Pinhole Camera Coordinate Transform Solver

Owen Lu
Electroimpact Inc.
owenl@electroimpact.com

***Abstract*— In order to understand the location of 4 cameras in relation to the tool-point coordinate system an algorithm was implemented using the method proposed by Zhengyou Zhang in his paper "A Flexible New Technique for Camera Calibration". It uses at least 3 images of a checkerboard pattern to locate the camera and define the intrinsic/extrinsic parameters of a pinhole camera model. A set of transforms then can be combined with tool-point transforms to get the relative transform from the tool-point to the camera. Refinements for lens distortion are assumed to be taken care of in the Keyence system so no non-linear solvers are used. This allows the images to appear as if they were taken from a pinhole camera.**

I. Introduction

The main goal of this calibration routine is not to find the intrinsic parameters of the camera, but instead to find its location in relation to the tool-point coordinates. This allows us to know the position of the camera as we move it through space, and project the images it takes onto a 3D surface model virtually.

II. Conventions and conversion

Dependent on whether the data is stored in rows or columns the transform has a slightly different form. We assume that a vector is a column vector, and thus, the rotation matrix appears to the left of the vector when applying the transform, shown in Eq, 1.

$$x' = \mathbf{R}x + t \tag{1}$$

$$x = [x_1 \quad x_2 \quad x_3]^T \tag{2}$$

$$t = [t_1 \quad t_2 \quad t_3]^T \tag{3}$$

The reason that this convention is being used is due to explicit solutions given to solve camera intrinsics and extrinsic in the paper using the same convention.

Augmentation of coordinate systems is also common, which allows the transform in Eq. 1 to be applied as one matrix multiplication.

Let x be the augmented vector. It follows that the transform can be written as a matrix multiplication as shown in Eq. (5).

$$x = [x_1 \quad x_2 \quad x_3 \quad 1]^T \tag{4}$$

$$x' = [\mathbf{R} \quad t]x \tag{5}$$

A key point is to notice that although x is augmented, after multiplication of the transform matrix, x' is not augmented. Another convention instead of using $[\mathbf{R} \quad t]$ as the transform matrix is to expand it even further from a 3×4 matrix into a 4×4 matrix to keep the augmentation. This allows successive transforms to be applied very easily. An example of such a matrix is shown in Eq. 6.

$$\mathbf{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Therefore, it follows that the augmented transformed vector x' can be written as the product of $\mathbf{T}$ and x.

$$x' = \mathbf{T}x \tag{7}$$

It is also common for the vectors to be written as row vectors instead of column vectors. Given that we have computed $\mathbf{T}$ already, to convert to row vectors we simply need the transpose of $\mathbf{T}$.

$$x'^T = x^T\mathbf{T}^T \tag{8}$$

This allows us to easily convert conventions depending on the implementations in different libraries. Notably, Matrix3D transforms in the standard C# Media3D assembly uses the transform $\mathbf{T}^T$ convention, which is why the transform must be converted if it is first obtained using the column vector convention.

III. Algorithm Summary

Within the C# implementation, it is assumed that the coordinates within a checkerboard frame are known and that the corners of the checkerboard within an image can be detected with the Keyence system. That is, we assume that we have one set of points that have the locations of the corners in a coordinate system, and n sets of points that correspond to the locations of the corners within each image.

The algorithm is then broken down into 4 major steps

1. Initial homography estimation
2. Solve for camera intrinsic matrix
3. Solve for a set of camera extrinsic transforms
4. Combine camera extrinsic transforms with tool-point transforms to solve tool-point to camera transform

These steps are described in detail in the following sections.

IV. Homography Estimation

Due to the checkerboard pattern being contained within a plane, we can define a coordinate system that makes all checkerboard corners have Z coordinate equal to zero. Originally the equation that maps 3D points onto the 2D image is in Eq. 9.

$$s\begin{bmatrix}u\\v\\1\end{bmatrix} = \mathbf{A}[\mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{r_3} \quad \mathbf{t}]\begin{bmatrix}x\\y\\z\\1\end{bmatrix} \tag{9}$$

However, since $Z = 0$, column $\boldsymbol{r_3}$ of the rotation matrix can be removed for the analysis. Therefore, we can relate the vector on the left with an augmented 2D vector $[X \quad Y \quad 1]^T$ on the right via homography $\mathbf{H}$.

$$\begin{bmatrix}u\\v\\1\end{bmatrix} = \mathbf{H}\begin{bmatrix}x\\y\\1\end{bmatrix} \tag{10}$$

$$\mathbf{H} = \mathbf{A}[\mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{t}] \tag{11}$$

Writing $\mathbf{H}$ in terms of its individual entries gives Eq. 12.

$$\mathbf{H} = \begin{bmatrix}H_{11} & H_{12} & H_{13}\\H_{21} & H_{22} & H_{23}\\H_{31} & H_{32} & H_{33}\end{bmatrix} \tag{12}$$

From Eq. 10 and Eq. 12, we can rewrite the equations for pixel coordinates u and v.

$$u = \frac{xH_{11} + yH_{12} + H_{13}}{xH_{31} + yH_{32} + H_{33}} \tag{13}$$

$$y = \frac{xH_{21} + yH_{22} + H_{23}}{xH_{31} + yH_{32} + H_{33}} \tag{14}$$

Let $\mathbf{h}$ be the listed elements of $\mathbf{H}$ in row major order.

$$\mathbf{h} = [H_{12}, H_{12}, H_{13}, H_{21}, H_{22}, H_{23}, H_{31}, H_{32}, H_{33}]^T \tag{15}$$

Then Eq. 13 and Eq. 14 can be written as two homogeneous equations using $\mathbf{h}$.

Let

$$\boldsymbol{g_x} = [-x, -y, -1, 0, 0, 0, ux, uy, u] \tag{16}$$

$$\boldsymbol{g_y} = [0, 0, 0, -x, -y, -1, vx, vy, v] \tag{17}$$

$$\boldsymbol{g_x}\mathbf{h} = 0 \tag{18}$$

$$\boldsymbol{g_y}\mathbf{h} = 0 \tag{19}$$

Therefore, for each pair of points (u, v) and (x, y) we get 2 equations. Since each point gives two equations, we can construct the matrix $\mathbf{G}$ which concatenates all of the equations by stacking them vertically.

Matrix $\mathbf{G}$ must then be a $2n \times 9$ matrix where n is the number of coordinate pairs. This formulates a homogenous system of equations.

$$\mathbf{Gh} = \mathbf{0} \tag{20}$$

At this point, Singular Value Decomposition (SVD) is used to solve for the least squares solution to $\mathbf{h}$ which is returned as the right hand singular vector associated with the smallest singular value σ_9.

V. Intrinsic Estimation

The basic principle behind intrinsic estimation is that each image taken will result in 2 constraints on the intrinsic parameters of the camera. Taking a sufficient number of pictures allows the intrinsic matrix to be solved. The derivation is not shown here, but the method to construct constraint matrix $\mathbf{V}$ to solve transformed variables $\mathbf{b}$, which can be used to calculate intrinsic matrix $\mathbf{A}$, is given.

Let v_{ij} be calculated as below from estimated homography $\mathbf{H}$.

$$v_{ij} = \begin{bmatrix}H_{1i}H_{1j}\\H_{1i}H_{2j} + H_{2i}H_{1j}\\H_{2i}H_{2j}\\H_{3i}H_{1j} + H_{1i}H_{3j}\\H_{3i}H_{2j} + H_{2i}H_{3j}\\H_{3i}H_{3j}\end{bmatrix}^T \tag{21}$$

Then $\mathbf{V}$ can be formulated by stacking 2 equations for each of the m images taken. Therefore, $\mathbf{V}$ is a $2m \times 6$ matrix.

$$\begin{bmatrix}v_{12}\\v_{11} - v_{12}\end{bmatrix}\mathbf{b} = \mathbf{0} \tag{22}$$

We need at least 3 images in order to have a unique solution to $\mathbf{b}$ up to scale.

$$\mathbf{b} = [b_1, b_2, b_3, b_4, b_5, b_6]^T \tag{23}$$

The intrinsic matrix $\mathbf{A}$ is a 3×3 that maps 3D points onto the 2D image pixels.

$$\mathbf{A} = \begin{bmatrix}\alpha & \gamma & u_0\\0 & \beta & v_0\\0 & 0 & 1\end{bmatrix} \tag{24}$$

Each parameter in $\mathbf{A}$ can then be calculated from $\mathbf{b}$. From Zhang's paper w,e have the following formulae for the parameters which should be calculated in the same sequence.

$$v_0 = \frac{b_2 b_4 - b_1 b_5}{b_1 b_3 - b_2^2} \tag{25}$$

$$\phi = b_6 - \frac{\left(b_4^2 - v_0 \;\; v_2 b_4 - b_1 b_5 \;\right)}{b_1} \tag{26}$$

$$\alpha = \sqrt{\frac{\phi}{b_1}} \tag{27}$$

$$\beta = \sqrt{\frac{\phi b_1}{b_1 b_3 - b_2^2}} \tag{28}$$

$$\gamma = -\frac{b_2 \alpha^2 \beta}{\phi} \tag{29}$$

$$u_0 = \frac{\gamma u_0}{\alpha} - \frac{b_4 \alpha^2}{\phi} \tag{30}$$

At this point matrix, $\mathbf{A}$ can be solved via substitution.

VI. Extrinsic Estimation

Now that $\mathbf{A}$ is known it is a straightforward calculation to find the rotation and translation matrices defined in Eq. 9.

Let

$$\mathbf{R} = [\mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{r_3}] \tag{31}$$

$$\mathbf{H} = [\mathbf{h_1} \quad \mathbf{h_2} \quad \mathbf{h_3}] \tag{32}$$

Let $\mathbf{C} = \frac{\mathbf{A}^{-1}}{\|\mathbf{A}^{-1}\mathbf{h_1}\|} = \frac{\mathbf{A}^{-1}}{\|\mathbf{A}^{-1}\mathbf{h_2}\|}$

$$\mathbf{r_1} = \mathbf{C}\mathbf{h_1} \tag{33}$$

$$\mathbf{r_2} = \mathbf{C}\mathbf{h_2} \tag{34}$$

$$\mathbf{r_3} = \mathbf{r_1} \times \mathbf{r_2} \tag{35}$$

$$\mathbf{t} = \mathbf{C}\mathbf{h_3} \tag{36}$$

At this point, we have all the parameters to construct transformation matrix $\mathbf{T}$ in Eq. 6.

VII. Combining tool point transforms

The purpose of performing this calibration method was to find the transform from the tool-point to the camera. Up to this point, the algorithm has found the rotation and translation of the camera relative to the checkerboard coordinates. In our application, the checkerboard is located in space and thus, the transform to convert a point from the checkerboard coordinates to the tool-point and vice-versa is assumed to be known.

Thus, the process is to find the aggregate transform from tool-point to checkerboard to camera m times and average the transform.

Suppose we have two transforms defined by $\mathbf{R_1}, \mathbf{t_1}$ and $\mathbf{R_2}, \mathbf{t_2}$ applied in sequence.

Then the aggregate transform has the following properties

$$\mathbf{T_a} = [\mathbf{R_2 R_1} \quad \mathbf{R_2 t_1} + \mathbf{t_2}] \tag{37}$$

Then all we must do is compute the aggregate rotation matrix and translation as in Eq. 37 to find the aggregate transform from tool-point to the camera.

VIII. Averaging multiple transforms

The concept of averaging, in this case, means the following. Suppose p is a point that we wish to transform by $\mathbf{T}_{\boldsymbol{a}}$. Since there are m estimates of $\mathbf{T}_{\boldsymbol{a}}$ we need to average them.

$$p_i' = \mathbf{T}_{\boldsymbol{a}_i} p \tag{38}$$

An easy method would simply be to add all the estimations of p_i and average them.

$$p'_{avg} = \frac{1}{m}\sum_{i=1}^{m} p_i' = \frac{1}{m}\left(\sum_{i=1}^{m} \overline{\mathbf{T}_{\boldsymbol{a}_i}}'\right) p \tag{39}$$

This means that the averaged transformed can be thought of as adding all the transforms element by element and averaging them.

IX. Rotation Matrix Correction

Due to estimation error, the rotation matrix component found by averaging in Eq. 39 will not, in general, satisfy the rotation matrix. We then find the matrix that satisfies both the properties of the rotation matrix and minimizes the Frobenius norm σ.

$$\sigma = \left\|\mathbf{R}_{\mathrm{avg}} - \mathbf{R}_{\mathrm{c}}\right\| \tag{40}$$

This can be done also with the SVD. Suppose we use SVD to obtain matrices $\mathrm{U\Sigma V^T}$, then the expression for the best fit rotation matrix can be calculated using $\mathbf{UV}$.

$$\mathbf{R}_{\mathrm{avg}} = \mathbf{U\Sigma V^T} \tag{41}$$

$$\mathbf{R}_{\mathrm{c}} = \mathbf{UV} \tag{42}$$

If the rotation matrix $\mathbf{R}_c$ is improper, then use a negative left singular vector $\mathbf{u}_3$ in order to computer $\mathbf{R}_c$.