Assignment 1: Geometric Camera Models & Calibration

Practical Report

Chuan Long

1. Experiment Procedure

1.1 Date capturing and Methods used

Glue the two checkerboards to the corner of the wall. Make these two checkerboard planes to be orthogonal to each other by using the orthogonal relation in the wall corner. Then, I set up the 3D world coordinate including the origin, X axis, Y axis and Z axis. Last, I selected six points in the checkerboard and measured their corresponding world coordinates by using a ruler (1 mm represents 1 unit). See Fig.1, the selected points are marked by the red circle, and the three green directed lines build up the world coordinates.

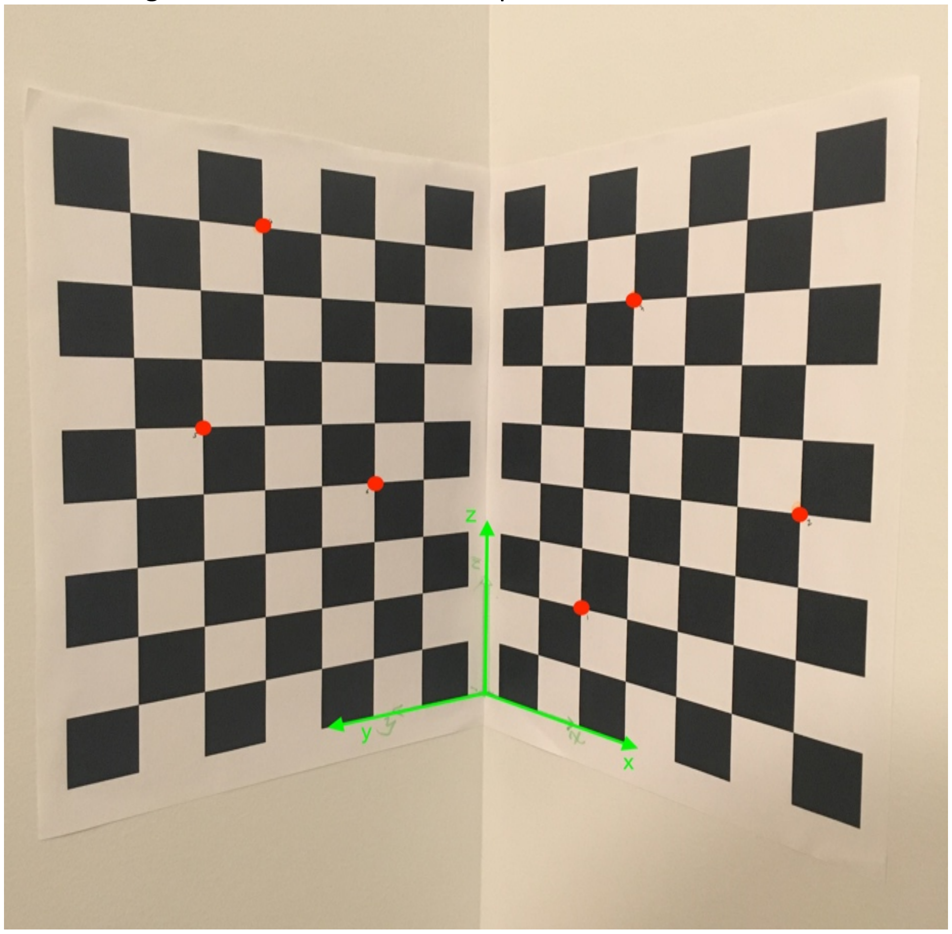


Fig.1

1.2 Type of Camera

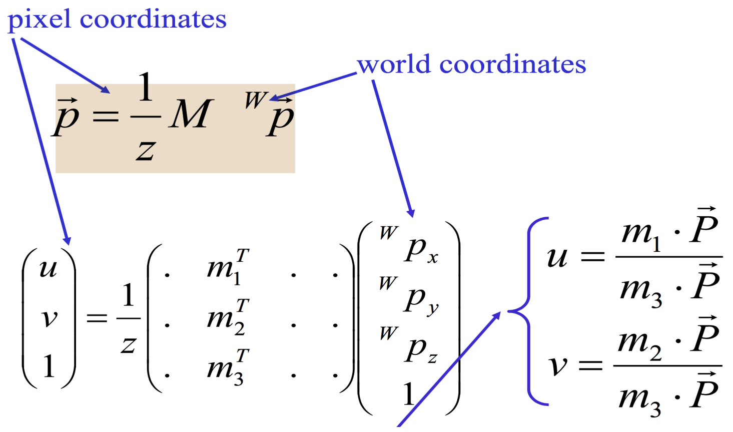
Camera	iPhone 6s rear camera
Aperture	f/2.2
Pixel	12 Megapixel
Resolution	3000 * 4000
Focal Length	4.15mm(err 29mm)

1.3 Type of Sensor

Sensor Size	4.8 mm * 3.6 mm
Pixel size	1.22 µm
Sensor Type	1/3"
Aspect Ratio	4:3

1.4 Setup of LSE

First, we have the equation about transform the world coordinates to image pixels, which is:



$$\vec{p} = \frac{1}{z} M \, {}^{W}\vec{p}$$

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{z} \begin{pmatrix} . & m_1^T & . & . \\ . & m_2^T & . & . \\ . & m_3^T & . & . \end{pmatrix} \begin{pmatrix} {}^{W}p_x \\ {}^{W}p_y \\ {}^{W}p_z \\ 1 \end{pmatrix} \quad \left\{ \begin{aligned} u &= \frac{m_1 \cdot \vec{P}}{m_3 \cdot \vec{P}} \\ v &= \frac{m_2 \cdot \vec{P}}{m_3 \cdot \vec{P}} \end{aligned} \right.$$

For each feature point, we have:

$$(m_1 - u_i m_3) \cdot \vec{P}_i = 0$$
$$(m_2 - v_i m_3) \cdot \vec{P}_i = 0$$

And then, we stack all these measurements of i = 1…n points into a big matrix.

$$\begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \cdots & \cdots & \cdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

To calculate the M matrix, we use Least Squares Method. For one point, there are 2 corresponding homogeneous linear equations. There are 12 homogeneous linear equations since there are 6 selected points in twelve variables, which are the coefficients of the calibration matrix M. For the very first coding part, we have to calculate the big matrix by using world coordinates and camera coordinates of the selected points that we collected before. And then, in this case, I used the Singular Value Decomposition(SVD) method powered by matlab to estimate the M matrix, a 1*12 matrix, and finally I transformed this 1*12 matrix to a 3*4 matrix, which is the final calibration matrix. The detailed process would be listed in the next session.

1.5 Solution strategy:

a) ***Collect:*** First of all, we have to collect the camera coordinates of the corresponding points that we selected at the world coordinates. After load an image into the matlab workspace, I used ginput() to click the selected points, and then I stored the dataset of camera coordinates of selected points into a matrix.
b) ***Estimate:*** Then, I declare a zero matrix with size of 12 * 12. And I coded to calculate the corresponding value to fill in this zero matrix, made it become the big matrix that used to estimate the calibration matrix. And then, utilized the SVD method, I got a 1*12 matrix and then transformed it into a 3*4 matrix.
c) ***Decompose:*** Now that we have to decompose the estimated calibration matrix to get the values of the intrinsic parameters and extrinsic parameters, mainly using the fact that the rows of a rotation matrix have unit length and are perpendicular to each other yields immediately. The parameters are listed behind.
d) ***Reconstruct:*** Finally, to make sure that I estimated the matrix and decomposed it in a right way. I reconstructed the the camera coordinates by calculate the product of calibration matrix and the world coordinates of the points. I coded to generate the world coordinates of all the 160 points on the checkerboard (because the length of one side of a check is 28mm, what I need to do it to measure the world coordinates of two initial points, and then generate the left points based on the initial point and the length of side). As the requirement of the assignment, I plotted all the reconstructed points on the checkerboard image. See Fig.2. Note that the reconstructed camera coordinates are

marked by red dot, and the selected points are marked by the green circle.

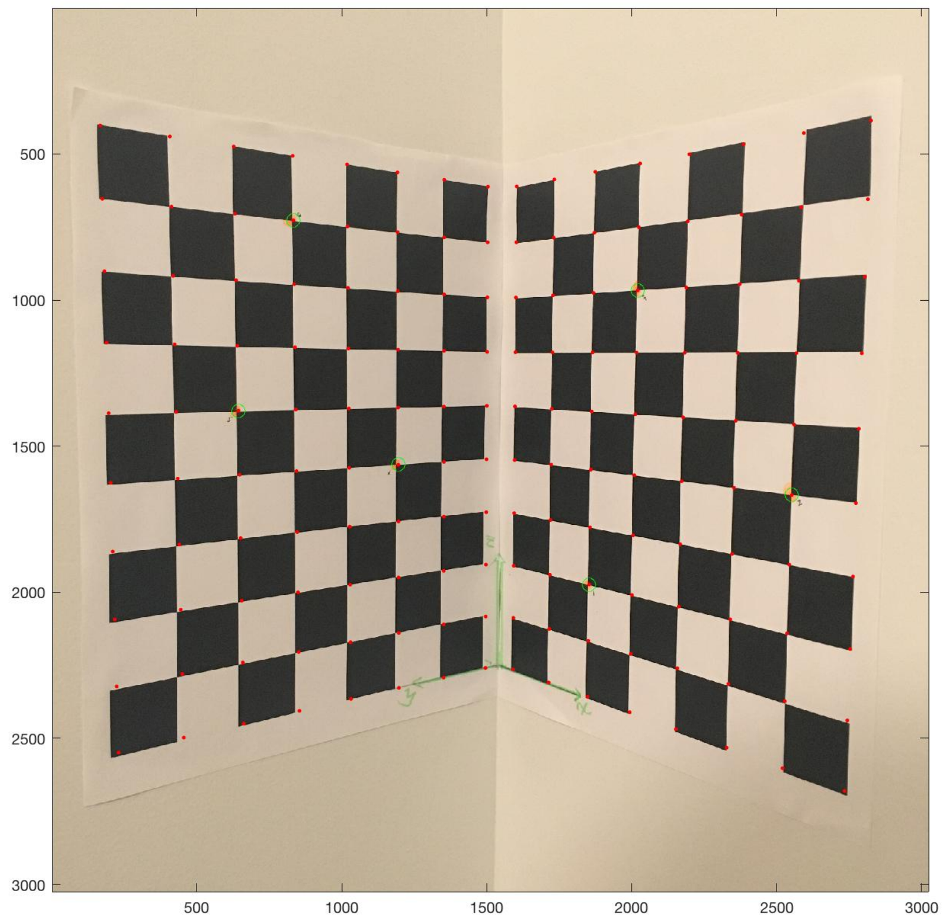


Fig.2

2. Intrinsic Parameters

U_0	1514.48613799808
V_0	1531.44839149705
θ	1.5707963267949
α	3661.26167336543
β	3637.64697022307

3. Extrinsic Parameters

R1	$[-0.644948727314538 \quad 0.764115271096525 \quad -0.0129996773976345]$
R2	$[-0.075644861750386 \quad -0.0469025752495532 \quad 0.996031125681183]$
R3	$[0.760472875273091 \quad 0.643372365673289 \quad 0.088051150258429]$
T	$[-3.03530470374574 \quad -113.374427140428 \quad -568.670789495391]$
$[\boldsymbol{Rz}, \boldsymbol{Ry}, \boldsymbol{Rx}]$ **in radians**	$[-3.02483790422824 \quad -0.864041011395236 \quad 1.4347825509297]$
$[\boldsymbol{Rz}, \boldsymbol{Ry}, \boldsymbol{Rx}]$ **in degree**	$[-173.310445623475 \quad -49.5059032791621 \quad 82.206984687286]$

4. Discussion and Critical Assessment

4.1 Reconstruction

To verify the estimated calibration matrix that I got before, one of the simple method is to reconstruct the points at the world coordinates to camera coordinates. Firstly, I chose all the world coordinates to be transformed to the camera coordinates. Because there are totally 160 points (80 points for one checkerboard), I haven't measure the world coordinates by myself but coded to generate them by using the fact that the world coordinate of one point is relative to its neighbors. All I need to do is to measure the two points that the closest to the origin point, and then we can get the left points by adding the length of side of a check sequentially. And then, I coded to calculate the product of the calibration matrix and the world coordinates, and plot the camera coordinates on the checkerboard image. (The figure is shown before) And we can see that the calibration matrix has a good performance because most of the points being plotted at the almost right place. For some points that are far from the origin point may not being plotted very precisely because the world coordinates corresponding to these points are also not so precise since in the real world, there must be some error in the world coordinates that I generated automatically. But this result means the calibration matrix has been estimated in a right way.

4.2 Decomposition

To verify whether I decomposed the extrinsic parameters and intrinsic parameters in a right way. I used the following equation:

$$\rho(\mathcal{A} \quad \boldsymbol{b}) = \mathcal{K}(\mathcal{R} \quad \boldsymbol{t}) \cdot$$

And here is the result:

$$p * M = \begin{array}{cccc} -1209.84797981621 & 3771.84927922547 & 89.0291164949387 & -872357.072554003 \\ 889.455659454879 & 814.676563791814 & 3758.05539901477 & -1283306.10725223 \\ 0.760472875273091 & 0.643372365673289 & 0.088051150258429 & -568.670789495391 \end{array}$$

$$K * [R \ \ T] = \begin{array}{cccc} -1209.60042867789 & 3772.00448548217 & 85.7570258791027 & -872357.072554003 \\ 889.455659454879 & 814.676563791814 & 3758.05539901476 & -1283306.10725223 \\ 0.760472875273091 & 0.643372365673289 & 0.088051150258429 & -568.670789495391 \end{array}$$

The results of these two products are almost the same, which means I have calculated the K matrix (formed by intrinsic parameters) and $[R\ T]$ (formed by extrinsic parameters) on the right track. The tiny difference between them may be caused by the loss precision during the process of calculation.

4.3 Intrinsic Parameters

θ: This parameter shows the the angle between the two image axes in the camera coordinate system, my result is very close to 90 degrees, which means the angle is almost a right angle without any big error.

U_0 and V_0: Since the actual origin of the camera coordinate system is at a corner of the retina, but the origin of the calibration matrix is at the center of the sensor, so we have to add the offset U_0 and V_0 to make it coincide with the camera coordinate system. And these two parameters are about half of the length and the width of the physical retina. The resolution of this figure is 3024 * 3024, and the values of U_0 and V_0 are approximately half of the length and width of the dimensions. So, these two values also make sense at some extend.

α and β: now that we have equations which are:

$$\alpha = k * f \text{ and } \beta = l * f.$$

And f represents the focal length of the camera, k and l represent the pixel density by width and by height. And we can get the values of k and l according to the parameters that we google before.

$$k = 4000\, pixels\ /\ 4.8\, mm = 833.3333\, pixels\ /\ mm$$

$$l = 3000\, pixels\ /\ 3.6\, mm = 833.3333\, pixels\ /\ mm$$

We have $f = 4.15\, mm$, and then:

$$\alpha = k * f = 833.3333 * 4.15 = 3456.95$$

$$\beta = l * f = 833.3333 * 4.15 = 3456.95$$

And we have decomposed the calibration matrix and got the values of them which are about 3661 and 3637 accordingly before, and they are almost the same as the calculated standard data, so both of them make sense at this time.

4.4 Extrinsic Parameters

For the rotation part, note that the rotation matrix in radian has become $[Rz, Ry, Rx]$. So, the world coordinate has rotated about 173 degrees around the Z-axis, 50 degrees around the Y-axis, 82 degrees around the X-axis.

For the translation part, please note that the translation vector is Tz is roughly the distance between the camera frame and the checkerboard pattern. And before the calculation part, I have measured the distance between the optical center and the origin of the world coordinate which is about 50cm. The value of Tz is 568.670789495391, so this value is also the real distance approximately.

For the Extrinsic parameters, we get different values each time because they are extrinsic, which are depend on the relative position between the world coordinate and the camera coordinate. So it is very reasonable we got these parameters like this.

Bonus Question

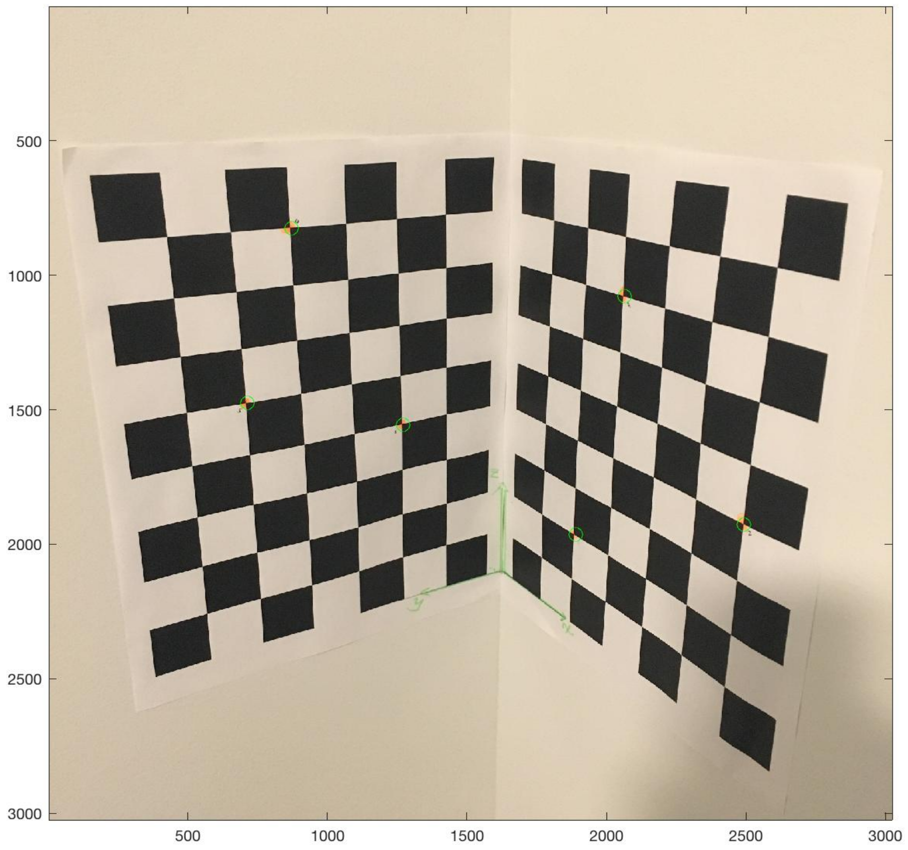


Fig.3

I took another checkerboard image (see Fig.3) and adopted the same selected points at the world coordinates (selected points marked by the green circle), and I calculated the intrinsic parameters by the method that have been discussed:

U_0	1554.14112173269
V_0	1430.63652827533
θ	1.5707963267949
α	3490.79064605365

β	3523.29181857043

Compare with these intrinsic parameters for Fig.1 and Fig.3 are as following:

	Figure 1	**Figure 3**
U_0	1514.48613799808	1554.14112173269
V_0	1531.44839149705	1430.63652827533
θ	1.5707963267949	1.5707963267949
α	3661.26167336543	3490.79064605365
β	3637.64697022307	3523.29181857043

The reconstructed camera coordinates of the points for the second image is as following:

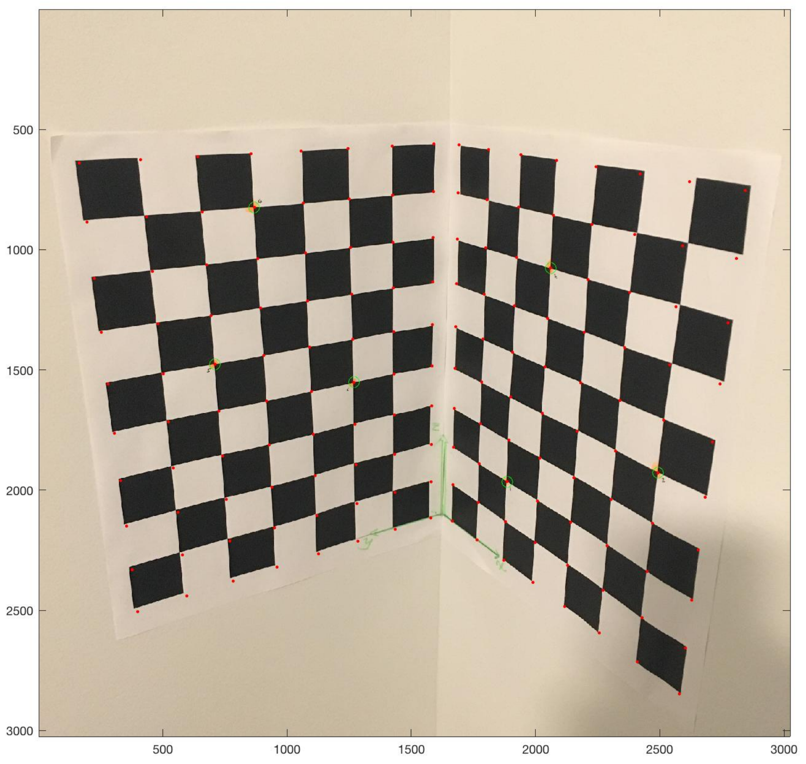


Fig.4