PnP: Perspective-n-Point Overview

Tuan-Linh TU

The lastest version: February 1, 2021

Given n 3D reference points whose 3D coordinates are known in the world coordinate: $\mathbf{p}_i = \big(x_i, y_i, z_i\big)^T \big| i = 1, 2, ..., n$ and whose 2D image corresponding are also known: $\mathbf{u}_i = \big(u_i, v_i\big)^T \big| i = 1, 2, ..., n$, the **PnP Problem** aims to retrieve the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$, accounting for camera orientation and position, respectively by the perspective constraint:

$$w_i \begin{pmatrix} \mathbf{u}_i \\ 1 \end{pmatrix} = \mathbf{K}\big[\mathbf{R}\big|\mathbf{t}\big] \begin{pmatrix} \mathbf{p}_i^w \\ 1 \end{pmatrix} = \mathbf{K}\mathbf{p}_i^c \tag{1}$$

in which, w_i are scalar projective parameters and $\mathbf{K}$ be the camera internal calibration matrix.

Let the n reference points whose 3D coordinates are known in the world coordinate system be: $\mathbf{p}_i \big| i = 1, 2, ..., n$ and let the 4 control points used to express their world coordinates be: $\mathbf{c}_j \big| j = 1, 2, 3, 4$. Let's express superscript ^{w} stand for point coordinates in the world coordinate system and superscript ^{c} stand for points in camera coordinate system, so, each reference point can be depicted as a weighted sum of control points:

$$\mathbf{p}_i^w = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^w \text{ with } \sum_{j=1}^{4} \alpha_{ij} = 1 \tag{2}$$

in which, α_{ij} are homogeneous barycentric coordinates. Similarly, the same relation holds in the camera coordinate system and can be writen:

$$\mathbf{p}_i^c = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c \text{ with } \sum_{j=1}^{4} \alpha_{ij} = 1 \tag{3}$$

0.1 *Solution as Weighted Sum of Eigenvectors*

Combining Equation 1 and Equation 3, perspective constraint equation can be writen as below:

$$w_i \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} = \begin{pmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{pmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{pmatrix} x_j^c \\ y_j^c \\ z_j^c \end{pmatrix} \tag{4}$$

in which, f_u, f_v are the focal length and u_c, u_v are the principle points of camera.

The last row of Equation 4 implies that: $w_i = \sum_{j=1}^{4} \alpha_{ij} z_j^c$. Subtituting this expression in the first two rows yields two linear equations for each reference point:

$$\sum_{j=1}^{4} \Big(\alpha_{ij} f_u x_j^c + \alpha_{ij} \big(u_c - u_i\big) z_j^c \Big) = 0 \tag{5}$$

$$\sum_{j=1}^{4}\left(\alpha_{ij}f_v y_j^c + \alpha_{ij}\big(v_c - v_i\big)z_j^c\right) = 0 \tag{6}$$

Equation 5 and Equation 6 can be combined and rewriten under matrix form for each reference point as below:

$$\begin{pmatrix}\alpha_{i1} & \alpha_{i2} & \alpha_{i3} & \alpha_{i4}\end{pmatrix} \otimes \begin{pmatrix} f_u & 0 & u_c - u_i \\ 0 & f_v & v_c - v_i \end{pmatrix} \mathbf{x} = 0 \tag{7}$$

in which $\mathbf{x} = \big[x_1^c, y_1^c, z_1^c, x_2^c, y_2^c, z_2^c, x_3^c, y_3^c, z_3^c, x_4^c, y_4^c, z_4^c\big]^T$ is unique unknown of Equation 7, $\otimes$ is Kronecker product of two vectors.

Finally, concatenating Equation 7 for all n reference points can be expressed as a linear system: $\mathbf{Mx} = 0$ where $\mathbf{M}$ is a $2n \times 12$ matrix. The solution of linear equation: $\mathbf{Mx} = 0$ belongs to kernel space (or null space) of $\mathbf{M}$ and can be expressed as:

$$\mathbf{x} = \sum_{i=1}^{N} \beta_i \mathbf{v}_i \tag{8}$$

in which, $\mathbf{v}_i$ are the columns of right-singular vectors of $\mathbf{M}$ corresponding to the N null-singular values of $\mathbf{M}$ with $N = \text{rank}\big(\text{Ker}\big(\mathbf{M}\big)\big)$.

By definition, if $\mathbf{v}_i$ is the i^{th} column of right-singular vectors gained by applying SVD on $\mathbf{M}$, so: $\mathbf{M} = \mathbf{U}\Sigma\mathbf{V}^T$. We also have: $\mathbf{M}^T\mathbf{M} = \mathbf{V}\Sigma^T\mathbf{U}^T\mathbf{U}\Sigma\mathbf{V}^T = \mathbf{V}\Sigma^T\Sigma\mathbf{V}^T$, means that $\mathbf{v}_i$ is the i^{th} null eigenvector of matrix $\mathbf{M}^T\mathbf{M}$. Solving for the matrix $\mathbf{M}$ is not as efficient as solving for matrix $\mathbf{M}^T\mathbf{M}$ because the number of equations is reduced in the second choice.

In theory, N equals to 1 with at least 6 reference points imaged by a perspective camera. If the camera becomes orthographic, the value of N increase to 4. Without loss of generality, we consider that the effective dimension N of null space of $\mathbf{M}^T\mathbf{M}$ equals to 4 and take some tests to estimate $\big\{\beta_i \big| i = 1, 2, 3, 4\big\}$.

0.2 *Choosing the Right Linear Combination*

The solution of Equation 7 can be expressed as a linear combination of the null eigenvectors of $\mathbf{M}^T\mathbf{M}$ as depicted in Equation 8 and finding this solution amounts to calculating the appropriate values of $\big\{\beta_i \big| i = 1, 2, ..., N\big\}$ coefficients of Equation 8.

In order to solve Equation 8, the constraints that the distances between control points as retrieved in the camera coordinate system should be equal to the control points as computed in the world coordinate system is considered, means that:

$$\| \mathbf{x}_i - \mathbf{x}_j \|^2 = \| \mathbf{c}_i^w - \mathbf{c}_j^w \|^2 \tag{9}$$

Subtituting Equation 8 to Equation 9 with two reference points i and j, we have:

$$\begin{aligned} &\| \big(\beta_1\mathbf{v}_1^i + \beta_2\mathbf{v}_2^i + \beta_3\mathbf{v}_3^i + \beta_4\mathbf{v}_4^i\big) - \big(\beta_1\mathbf{v}_1^j + \beta_2\mathbf{v}_2^j + \beta_3\mathbf{v}_3^j + \beta_4\mathbf{v}_4^j\big) \|^2 = \| \mathbf{c}_i^w - \mathbf{c}_j^w \|^2 \\ \Leftrightarrow &\| \beta_1\big(\mathbf{v}_1^i - \mathbf{v}_1^j\big) + \beta_2\big(\mathbf{v}_2^i - \mathbf{v}_2^j\big) + \beta_3\big(\mathbf{v}_3^i - \mathbf{v}_3^j\big) + \beta_4\big(\mathbf{v}_4^i - \mathbf{v}_4^j\big) \|^2 = \| \mathbf{c}_i^w - \mathbf{c}_j^w \|^2 \end{aligned} \tag{10}$$

Applying equality: $\big(a+b+c+d\big)^2 = a^2+2ab+b^2+2ac+2bc+c^2+2ad+2bd+2cd+d^2$ into Equation 10, we have:

$$\begin{aligned} \| \mathbf{c}_i^w - \mathbf{c}_j^w \|^2 = &\big(\beta_1\big)^2\big(\mathbf{v}_1^i - \mathbf{v}_1^j\big)^T\big(\mathbf{v}_1^i - \mathbf{v}_1^j\big) \\ &+ 2\beta_1\beta_2\big(\mathbf{v}_1^i - \mathbf{v}_1^j\big)^T\big(\mathbf{v}_2^i - \mathbf{v}_2^j\big) + \big(\beta_2\big)^2\big(\mathbf{v}_2^i - \mathbf{v}_2^j\big)^T\big(\mathbf{v}_2^i - \mathbf{v}_2^j\big) + 2\beta_1\beta_3\big(\mathbf{v}_1^i - \mathbf{v}_1^j\big)^T\big(\mathbf{v}_3^i - \mathbf{v}_3^j\big) \\ &+ 2\beta_2\beta_3\big(\mathbf{v}_2^i - \mathbf{v}_2^j\big)^T\big(\mathbf{v}_3^i - \mathbf{v}_3^j\big) + \big(\beta_3\big)^2\big(\mathbf{v}_3^i - \mathbf{v}_3^j\big)^T\big(\mathbf{v}_3^i - \mathbf{v}_3^j\big) + 2\beta_1\beta_4\big(\mathbf{v}_1^i - \mathbf{v}_1^j\big)^T\big(\mathbf{v}_4^i - \mathbf{v}_4^j\big) \\ &+ 2\beta_2\beta_4\big(\mathbf{v}_2^i - \mathbf{v}_2^j\big)^T\big(\mathbf{v}_4^i - \mathbf{v}_4^j\big) + 2\beta_3\beta_4\big(\mathbf{v}_3^i - \mathbf{v}_3^j\big)^T\big(\mathbf{v}_4^i - \mathbf{v}_4^j\big) + \big(\beta_4\big)^2\big(\mathbf{v}_4^i - \mathbf{v}_4^j\big)^T\big(\mathbf{v}_4^i - \mathbf{v}_4^j\big) \end{aligned}$$

Let's define $\boldsymbol{\beta} = \big[\beta_{11}, \beta_{12}, \beta_{22}, \beta_{13}, \beta_{23}, \beta_{33}, \beta_{14}, \beta_{24}, \beta_{34}, \beta_{44}\big]^T$ where $\beta_{ab} = \beta_a \beta_b$ with $\big(a, b\big) = \{1, 2, 3, 4\}$, so 4 control points will produce a linear system with 6 equations in the β_{ab} that can be writen as below:

$$\mathbf{L}\boldsymbol{\beta} = \boldsymbol{\rho} \tag{11}$$

in which, $\mathbf{L}$ is a 6×10 matrix formed with elements of $\big(\mathbf{v}^i - \mathbf{v}^j\big)^T \big(\mathbf{v}^i - \mathbf{v}^j\big)$, $\boldsymbol{\rho}$ is a 6-vector with the squared distances $\parallel \mathbf{c}_i^w - \mathbf{c}_j^w \parallel^2$.

0.3 *Efficient Gauss-Newton Optimization*

This step is designed to increase the accuracy of camera pose at very little extra computation cost by choosing the values $\boldsymbol{\beta} = \big[\beta_1, \beta_2, \beta_3, \beta_4\big]$ of Equation 8 that minimize the gap in distance between control points. It means refining $\boldsymbol{\beta}$ by finding out the values that minimize the error:

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \parallel \mathbf{L}\boldsymbol{\beta} - \boldsymbol{\rho} \parallel^2 \doteq \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \big(\mathbf{L}\boldsymbol{\beta} - \boldsymbol{\rho}\big)^T \big(\mathbf{L}\boldsymbol{\beta} - \boldsymbol{\rho}\big) \tag{12}$$

in which $\mathbf{r} = \mathbf{L}\boldsymbol{\beta} - \boldsymbol{\rho}$ is the sum of distances between control points.

Define $\mathbf{J} = \big[\frac{\partial \mathbf{r}}{\partial \beta_1}, \frac{\partial \mathbf{r}}{\partial \beta_2}, \frac{\partial \mathbf{r}}{\partial \beta_3}, \frac{\partial \mathbf{r}}{\partial \beta_4}\big]$ is Jacobian matrix of $\mathbf{r}$, with some initial value $\boldsymbol{\beta}$ that gained from the previous step, the Gauss-Newton algorithm is utilized to iteratively update $\boldsymbol{\beta}$ to find out the optimal ones as formula below:

$$\boldsymbol{\beta}_{i+1} = \boldsymbol{\beta}_i + \big(\mathbf{J}^T\mathbf{J}\big)^{-1}\mathbf{J}^T\mathbf{r} \tag{13}$$

in which, $\big(\mathbf{J}^T\mathbf{J}\big)^{-1}\mathbf{J}^T$ is *pseudo-inverse* matrix of $\mathbf{J}$. In general, $\big(\mathbf{J}^T\mathbf{J}\big)^{-1}$ can be computed by several divergent ways, and one of them is to utilize **QR decomposition**. With **QR decomposition**, a $n \times n$ matrix $\mathbf{A}$ can be extracted by multiplying a $n \times n$ orthogonal matrix (denoted by $\mathbf{Q}$) with an $n \times n$ upper-triangular matrix (denoted by $\mathbf{R}$), indeed, $\mathbf{A} = \mathbf{QR}$. In other way, because $\mathbf{Q}$ is an orthogonal matrix, so $\mathbf{Q}^{-1} = \mathbf{Q}^T$. Hence, we have: $\mathbf{A}^{-1} = \big(\mathbf{QR}\big)^{-1} = \mathbf{R}^{-1}\mathbf{Q}^{-1} = \mathbf{R}^{-1}\mathbf{Q}^T$.

0.4 *Computing Rotation Matrix and Translation Vector*

According to Equation 3, camera's coordinates $\mathbf{p}^c$ can be reached utilizing values $\boldsymbol{\beta}$ gained from previous steps, and from set of points in world coordinate system $\mathbf{p}^w$ and set of points in camera coordinate system $\mathbf{p}^c$, rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ can be closely computed by using *Iterative Closest Point* algorithm with the purpose for minimizing error function:

$$\min_{\mathbf{R},\mathbf{t}} \parallel \mathbf{p}^c - \big(\mathbf{R}\mathbf{p}^w + \mathbf{t}\big) \parallel^2 \tag{14}$$

subject to: $\mathbf{R} \in SO\big(3\big)$ and $\det\big(\mathbf{R}\mathbf{R}^{\mathrm{T}}\big) = 1$.

Let's define $\overline{\mathbf{p}}^w = \frac{1}{N}\sum_{i=1}^N \mathbf{p}_i^w$ and $\overline{\mathbf{p}}^c = \frac{1}{N}\sum_{i=1}^N \mathbf{p}_i^c$ are the central points of set of point in world coordinate system and camera coordinate system, respectively, and then, align all points in two sets using corresponding central points, we get two new sets of point as:

$$\begin{cases} \hat{\mathbf{p}}_i^w = \mathbf{p}_i^w - \overline{\mathbf{p}}^w \\ \hat{\mathbf{p}}_i^c = \mathbf{p}_i^c - \overline{\mathbf{p}}^c \end{cases} \tag{15}$$

From these sets, let's define correlation matrix $\mathbf{R} = \sum_{i=1}^{N} \hat{\mathbf{p}}_i^w \left(\hat{\mathbf{p}}_j^c\right)^T$ and perform SVD on $\mathbf{R}$, we get: $\mathbf{R} = \mathbf{U}\Sigma\mathbf{V}^T$. Finally, rotation matrix $\hat{\mathbf{R}}$ and translation vector $\mathbf{t}$ are extracted as below:

$$\begin{cases} \hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^T \\ \mathbf{t} = \overline{\mathbf{p}}^c - \hat{\mathbf{R}}\overline{\mathbf{p}}^w \end{cases} \tag{16}$$

in which $\hat{\mathbf{R}}$ is the nearest orthogonal matrix of $\mathbf{R}$.