin{aligned}
s_1^2 &= \frac{c^2}{1+u^2-2u\cos\gamma} \\
s_1^2 &= \frac{b^2}{1+v^2-2v\cos\beta} \\
s_1^2 &= \frac{a^2}{u^2+v^2-2uv\cos\alpha}
\end{aligned}
$$

they combined the second and third equations to get

$$
v^2\left(1-\frac{a^2}{b^2}\right) + 2v\left(\frac{a^2}{b^2}\cos\beta - u\cos\alpha\right) + u^2 - \frac{a^2}{b^2} = 0 \tag{6}
$$

and they combined the first and third equations to get

$$
v^2 + 2v(-\cos\alpha u) + u^2\left(1-\frac{a^2}{c^2}\right) + 2u\frac{a^2}{c^2}\cos\gamma - \frac{a^2}{c^2} = 0 \tag{7}
$$

With Eq 6 and 7, they observed that they can eliminate v by performing a few multiplications and substractions. Multiply Eq. 6 by $\left(1-\frac{a^2}{c^2}\right)u^2 + \frac{2a^2}{c^2}\cos\gamma u - \frac{a^2}{c^2}$, Eq. 7 by $u^2 - \frac{a^2}{b^2}$, and subtract, we have

$$
\begin{aligned}
&\left[\left(a^2-b^2-c^2\right)u^2 + 2\left(b^2-a^2\right)\cos\gamma u + \left(a^2-b^2+c^2\right)\right]v \\
&+2b^2\cos\alpha u^3 + \left(2\left(c^2-a^2\right)\cos\beta - 4b^2\cos\alpha\cos\gamma\right)u^2 \\
&+\left[4a^2\cos\beta\cos\gamma + 2\left(b^2-c^2\right)\cos\alpha\right]u - 2a^2\cos\beta = 0
\end{aligned} \tag{8}
$$

Multiply Eq. 7 by $1-\frac{a^2}{b^2}$, and subtract from Eq. 6, we have

$$2c^2(\cos\alpha u-\cos\beta)v+\left(a^2-b^2-c^2\right)u^2+2\left(b^2-a^2\right)\cos\gamma u+a^2-b^2+c^2=0 \tag{9}$$

Then, we multiply Eq. 8 by $2c^2(\cos\alpha u-\cos\beta)$, and Eq. 9 by $\left[\left(a^2-b^2-c^2\right)u^2+2\left(b^2-a^2\right)\cos\gamma u+\left(a^2-b^2+c^2\right)\right]$, and subtract them to get

$$B_4u^4+B_3u^3+B_2u^2+B_1u+D_0=0 \tag{10}$$

where

$$\begin{aligned}
B_4=&4b^2c^2\cos^2\alpha-\left(a^2-b^2-c^2\right)^2\\
B_3=&-4c^2\left(a^2+b^2-c^2\right)\cos\alpha\cos\beta\\
&-8b^2c^2\cos^2\alpha\cos\gamma\\
&+4\left(a^2-b^2-c^2\right)\left(a^2-b^2\right)\cos\gamma\\
B_2=&4c^2\left(a^2-c^2\right)\cos^2\beta\\
&+8c^2\left(a^2+b^2\right)\cos\alpha\cos\beta\cos\gamma\\
&+4c^2\left(b^2-c^2\right)\cos^2\alpha\\
&-2\left(a^2-b^2-c^2\right)\left(a^2-b^2+c^2\right)\\
&-4\left(a^2-b^2\right)^2\cos^2\gamma\\
B_1=&-8a^2c^2\cos^2\beta\cos\gamma\\
&-4c^2\left(b^2-c^2\right)\cos\alpha\cos\beta\\
&-4a^2c^2\cos\alpha\cos\beta\\
&+4\left(a^2-b^2\right)\left(a^2-b^2+c^2\right)\cos\gamma\\
B_0=&4a^2c^2\cos^2\beta-\left(a^2-b^2+c^2\right)^2
\end{aligned}$$

As mentioned at the beginning, this solution to P3P does not have an algebraic singularity.

Gao's P3P

The above two approaches are similar in the way of problem formulations: they both start with the law of cosines, and try to solve a system of polynomial solutions. A natural (perhaps more theoretical) extension is therefore how to systematically analyze what are all the possible solutions given the law of cosines equations. The paper *Complete Solution Classification for the Perspective-Three-Point Problem* by Xiao-Shan Gao et. al. does exactly this.

They start from the same system of equations (Eq. 1), and add the following "reality conditions", which correspond to geometrically nondegenerate solutions:

$$\begin{aligned}
&s_1>0,s_2>0,s_3>0,a>0,b>0,c>0,a+b>c,a+c>b,b+c>a\\
&0<\alpha,\beta,\gamma<\pi,0<\alpha+\beta+\gamma<2\pi\\
&\alpha+\beta>\gamma,\alpha+\gamma>\beta,\gamma+\beta>\alpha\\
&I_0=(2\cos\alpha)^2+(2\cos\beta)^2+(2\cos\gamma)^2-8\cos\alpha\cos\beta\cos\gamma-1\neq0
\end{aligned}$$

They then proceed to simplify Eq. 1 by introducing x,y,z,r,p,q, where $x=s_1/s_3$, $y=s_2/s_3$, $v=(c/s_3)^2$, $r=2\cos\gamma$, $q=2\cos\beta$, $p=2\cos\alpha$, $a'=(a/s_3)^2/v$, $b'=(b/s_3)^2/v$:

$$
\begin{aligned}
y^2 + 1 - yp - a'v = 0 \\
x^2 + 1 - xq - b'v = 0 \\
x^2 + y^2 - xyr - v = 0
\end{aligned}
$$

Then they eliminate v by substitution, reaching

$$
\begin{aligned}
(1-a)y^2 - ax^2 - py + arxy + 1 = 0 \\
(1-b)x^2 - by^2 - qx + brxy + 1 = 0
\end{aligned}
$$

which are two quadratic equations, and therefore the P3P problem can only have either an infinite number of solutions or at most four physical solutions. For convenience, I will refer to this two equations as ES for the rest of this section.

To solve this two quadratic equations, they employ a method called Wu-Ritt's zero decomposition. Essentially, it is a method for solving system of quadratic equations by first decomposing it into a series of equations in triangular forms:

$$
f_1\left(u, x_1\right)=0, f_2\left(u, x_1, x_2\right)=0, \ldots, f_p\left(u, x_1, \ldots, x_p\right)=0
$$

where u are a set of parameters and x_i are the variables we are trying to solve. And the zero set of the original polynomial system is the union of zero sets of these triangular forms.

Using Wu-Ritt's zero decomposition method, Gao and his colleagues decompose $Zero(ES/I_0)$ (which are the zero set of ES such that $I_0 \neq 0$) into 10 disjoint components:

$$
\operatorname{Zero}\left(ES / I_0\right)=\bigcup_{i=1}^{10} \mathbf{C}_i
$$

where $\mathbf{C}_i=\operatorname{Zero}\left(TS_i / T_i\right), i=1, \cdots, 9$ and
$C_{10}=\operatorname{Zero}\left(TS_{10} / T_{10}\right) \cup \operatorname{Zero}\left(TS_{11} / T_{11}\right)$. TS_i are polynomials in triangular form and T_i are polynomials.

If you want to see the exact expressions of these 10 components, you need to refer to their original paper. However, the key insight is that because the 10 components are disjoint, the solution therefore has to fall within 1 component only. In addition, within each component the parameters a', b', p, q and r have to satisfy some algebraic conditions, which may correspond to the geometrically degenerate cases of P3P. Take TS_4 as an example:

$$
\begin{aligned}
&\left(p^2 b+q^2 b-p^2\right) x^2+\left(-4 b q+p^2 q\right) x+4 b-p^2 \\
&p y+q x-2 \\
&a'+b'-1 \\
&r
\end{aligned}
$$

The equation $a'+b'-1=0$ correspond to the case where $\angle P_1 P_3 P_2=90^{\circ}$. Thus to get the final solution, you just need to check all the 10 components and see which one is satisfied. In practice, the two implementations available for this

method only return the solution to the main branch (see OpenCV's implementation and OpenGV's implementation).

Kneip P3P

Laurent Kneip in his 2011 paper *A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation* proposes yet another P3P solution. The unique part of his method comparing to the aforementioned 3 methods is that it directly solves for the rotation and translation through a clever parametrization of the problem.

Let's go over this method step by step. First, we need to define two new reference frames: one oriented on the bearing vectors of the three known projections of the three 3D points (τ), and one oriented on the three 3D points directly (η).

In the following section, to avoid confusion with the paper, we define $\vec{f}_1 = \vec{s}_1$, $\vec{f}_2 = \vec{s}_2$ and $\vec{f}_3 = \vec{s}_3$. Define τ's axes as

$$\begin{aligned}\vec{t}_x &= \vec{f}_1 \\ \vec{t}_z &= \frac{\vec{f}_1 \times \vec{s}_2}{\left\|\vec{f}_1 \times \vec{f}_2\right\|} \\ \vec{t}_y &= \vec{t}_z \times \vec{t}_x.\end{aligned}$$

and τ's origin is the camera center in the world frame. If we define $T = \left[\vec{t}_x, \vec{t}_y, \vec{t}_z\right]^T$, then the bearing vectors of the input 2D points (in the camera's frame) can be transformed into τ by:

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

Define η's axes as

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

and with its origin be P_1. Note that these axes and the origin are in the fixed world frame. If we define $N = \left[\vec{n}_x, \vec{n}_y, \vec{n}_z\right]^T$, then it follows that we can transform the world points P_i into η as:

$$P_i^{\eta} = N(P_i - P_1)$$

Note that the condition $\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3}$ has to be satisfied (P_1, P_2 and P_3 not colinear).

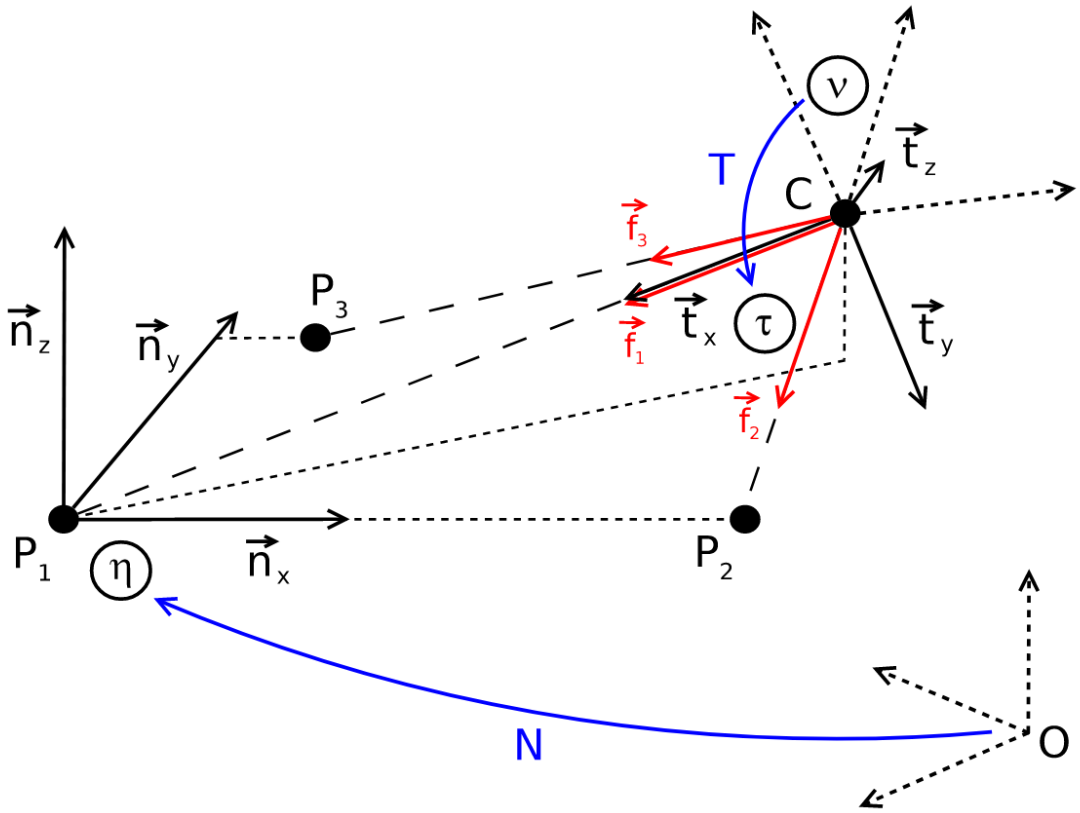

Figure 3: Illustration of η and τ. Note that f_1, f_2 and f_3 are equivalent with our previously defined s_1, s_2 and s_3. (credit: Kneip, Laurent, Davide Scaramuzza, and Roland Siegwart. "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation." CVPR 2011. IEEE, 2011.)

After we have the two intermediate frames, we only need to find the transformation between them to recover the relative transformation between the fixed world frame and the camera frame. The rest of the method therefore focuses on finding the transformation between η and τ.

To identify the transformation, we need to find the coordinates of C (origin of the τ frame) in the η frame, as well as a rotation matrix that align the two sets of axes. First, let's find the coordinates of C in η. Note that if we construct a plane Π containing the points P_1, P_2 and C (Fig. 4), we can describe C's coordinate on that plane based on the distance between P_1 and P_2, the angle between f_1 and f_2, which are both known and fixed, and lastly the angle between P_1C and P_1P_2, which is unknown. And together with the angle formed between the plane Π and the plane spanned by $\vec{n}_x$ and $\vec{n}_y$, we can write down an expression for C in η frame.

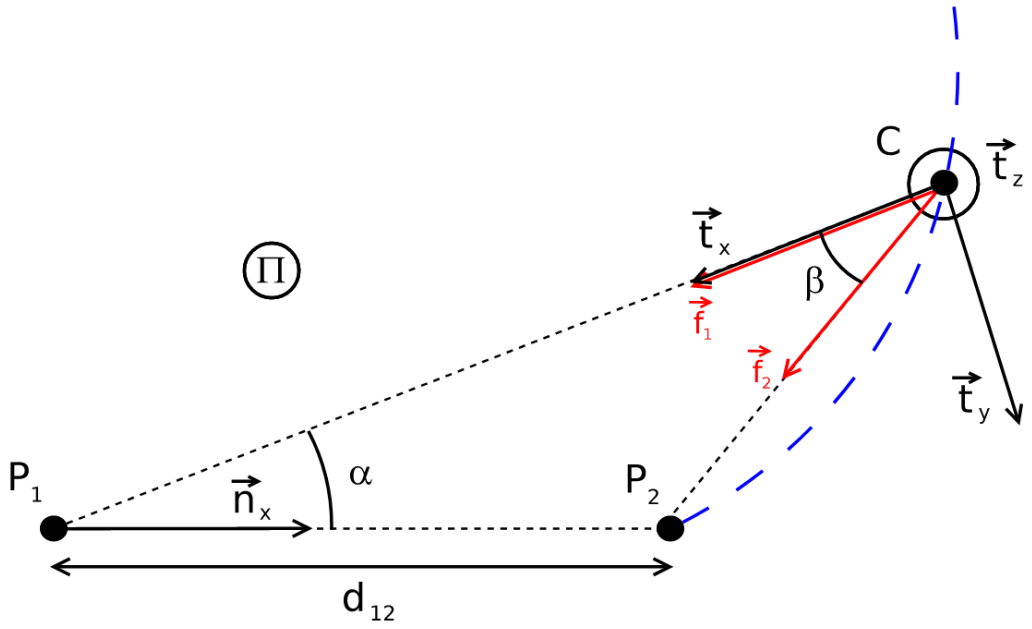

Figure 4: Plane Π *containing the points* P_1, P_2 *and* C. *(credit: Kneip, Laurent, Davide Scaramuzza, and Roland Siegwart. "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation." CVPR 2011. IEEE, 2011.)*

So it follows that we can define a b such that

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

and $\cos\beta = \vec{f}_1 \cdot \vec{f}_2$. Then, define $\alpha = \angle P_2 P_1 C$, and we have

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

and the position of C in Π's frame (with P_1 as the origin, $\vec{n}_x$ as the x-axis, z-axis perpendicular on the plane) is therefore

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

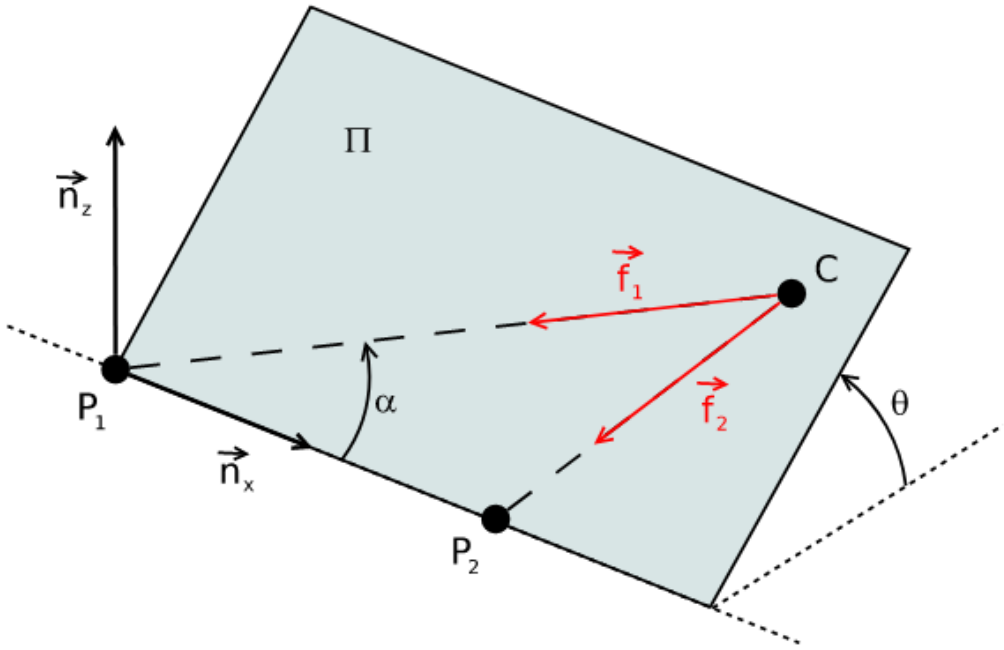

Figure 5: Rotation of plane Π *around* $\vec{n}_x$ *by* θ*. (credit: Kneip, Laurent, Davide Scaramuzza, and Roland Siegwart. "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation." CVPR 2011. IEEE, 2011.)*

The rotation of plane Π around $\vec{n}_x$ (Fig. 5) can be expressed as the following rotation matrix (just an elementary rotation matrix around the x-axis):

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

Note how this matrix's columns are the expressions of the new axes (η frame's axes) in the old frame (plane Π's frame). So we have the coordinate of C in η as:

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

Now that we have C^{η}, we only need the rotation matrix describing transformation from η to τ. It turns out that it's easier for us to get the rotation from τ to η first, then take the transpose. For that, we need to find out the expressions for $\vec{t}_x$, $\vec{t}_y$ and $\vec{t}_z$ in τ. It follows that we have

$$R_\eta^\tau = \left[R_\theta \cdot \left(\vec{t}_x^{\,\Pi} | \vec{t}_y^{\,\Pi} | \vec{t}_z^{\,\Pi}\right)\right] = \begin{pmatrix} -\cos\alpha & -\sin\alpha\cos\theta & \sin\alpha \\ -\sin\alpha\sin\theta & -\cos\alpha\cos\theta & -\sin\theta \\ 0 & -\cos\alpha\sin\theta & \cos\theta \end{pmatrix}$$

And therefore we have

$$Q(\alpha,\theta) = R_\tau^\eta = \begin{pmatrix} -\cos\alpha & -\sin\alpha\cos\theta & -\sin\alpha\sin\theta \\ \sin\alpha & -\cos\alpha\cos\theta & -\cos\alpha\sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix}$$

With $Q(\alpha, \theta)$ defined, we now need to write down the systems of equations to solve for α and θ. To do that, we simply write out the equations for transforming P_3^η into P_3^τ. Let $P_3^\eta = (p_1, p_2, 0)^T$ (which is known), it follows that

$$\begin{aligned} P_3^\tau &= Q(\alpha, \theta) \cdot \left(P_3^\eta - C^\eta(\alpha, \theta)\right) \\ &= \begin{pmatrix} -\cos\alpha \cdot p_1 - \sin\alpha\cos\theta \cdot p_2 + d_{12}(\sin\alpha \cdot b + \cos\alpha) \\ \sin\alpha \cdot p_1 - \cos\alpha\cos\theta \cdot p_2 \\ -\sin\theta \cdot p_2 \end{pmatrix} \end{aligned}$$

In addition, notice that the direction of P_3^τ is the same as that of $\vec{f}_3^\tau$ (also known). Define

$$\phi_1 = \frac{f_{3,x}^\tau}{f_{3,z}^\tau}, \qquad \phi_2 = \frac{f_{3,y}^\tau}{f_{3,z}^\tau}$$

It follows that

$$\begin{aligned} &\begin{cases} \phi_1 = \frac{P_{3,x}^\tau}{P_3^\tau z} \\ \phi_2 = \frac{P_{3,y}^\tau}{P_{3,z}^\tau} \end{cases} \\ \Rightarrow &\begin{cases} \phi_1 = \frac{-\cos\alpha\cdot p_1 - \sin\alpha\cos\theta\cdot p_2 + d_{12}(\sin\alpha\cdot b + \cos\alpha)}{-\sin\theta\cdot p_2} \\ \phi_2 = \frac{\sin\alpha\cdot p_1 - \cos\alpha\cos\theta\cdot p_2}{-\sin\theta\cdot p_2} \end{cases} \\ \Rightarrow &\begin{cases} \frac{\sin\theta}{\sin\alpha} p_2 = \frac{-\cot\alpha\cdot p_1 - \cos\theta\cdot p_2 + d_{12}(b + \cot\alpha)}{-\phi_1} \\ \frac{\sin\theta}{\sin\alpha} p_2 = \frac{p_1 - \cot\alpha\cos\theta\cdot p_2}{-\phi_2} \end{cases} \\ \Rightarrow &\cot\alpha = \frac{\frac{\phi_1}{\phi_2} p_1 + \cos\theta \cdot p_2 - d_{12} \cdot b}{\frac{\phi_1}{\phi_2}\cos\theta \cdot p_2 - p_1 + d_{12}} \end{aligned} \tag{11}$$

In addition, from definition of ϕ_2, we also have

$$\begin{aligned} \sin^2\theta \cdot f_2^2 p_2^2 &= \sin^2\alpha (p_1 - \cot\alpha\cos\theta \cdot p_2)^2 \\ \left(1 - \cos^2\theta\right)\left(1 + \cot^2\alpha\right) f_2^2 p_2^2 &= p_1^2 - 2\cot\alpha\cos\theta \cdot p_1 p_2 + \cot^2\alpha\cos^2\theta \cdot p_2^2 \end{aligned} \tag{12}$$

Substitute Eq. 11 into Eq. 12, we have

$$a_4 \cdot \cos^4\theta + a_3 \cdot \cos^3\theta + a_2 \cdot \cos^2\theta + a_1 \cdot \cos\theta + a_0 = 0 \tag{13}$$

where

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

We can solve Eq. 13 in closed form for $\cos\theta$, and using Eq. 11 to recover $\cot\alpha$. With α, and θ, we can then calculate C and Q to recover the camera center and orientation with respect to the world frame:

$$C = P_1 + N^{\mathcal{U}} \cdot C^{\eta}$$
$$R = N^T \cdot Q^T \cdot T$$

Implementation

You can find the implementations of the above algorithms in recipnps (except Gao's), a Rust library that I wrote for solving PnP problems. I use the nalgebra library extensively for linear algebra operations.

References

See the following papers:

- B. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle, "Review and analysis of solutions of the three point perspective pose estimation problem," Int J Comput Vision, vol. 13, no. 3, pp. 331–356, Dec. 1994, doi: 10.1007/BF02028352.
- L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in CVPR 2011, Jun. 2011, pp. 2969–2976. doi: 10.1109/CVPR.2011.5995464.
- X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 8, pp. 930–943, Aug. 2003, doi: 10.1109/TPAMI.2003.1217599.

For Kneip's P3P algorithm, I also referenced his excellent OpenGV library.