Camera Calibration Report

Suyang Jiang sj2364

February 22, 2017

Abstract

In this assignment, I calibrate the iPhone7's camera which the focal length is 4mm so as to be able to capture images of objects from known locations and with a known camera model. I estimated the calibration matrix using least square method using MATLAB and then extract the intrinsic and extrinsic parameters from calibration matrix. Finally I reconstructed the image coordinates from the world coordinates using this estimate of the calibration matrix.

1 Introduction

I took a picture of the pattern which has six points on it in world space. Then use least squares method to estimate the calibration matrix and solve the homogeneous over-constrained equation system by performing singular value decomposition(SVD). Use the calibration matrix to calculate world to camera(extrinsic) and camera to pixel raster parameters(intrinsic). At last, I used the world coordinate and the calibration matrix to calculate the pixel locations, then compared the calculated pixel locations to the measured location and plot the difference.

2 Experiment Design

2.1 Data Capture

I used the two copies of the patterns and mount it into a straight corner of a wall as shown in Figure 1. I marked six points at different locations with red ink,and marked the origin and x,y,z axis. Real world coordinates (X,Y,Z) are measured via a ruler and stored as .xls type file. I took a picture about 45.5cm away from the world coordinate origin to the optical center. I use world coordinates similarly to the camera coordinates that point towards the camera. Corresponding pixel locations are obtained in the image by using ginput() function in MATLAB by mouses clicking the corresponding points in the image. After get world and image coordinates, add a new column to the right which all elements are one.

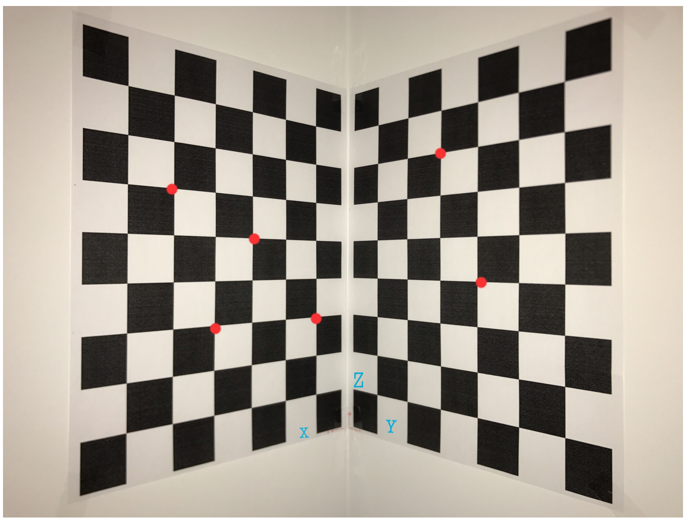


Figure 1: Checkerboard pattern on the wall corner

2.2 Estimation of Calibration Matrix

LaTeX In this part, I use least square method to calibrate the dataset and use the MATLAB functions to solve the homogeneous over-constrained equation system, then we can get the estimation of calibration matrix M. After decomposition of matrix M, we can get the extrinsic and intrinsic parameters. First, given n points $P_1, P_2, \ldots, P_n$ with known positions and their images $p_1, p_2, \ldots, p_n$, we can have:

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \dfrac{\boldsymbol{m}_1 \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3 \cdot \boldsymbol{P}_i} \\ \dfrac{\boldsymbol{m}_2 \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3 \cdot \boldsymbol{P}_i} \end{pmatrix} \Longleftrightarrow \begin{pmatrix} \boldsymbol{m}_1 - u_i \boldsymbol{m}_3 \\ \boldsymbol{m}_2 - v_i \boldsymbol{m}_3 \end{pmatrix} \boldsymbol{P}_i = 0$$

The constraints associated with the n points yield a system of 2n homogeneous linear equations in the 12 coefficients of the matrix M.

$$\mathcal{P}\boldsymbol{m} = 0 \text{ with } \mathcal{P} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{P}_1^T & \boldsymbol{0}^T & -u_1 \boldsymbol{P}_1^T \\ \boldsymbol{0}^T & \boldsymbol{P}_1^T & -v_1 \boldsymbol{P}_1^T \\ \ldots & \ldots & \ldots \\ \boldsymbol{P}_n^T & \boldsymbol{0}^T & -u_n \boldsymbol{P}_n^T \\ \boldsymbol{0}^T & \boldsymbol{P}_n^T & -v_n \boldsymbol{P}_n^T \end{pmatrix} \text{ and } \boldsymbol{m} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \\ \boldsymbol{m}_3 \end{pmatrix} = 0$$

When n $\geq$ 6, homogeneous linear least-square can be used to compute the value of the unit vector m (hence the matrix M) that minimizes $|Pm|^2$ as the solution of an eigenvalue problem. The solution is the eigenvector with least eigenvalue of $P^T P$. By using the coordinates we have, we can get the 12*12 matrix of P. The eigenvectors of matrix $P^T P$ can be computed by performing the singular value decomposition(SVD) of P, functions in MATLAB are:

```
%Perform SVD of P
[U S V] = svd(P);
[min_val, min_index] = min(diag(S(1:12,1:12)));

%m is given by right singular vector of min. singular value
m = V(1:12, min_index);
```

2.3 Calculation of Intrinsic and Extrinsic Parameters

After getting calibration matrix M, we can extract the parameters from this matrix. Using the fact that the rows of a rotation matrix have unit length and are perpendicular to each other yields:

$$\begin{cases} \rho = \varepsilon / |\boldsymbol{a}_3|, \\ \boldsymbol{r}_3 = \rho \boldsymbol{a}_3, \\ u_0 = \rho^2 (\boldsymbol{a}_1 \cdot \boldsymbol{a}_3), \\ v_0 = \rho^2 (\boldsymbol{a}_2 \cdot \boldsymbol{a}_3), \end{cases} \quad \text{where} \quad \varepsilon = \mp 1. \quad \begin{cases} \cos\theta = -\dfrac{(\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3)}{|\boldsymbol{a}_1 \times \boldsymbol{a}_3||\boldsymbol{a}_2 \times \boldsymbol{a}_3|}, \\ \alpha = \rho^2 |\boldsymbol{a}_1 \times \boldsymbol{a}_3| \sin\theta, \\ \beta = \rho^2 |\boldsymbol{a}_2 \times \boldsymbol{a}_3| \sin\theta, \end{cases}$$

$$\begin{cases} \boldsymbol{r}_1 = \dfrac{\rho^2 \sin\theta}{\beta} (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{1}{|\boldsymbol{a}_2 \times \boldsymbol{a}_3|} (\boldsymbol{a}_2 \times \boldsymbol{a}_3), \\ \boldsymbol{r}_2 = \boldsymbol{r}_3 \times \boldsymbol{r}_1. \end{cases}$$

Thus, we can follow the equations to calculate the extrinsic and intrinsic parameters.

3 Results

3.1 Calibration Matrix

The calibration matrix M is a 3*4 matrix, which is:

$$M = \begin{bmatrix} 0.0260 & -0.0068 & -0.0006 & -0.6358 \\ 0.0071 & 0.0068 & 0.0222 & -0.7710 \\ 0.0000 & 0.0000 & -0.0000 & -0.0003 \end{bmatrix}$$

3.2 Extrinsic Parameters

By using MATLAB functions we can calculate the parameters. Extrinsic parameters: rotation matrix and translation matrix are shown as follow:

$$R = \begin{bmatrix} -0.6905 & 0.7234 & -0.0028 \\ -0.0238 & -0.0266 & -0.9994 \\ -0.7230 & -0.6900 & 0.0356 \end{bmatrix} \quad t = \begin{bmatrix} -1.1019 \\ 15.3234 \\ 47.1742 \end{bmatrix}$$

3.3 Intrinsic Parameters

And the intrinsic parameters: pixel unit α, β, image center position u_0, v_0, the angle between the two image axes θ are shown as then following Table 1:

Intrinsic Parameters	Value
α	3453.2
β	3394.2
u_0	2123.9
v_0	1364.4
θ	89.5315

Table 1: Intrinsic Parameters.

4 Discussion of Results

4.1 Intrinsic Parameters

The sensor size of iPhone7 rear camera is shown in the Figure 2. Rear camera's focal length is 4 mm and effective pixels is 12 mega-pixels. The pixel size of iPhone7 rear camera is $1.22 * 10^{-}3$mm, so that the α and β equal to focal length divided by pixel size, which is 3278.69. The results of α and β are 3453.2 and 3394.2, which are reasonable with little difference from the real value.

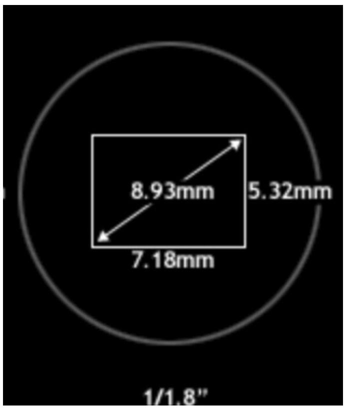


Figure 2: Sensor size of f/1.8 camera

I get θ equals to 89.5315, which means that the camera coordinate are not skewed much and the X and Y axes in the image frame are almost perpendicular to each other.

The image dimension is 2546×1910, which means the center of the image lies at (1273,955). I calculated the u_0 and v_0 equal to 2123.9 and 1364.4 respectively. The image center does not coincide with the principle point and is offset by (850.9,409.4), as the results in the experiments.

4.2 Extrinsic Parameters

The three rows of rotation matrix have norm equal to 1 and also it is a very small number of their dot product with each other . This is justified because of possible errors in the measurement process which is purely measured manually.

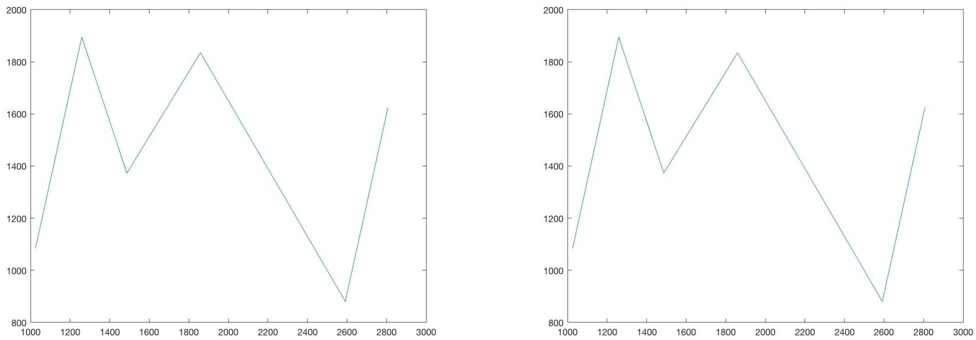
The transition vector also matches the real value nearly. The plain distance from the world origin to the camera was about 47.5 centimeters which is approximately equal to the norm of the translation vector, which is 47 centimeters.We can see that these points can give a very good estimate of the camera parameters.

4.3 Image Coordinates Reconstruction

After we have estimated the calibration matrix M, and measured the world coordinates, we can perspective projection equation(homogeneous) to calculate the image location, so that we can compare the calculated locations to the image coordinates to observe the difference. I plot the points in these two matrix as shown as follows. As we can see that the difference of these corresponding points is very small. The sum of the distance between the points on the image and the new image is 0.0332 which is justified. I We note that the reconstruction errors are not so big and can be justified given the amount of inherent errors in the measuring process itself. The results of this experiment are tabulated below:

World Coordinates	Image Coordinates	NewImage Coordinates	Distance
(15.3,0,18)	(1259,1895)	(1025.4,1084.6)	0.5844
(12.3,0,9)	(1259,1895)	(1258.9,1895.1)	0.1435
(9.3,0,15)	(1487,1373)	(1486.4,1373.6)	0.8450
(3.3,0,9)	(1859,1835)	(1859.3,1834.7)	0.4035
(0,9.3,21)	(2591,881)	(2591,881)	0.414
(0,12.3,12)	(2807,1625)	(2807,1625)	0.0332

Table 2: Reconstruction.



(a) measured locations (b) calculated pixel locations

Figure 3: Points plot comparison

5 Rerun a Second Image-bonus question

I marked another 6 points on the checkerboard pattern with green ink and take a picture at different place with the same camera which is shown as below. Using the same method, I estimated the intrinsic and extrinsic parameters.

Intrinsic Parameters	Value
α	3370.4
β	3298.1
u_0	1387.1
v_0	1007.9
θ	88.6103

Table 3: Intrinsic Parameters.

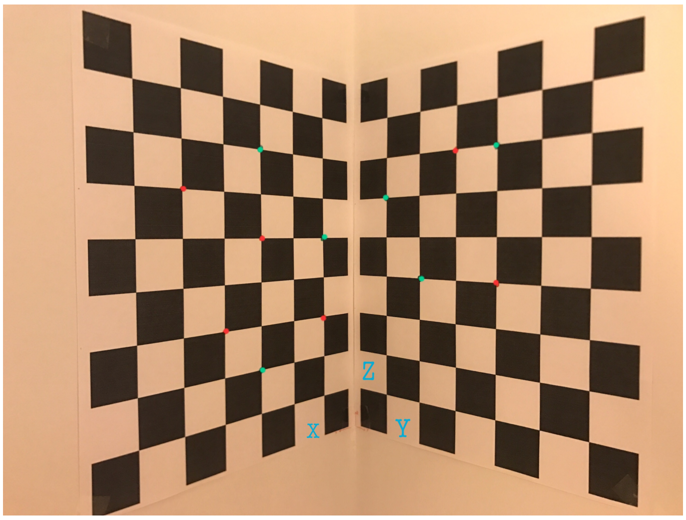


Figure 4: Checkerboard pattern on the wall corner with new 6 points

We can see that the new θ equals to 88.6103, which means that the camera coordinate are not skewed much and the X and Y axes in the image frame are almost perpendicular to each other. The image center does not coincide with the principle point and is offset by (114.1,52.9), as the results in the experiments. Also the α and β values are reasonable with little difference from the real value.

6 Code

There are three codes in the uploaded code file. The first file GetCoordinate.m is the procedure of import data from .xls file for world coordinates, and getting image coordinates with mouse clicking. The second file parameters.m computes the calibration matrix and parameters. The third file reconstruct.m is the procedure of reconstruction.