CSCE 643 Multi-View Geometry CV Project II

***Abstract*—In this project, we explore and investigate different approaches for estimation in 2D projective transformation. In total, we implemented 3 approaches for finding the homography transformation between two different images, Direct Linear Transformation (DLT), normalized DLT and DLT with Sampson error minimization. This paper mainly focuses on the mathematical foundations for those methodologies used in our implementation and presents the results through combining two images into one panorama to validate the correctness of our approach. We can see clearly that with the advancement of our approach, the sampson error is gradually reduced (though there are few differences we can notice between those image panorama results).**

I. Calibrate Lens Distortion

A basic assumption in camera geometry study is that a linear model is an accurate model of the imaging process, i.e., world point, image point and optical center point are collinear and world lines are imaged as lines and so on. However, this might not be true for real lens in the world. Radial distortion is a typical deviation that real lens might have, we need to correct image measurements to those that should be obtained under a perfect linear camera action.

A. Problem Formulation

Suppose $(\tilde{x}, \tilde{y})$ is the ideal image position that obeys linear projection (under perfect lens settings), (x_d, y_d) to be the actual image position after the effects of radial distortion, $\tilde{r} = \sqrt{\tilde{x^2} + \tilde{y^2}}$ is the radial distance from the center of radial distortion, $L(\tilde{r})$ is a function of radius $\tilde{r}$ as the distortion factor.

For the coordinates of point under non-distorted pinhole projection $(\tilde{x}, \tilde{y})$ (in units of focal-length), we have:

$$(\tilde{x}, \tilde{y}, 1)^T = [I|\mathbf{0}]\mathbf{x}_{cam} \tag{1}$$

where $\mathbf{x}_{cam}$ is the 3D point in camera coordinates and it is related to world coordinates as follows:

$$\mathbf{x}_{cam} = \begin{bmatrix} R & -R\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{x} \tag{2}$$

Now we can easily model the radial distortion by introducing the function of $\tilde{r}$:

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = L(\tilde{r}) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \tag{3}$$

If we dig into a single point coordinate (x, y), we have:

$$\begin{aligned} \tilde{x} &= x_c + L(r)(x - x_c) \\ \tilde{y} &= y_c + L(r)(y - y_c) \end{aligned} \tag{4}$$

where (x_c, y_c) is center of the image, $(\tilde{x}, \tilde{y})$ is the coordinates after correcting radial distortion. Also be noticed that r can is the radius from center of the image to the measured coordinates, which can be solved by:

$$r^2 = (x - x_c)^2 + (y - y_c)^2 \tag{5}$$

B. Distortion function and image center

Now let's talk about the choice of function $L(r)$ and center of image (x_c, y_c). According to the textbook, $L(r)$ is only defined for positive r and $L(0) = 1$. We can use Taylor expansion as an approximation of the distortion function:

$$L(r) = 1 + k_1 r + k_2 r^2 + k_3 r^3 + \dots \tag{6}$$

For image center (x_c, y_c), we often use principal point though it might not reside in the exact same location. On choosing the distortion function for this project, we did a tradeoff consideration. Apparently if we have higher order Taylor expansion for distortion function, we can get better correcting results, however, as the camera we are actually using is really good (though it is a relatively old model, modern cameras are generally good enough and generates very little distortion), we decided to choose $L(r) = 1 + k_1 r + k_2 r^2$ as our distortion function. The intuition behind this is that typically we choose higher order of distortion function as the degree of distortion increases, in our case the distortion is even ignorable thus we use a typical 2-order function to deal with our case.

C. Distortion function minimization

Similar as the minimization process we have done in the previous projects, here we need to minimize the geometric distance between predicted coordinates (by using $L(r)$) and measured coordinates. This can be done by exploiting the colinearity of points on the same line. In our project, we use the checker board image captured by our camera for lens distortion correction. The basic idea is that we can easily identify corner points on checker board and they naturally form parallel and perpendicular lines. If we determine a straight line using two corner points on the line, then all the other corner points along the line should be on the line if without radial distortion. Let's consider a simple scenario, suppose we have $\mathbf{A}, \mathbf{B}, \mathbf{D}$ on same line of a checker board, and the image center is $\mathbf{C}$, then we can do cross product to solve the crossing product of $\vec{AB}$ and $\vec{CD}$ to get $\mathbf{D}' = \vec{AB} \times \vec{CD}$, due to the radial distortion we have $\mathbf{D} \neq \mathbf{D}'$. If we apply $L(r)$ for the measured D to get corrected coordinates $\hat{\mathbf{D}}$, we want to

find a function that minimizes $d(\hat{\mathbf{D}}, \mathbf{D}')$. This is now reduced to a simple minimization problem of geometric distance like we did before, we can construct the corresponding error function based on discussions above and use non linear solver in MATLAB to get function $L(r)$ that does the required minimization.

II. Camera Calibration

Recall from projects before, we have implemented DLT approach for finding the transformation homography from one image to another. Now in the camera calibration process, we can use similar methods, let's first review the original DLT approach and normalization process.

A. DLT Foundation

In the simplest DLT approach, we first transform the homogeneous equation given in equation 7 as follows:

$$\mathbf{x}_i \times \mathbf{P}\mathbf{X}_i = \mathbf{0} \tag{7}$$

Where $\mathbf{P}$ is the camera matrix and can be decomposed into combination of vectors like:

$$\mathbf{P} = \begin{pmatrix} \mathbf{p_1} \\ \mathbf{p_2} \\ \mathbf{p_3} \end{pmatrix} \tag{8}$$

Then we can rewrite equation 7 into the following form:

$$\mathbf{P}\mathbf{X}_i = \begin{pmatrix} \mathbf{p}_1\mathbf{X}_i \\ \mathbf{p}_2\mathbf{X}_i \\ \mathbf{p}_3\mathbf{X}_i \end{pmatrix} \tag{9}$$

Suppose the coordinates of the set of points we have are:

$$\mathbf{x}_i = (x_i, y_i, z_i) \tag{10}$$

As we have $\mathbf{p}_i\mathbf{X}_i = \mathbf{X}_i^T\mathbf{p}_j^T$, we can further get the following simultaneous linear equations set:

$$\begin{bmatrix} \mathbf{0}^T & -z_i\mathbf{X}_i^T & y_i\mathbf{X}_i^T \\ z_i\mathbf{X}_i^T & \mathbf{0}^T & -x_i\mathbf{X}_i^T \\ -y_i\mathbf{X}_i^T & x_i\mathbf{X}_i^T & \mathbf{0}^T \end{bmatrix} \begin{pmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{pmatrix} = \mathbf{0} \tag{11}$$

Be noticed that different from the math we did before in last project, here $\mathbf{X}_i$ is a 4D coordinates and p_i also contains 4 entries. Since we can obtain the 3rd row of the left matrix above using by combining first two rows up to scale, we can safely prune equation 7 to only preserve the linearly independent first two rows in that matrix and obtain:

$$\mathbf{A}\mathbf{p} = \mathbf{0} \tag{12}$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}^T & -z_i\mathbf{X}_i^T & y_i\mathbf{X}_i^T \\ z_i\mathbf{X}_i^T & \mathbf{0}^T & -x_i\mathbf{X}_i^T \\ -y_i\mathbf{X}_i^T & x_i\mathbf{X}_i^T & \mathbf{0}^T \end{bmatrix}, \mathbf{p} = \begin{pmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{pmatrix} \tag{13}$$

As we know:

$$\mathbf{P} = \begin{pmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{pmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \tag{14}$$

Now we have come to the point that is very similar to what we have done before in Project 1. After dehomogenization, the equation 12 we get is actually a equation with 11 unknowns in $\mathbf{p}$ (11 DoFs) which we can solve exactly if we have $5\frac{1}{2}$ point pairs (every point pair provide 2 equations), to get the homography we can do as the followings:

- If we have exact $5\frac{1}{2}$ point correspondences (which means the we can only know one of the coordinate for sixth points) in both images, we can solve equation 12 to get an exact solution, but as there might be some noises in both images, it could lead to very bad results.
- We can also solve this equation is we can find more point correspondence to get so-called overdetermined solution through singular value decomposition (SVD), that way we should get more accurate results since we are provided with more information in the image.

B. Normalized DLT

1) 2D Image Frame Normalization: Similar to the DLT part we did in finding transformation homography between two images, it's sometimes very important to carry out coordinates normalization to provide better homography (camera matrix in current case). $\mathbf{x}_i$ is in the same dimension as before, thus we can normalize it using previous method, i.e., the points should be translated so that centroid of all points is at the origin and we also need to scale them so that their RMS (root-mean-squared) distance from origin is $\sqrt{2}$.

2) 3D World Frame Normalization: However, with the introduction of points in world frame, we now have to consider the 3D points $\mathbf{X}_i$. To simplify the problem, here we consider the case where the variation in point depth from the camera is relatively slight we can do similar normalization for 3D points as we did for 2D points. Therefore, we translate centroid of all measured points to the origin and scale their coordinates so that RMS distance from the origin is $\sqrt{3}$. As mentioned in the textbook, such 2D-alike approach is actually suitable for a compact distribution of points.

C. Camera Calibration Steps

According to the textbook, the camera calibration steps can be carried out as shown in Figure II-C.

Objective

Given $n \geq 6$ world to image point correspondences $\{\mathbf{X}_i \leftrightarrow \mathbf{x}_i\}$, determine the Maximum Likelihood estimate of the camera projection matrix $\mathtt{P}$, i.e. the $\mathtt{P}$ which minimizes $\sum_i d(\mathbf{x}_i, \mathtt{P}\mathbf{X}_i)^2$.

Algorithm

(i) **Linear solution.** Compute an initial estimate of $\mathtt{P}$ using a linear method such as algorithm 4.2(p109):

  (a) **Normalization:** Use a similarity transformation $\mathtt{T}$ to normalize the image points, and a second similarity transformation $\mathtt{U}$ to normalize the space points. Suppose the normalized image points are $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$, and the normalized space points are $\tilde{\mathbf{X}}_i = \mathtt{U}\mathbf{X}_i$.

  (b) **DLT:** Form the $2n \times 12$ matrix $\mathtt{A}$ by stacking the equations (7.2) generated by each correspondence $\tilde{\mathbf{X}}_i \leftrightarrow \tilde{\mathbf{x}}_i$. Write $\mathbf{p}$ for the vector containing the entries of the matrix $\tilde{\mathtt{P}}$. A solution of $\mathtt{A}\mathbf{p} = \mathbf{0}$, subject to $\|\mathbf{p}\| = 1$, is obtained from the unit singular vector of $\mathtt{A}$ corresponding to the smallest singular value.

(ii) **Minimize geometric error.** Using the linear estimate as a starting point minimize the geometric error (7.4):

$$\sum_i d(\tilde{\mathbf{x}}_i, \tilde{\mathtt{P}}\tilde{\mathbf{X}}_i)^2$$

over $\tilde{\mathtt{P}}$, using an iterative algorithm such as Levenberg–Marquardt.

(iii) **Denormalization.** The camera matrix for the original (unnormalized) coordinates is obtained from $\tilde{\mathtt{P}}$ as

$$\mathtt{P} = \mathtt{T}^{-1}\tilde{\mathtt{P}}\mathtt{U}.$$

Fig. 1. Algorithm of Camera Calibration using Gold Standard