ASSIGNMENT 1 (ELL793)

Camera Calibration

Harsh Pandey-2021CSY7545

Nikki Tiwari-2021SIY7560

Objective

The goal of this assignment is to find intrinsic and extrinsic camera calibration parameters of our mobile phone's camera with two (or three) orthogonal checkerboard images so as to be able to capture images of objects from known locations and with a known camera model.

Creating Dataset:

World Coordinate System - Three orthogonal checkerboard patterns have been glued together onto the corner of the walls. Then we have defined the world coordinate system with its origin as the dark red dot where all the three orthogonal axes coincide. (See Figure 1), x-axis rightward, and y-axis leftward and z-axis upwards. The red dots are the points to be measured.

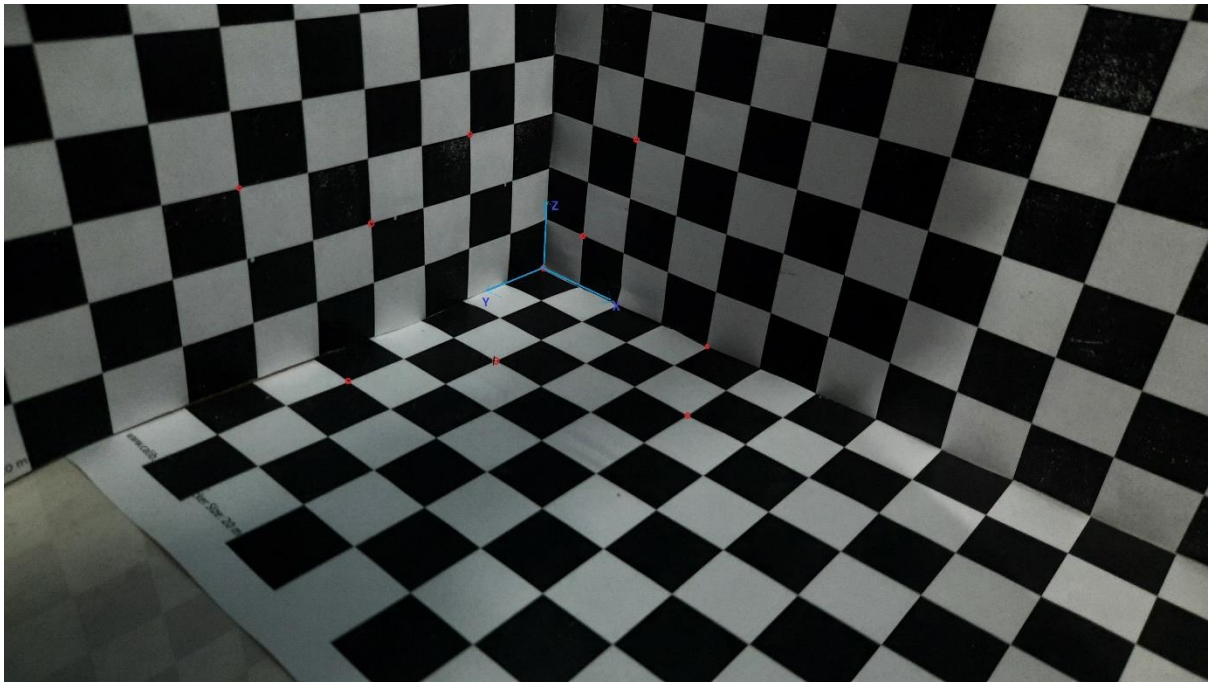

Figure 1

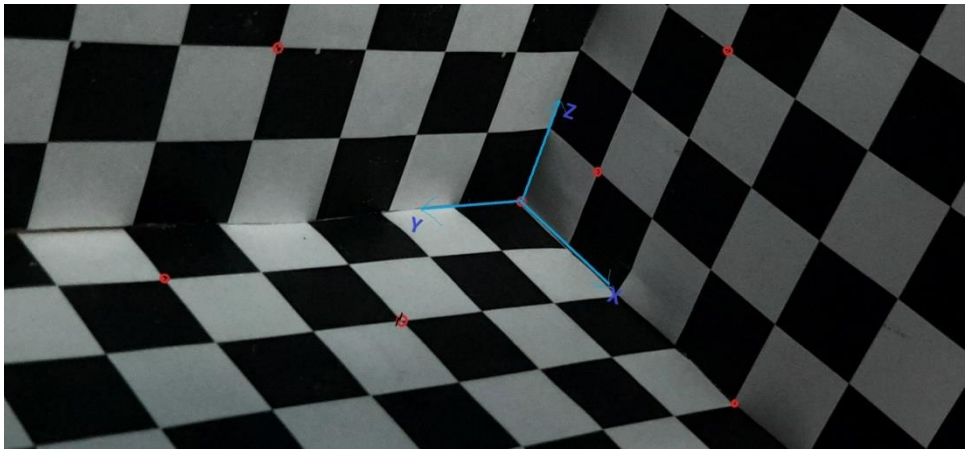

Figure 2

Image Coordinates (2-D points) Our dataset consists of 10 points. The unit of image coordinates is in pixels. We have found the image coordinates with mouse clicking and finding the two-pixel coordinates of the point save these points into a file.

World Coordinates (3-D points) According to the 10 picked points, we have computed the world coordinates of these points by counting the squares from the origin and saving the measurement of world coordinates in a file.

Normalizing the data:

Now we have normalized the data such that the centroid of 2D and 3D points are at origin and the average Euclidean distance of 2D and 3D points from the origin is √2 and √3, respectively. Find the transformation matrices T and U that achieve this for 2D and 3D respectively, i.e., ˆx = Tx and Xˆ = UX where x and X are the unnormalized 2D and 3D points in homogeneous coordinates.

Reason to normalize the points before DLT:

Data normalization is an essential part of the DLT algorithm. For all points, scale the homogenous vectors such that the last entry becomes 1, i.e., x = (x, y, 1) > and X = (x, y, z, 1) >. The first step is to transform the 2D and 3D input points so that the centroid of the points is at the origin and that the average Euclidean distance from the origin is √ 2 (for the image points) and √ 3 (for the object points). The distance to the origin is p $\sqrt{x^2 + y^2}$ p and $\sqrt{x^2 + y^2 + z^2}$, respectively. Find transformation matrices T and U such that the normalized image points are $\widehat{x_k} = Tx_k$ and the normalized object points are $\widehat{X_k} = UX_k$ (again, the last entries are 1).

1. The source and target coordinate data are usually **noisy**. Without normalization, the data from source would have two orders of magnitude larger variance than from target (or vice versa).
2. The error estimation usually finds parameters in a least-squares sense - hence the best statistical estimate is found only if variances of the parameters are the same (or known beforehand, but it is more practical just to normalize the input).
3. Direct solvers do not like poorly scaled problems because numerical instabilities appear (e.g., dividing very large number by a very small number easily leads to numerical overflow).

So, normalization is essential not only for numerical stability, but also for more accurate estimation in presence of noise and faster solution (in case of iterative solver).

Verification –

Performed on entire dataset (except excluded point, for details see **Description** section)

> Without Normalization we got re-projection error of = 5.802111556001503
>
> With Normalization we got re-projection error of = 5.79767845113394

So, we can see clearly that **normalization reduced the re-projection error**

But after choosing optimal points and using higher precision we got

> Without Normalization we got re-projection error of = 0.281807573920669
>
> With Normalization we got re-projection error of = 0.281811199860872

So, here normalization does not play a significant role.

Normalized Projection Matrix:

Our goal is to compute the calibration matrix M that relates the world space points and to the corresponding image space points. The equation

p = (1/ Z) MP can be solved equivalently.

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

which in turn can be equivalently expressed in a matrix vector multiplication form:

$$Qm = 0$$

$$\begin{bmatrix} P_{1x} & P_{1y} & P_{1z} & 1 & 0 & 0 & 0 & 0 & -u_1P_{1x} & -u_1P_{1y} & -u_1P_{1z} & -u_1 \\ 0 & 0 & 0 & 0 & P_{1x} & P_{1y} & P_{1z} & 1 & -v_1P_{1x} & -v_1P_{1y} & -v_1P_{1z} & -v_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ P_{nx} & P_{ny} & P_{nz} & 1 & 0 & 0 & 0 & 0 & -u_nP_{nx} & -u_nP_{ny} & -u_nP_{nz} & -u_n \\ 0 & 0 & 0 & 0 & P_{nx} & P_{ny} & P_{nz} & 1 & -v_nP_{nx} & -v_nP_{ny} & -v_nP_{nz} & -v_n \end{bmatrix} \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

where the 3x4 matrix M is flattened into a 12-dimensional unit vector **m**.

So, we could solve this using homogeneous least squares

We used SVD to find the least eigen vectors of the matrix for its solution.

Reprojection root mean square error:

1.The projection matrix is correctly verified by computing the RMSE between the 2D points marked by us and the estimated 2D projections of the marked 3D points.

2.We calculate the difference between the given 2D point and the estimated 2D points (calculated using the projection matrix we calculated).

3.The difference is squared and averaged over all the points and its square root is called re-projection error.

Intrinsic and Extrinsic Parameters Computation:

The camera calibration matrix M is:

$$\begin{bmatrix} \alpha r_1^T - \alpha \cot\theta\, r_2^T + u_0 r_3^T & \alpha t_x - \alpha \cot\theta\, t_y + u_0 t_z \\ \dfrac{\beta}{\sin\theta} r_2^T + v_0 r_3^T & \dfrac{\beta}{\sin\theta} t_y + v_0 t_z \\ r_3^T & t_z \end{bmatrix}$$

This is shown as a 3x3 and a 3x1 components. This can be written as $\boldsymbol{M} = \boldsymbol{\rho}(\boldsymbol{A}, \boldsymbol{b})$. ρ is a scale factor**,** which is used to scale the recovered calibration matrix **M** to be a canonical form.

$$\rho(A \;\; b) = K(R \;\; t) \Leftrightarrow \rho \begin{pmatrix} a_1^T \\ a_2^T \\ a_3^T \end{pmatrix} = \begin{pmatrix} \alpha r_1^T - \alpha \cot\theta\, r_2^T + u_0 r_3^T \\ \dfrac{\beta}{\sin\theta} r_2^T + v_0 r_3^T \\ r_3^T \end{pmatrix}$$

Given the fact that the rows (or columns) of a rotation matrix forms a right-handed orthonormal coordinate system, it can be inferred that its rows must have unit length and are perpendicular to each other. Thus, we have:

$$\begin{cases} \rho = \varepsilon / |a_3| \\ r_3 = \rho a_3 \\ u_0 = \rho^2 (a_1 \cdot a_3) \\ v_0 = \rho^2 (a_2 \cdot a_3) \end{cases}$$

where $\varepsilon = \mp 1$.

The method in the book uses the properties of rotation matrix and the sign of θ to determine all the other of the 11 parameters.

$$\begin{cases} \cos\theta = -\dfrac{(a_1 \times a_3) \cdot (a_2 \times a_3)}{|a_1 \times a_3||a_2 \times a_3|} \\ \alpha = \rho^2 |a_1 \times a_3| \sin\theta \\ \beta = \rho^2 |a_2 \times a_3| \sin\theta \\ r_1 = \dfrac{(a_2 \times a_3)}{|a_2 \times a_3|} \\ r_2 = r_3 \times r_1 \end{cases}$$

Now the K matrix containing the intrinsic parameters can be recovered.

The translation parameter t can be computer with the equation.

$$t = (t_1, t_2, t_3)^T$$

$$\mathrm{t} = \rho\kappa^{-1}\mathrm{b}$$

Description-

Making the dataset and getting initial values-

- We created a dataset of 10 points by mapping the 3d world coordinates to the 2d image.
- We did it manually by marking the points of interest in the clicked image from our camera and mapping it to the 3d world coordinates.
- Since we only need 6 points to calculate the projection matrix, we initially used all the points to calculate the projection matrix.

Some Filtering of points-

- Then we looked at the re-projection errors of the points, in which we found 1 point
    - 3D [5, 2,0] and 2D: [2271,1366]
    - This point was giving squared error of e^2= 1825 i.e. error = 42.720018.
- So, we discarded this point as all other errors were comparatively very less (average **error** $\cong$ **7**).

Final calculation-

- Now we selected the top 6 points to calculate the projection matrix.
- From the projection matrix we calculated the re-projection error for the 6 training points (Resultant error = 0.281811).
- We also calculated the re-projection error for the entire dataset (Resultant error = 7.611439).
- We then passed the projection matrix to get the intrinsic and extrinsic values of the parameter which are shown in detail in the result section.

Result-

Dataset Used

3D Points			2D Points		Remark
X	Y	Z	x	y	
[[0,	0,	0,	1792,	876],	For Calculation
[1,	5,	0,	1143,	1251],	For Calculation
[2,	0,	3,	2102,	452],	For Calculation
[0,	6,	3,	780,	610],	For Calculation
[1,	0,	1,	1923,	769],	For Calculation
[4,	0,	0,	2335,	1139],	For Calculation
[2,	3,	0,	1634,	1185],	For Testing
[5,	2,	0,	2271,	1366],	Bad point with **error = 42.7200** So, **Discarded**
[0,	4,	2,	1217,	731],	For Testing
[0,	2,	3,	1547,	435],	For Testing

--------**Projection Matrix**---------

[[2.51630751e+01, -1.91314958e+02, -3.89159065e+01, 1.79185304e+03]

[1.18446017e+01, 1.08430055e+01, -1.76169364e+02, 8.75728887e+02]

[-4.73758923e-02, -3.99615762e-02, -2.81141966e-02, 1.00000000e+00]]

--------**Reprojection error**--------

Reprojection Error for Train Set: 0.281811199860872

Reprojection Error for Test Set: 7.611439

--------**Camera values**----------

Intrinsic and Extrinsic

The difference between the centre of camera coordinate frame and image centre(x_0, y_0):

(110.8948, 58.16277)

Intrinsic camera matrix:

[[1.62431e+02, 8.20366e+00, 1.10894e+02]

[0.00000e+00, -1.67060e+02, 5.8162e+01]

[0.00000e+00, 0.000000e+00, 6.8057e-02]]

Translation matrix:

[1.00632, -0.12639, 14.69347]

Rotation matrix:

[[0.64598555, -0.76334138, -0.00355041]

[-0.31324804, -0.2693251, 0.91068087]

[-0.69611661, -0.58717452, -0.41309532]]

Camera focal length

in the x axis in pixels (f_x): 162.43177

Camera focal length

in the y axis in pixels (f_y): -167.06466

Skew parameter (s): 8.20366

The angle between the x-axis and y-axis in the image plane

Θ (in degree) = 87.10871

The value of alpha (α) is: 162.4317749384117

The value of beta (β) is: 166.851995882425

The value of x_0 is: 110.8948

The value of y_0 is: 58.16277