# ENPM673 Project 3

Vineet Kumar Singh (ID: vsingh03)

April 2023

## Contents

# 1 Problem 1

The camera projection matrix has 12 unknowns and after normalizing the number of unknowns is 11 so at least 11 equations are needed to solve this equation. Each matching point generates 2 equations, and thus **at least 6 matching points** are needed to solve the projection matrix.

## 1.1 Pipeline for Camera Calibration

The pipeline to extract the projection matrix for the given image is :

1. Read the image and mark the corners for the checkerboard section.
2. Extract the pixel coordinates for the corners and get the world coordinates for the same corner points and match the points.
3. Calculate the elements of the A matrix ($A\vec{m} = 0$) for the available corner points.
4. Minimizing the loss function $L(p, \lambda) = p^T A^T A_p - \lambda(p^T p - 1)$ which is just solving the equation $A^T Ap = \lambda p$. The Eigenvector corresponding to the smallest Eigenvalue gives the p vector.
5. Rearrange the p vector into a 3*4 matrix which is the Projection Matrix P. The matrix can be normalized by dividing each element of the matrix by its last element i.e. (3,4)th element.
6. The P matrix is then decomposed to get the Intrinsic and extrinsic matrix. First, the matrix is written as $P = \Big[M| - MC\Big]$. C is the null vector of P i.e.
7. C is calculated by SVD of P. $P = UDV^T$ and C is the unit singular vector of P corresponding to the smallest singular value (the last column of V). This is the translation matrix.
8. M matrix is composed of Rotation and Camera Intrinsic matrix i.e. $M = KR$. This is decomposed by RQ factorization by using the Gram-Schmidt method.
9. Thus the result is the K matrix, translation, and rotation matrix.
10. The given world points are then reprojected into the camera frame using the calculated projection matrix.
11. The reprojected points are normalized using its z-coordinate. The resulting points are then used to calculate the reprojection error using norm calculation between two coordinates.

## 1.2 Intrinsic Camera Matrix calculation

The calculation of the Intrinsic Camera Matrix is done by the process of RQ decomposition of the Left $3 \times 3$ Submatrix of P. The RQ decomposition is done using the Gram-Schmidt process as explained below with mathematical equations. The Gram-Schmidt method is an iterative process that generates an orthonormal basis for the columns of $M$, which can then be used to construct the matrices $K$ and $R$.

Let us assume, given an $m \times n$ matrix $A$, the RQ decomposition of $M$ expresses it as a product of an upper triangular matrix $K$ and an orthogonal matrix $R$.

The steps for the RQ decomposition using the Gram-Schmidt method:

1. Let $M = [a_1, a_2, \ldots, a_n]$, where $a_i$ is the $i$th column of $A$.
2. Initialize $q_n = \frac{a_n}{|a_n|}$.
3. For $i = n-1, n-2, \ldots, 1$, below steps are repeated:
   - Compute the projection of $a_i$ onto $q_{i+1}, q_{i+2}, \ldots, q_n$:

$$\hat{a_i} = a_i - (q_{i+1} \cdot a_i)q_{i+1} - (q_{i+2} \cdot a_i)q_{i+2} - \cdots - (q_n \cdot a_i)q_n \tag{1}$$

   - If $|\hat{a_i}| \neq 0$, set $q_i = \frac{\hat{a_i}}{|\hat{a_i}|}$.
   - If $|\hat{a_i}| = 0$, set $q_i = 0$.
4. Let $R = [q_1, q_2, \ldots, q_n]$ be the matrix whose columns are the orthonormal vectors obtained in steps 2 and 3.
5. Compute the upper triangular matrix $R$ by setting $r_{ij} = q_i \cdot a_j$ for $1 \leq i \leq j \leq n$.
6. The RQ decomposition of $M$ is then given by $M = KR$.

The K matrix is the Camera Intrinsic Matrix, and R is the Rotation matrix.

## 1.3 P matrix

The result of Projection Matrix P is :

```
Projection Matrix:
 [[ 2.87364445e+01 -1.75735415e+00 -7.00687538e+01  7.56890519e+02]
 [-2.01369011e+01  6.58890120e+01 -2.22140404e+01  2.13263797e+02]
 [-2.77042391e-02 -2.59559759e-03 -3.13888009e-02  1.00000000e+00]]
```

Figure 1: Projection matrix

## 1.4 Camera Intrinsic, Translation and Rotation Matrices

The result of Camera Intrinsic, Translation, and Rotation matrices :

## 1.5 Reprojection Error for points

The Reprojection error for each point is given in Figure 3

## 1.6 Problems Faced and Solutions Implemented

The major issue faced in this problem was the decomposition of the P matrix into Intrinsic and Rotation matrices using RQ decomposition. The issue with RQ decomposition is that the result of RQ decomposition isn't unique. So the K matrix was checked to ensure that the diagonal elements were not negative. This was done manually to ensure the correctness of the K matrix.

```
Intrinsic Camera Matrix:
  [[ 6.79123314e+01 -7.93927704e-02  3.35620430e+01]
 [ 2.81035877e-15  6.76190373e+01  2.58454266e+01]
 [ 6.23901708e-18  2.63787729e-19  4.19466186e-02]]


Rotational Matrix:
  [[ 0.74948643  0.00587017 -0.66199368]
 [-0.0453559   0.99806642 -0.04250013]
 [-0.66046418 -0.06187859 -0.74830349]]


Translational Matrix:
  [[-0.64862355]
 [-0.30183152]
 [-0.69751919]
 [-0.04064735]]


Normalized Translation Matrix:
  [[15.95734064]
 [ 7.42561435]
 [17.16026409]
 [ 1.        ]]
```

Figure 2: Projection matrix

```
Reprojection error (Distance in pixel units) for each point in sequence:
 [[0.28561277]
 [0.97258285]
 [1.03608178]
 [0.45408629]
 [0.19089832]
 [0.31899208]
 [0.19594241]
 [0.30829603]]


Mean Error [x, y, z]:
 [3.91405633e-04 9.32181370e-05 0.00000000e+00]
```

Figure 3: Reprojectinon Error for each given point

# 2 Problem 2

## 2.1 Pipeline

The pipeline to calibrate the camera using multiple calibration images using the OpenCV library. The below method uses OpenCV's *calibrateCamera()* function to solve the problem.

1. Capture a set of calibration images using the camera that is to be calibrated. The images should contain a known calibration pattern such as a chessboard.

2. Define the calibration pattern parameters, such as the number of corners in each row and column (6*9 in this problem), and the size of the square (21.5mm in this problem).

3. Load the calibration images into a list.

4. A list of object points is created which are the known 3D coordinates of the calibration pattern corners in the real world. The object points are the same for all calibration images and are defined based on the calibration pattern parameters.

5. Corners are detected in the calibration pattern in each calibration image using OpenCV function *findChessboardCorners()*. Store the detected corners in a list and the corresponding object points for each image. Figure 4 shows the detected corners.

6. OpenCV function *calibrateCamera()* is used to obtain the camera calibration parameters, including the camera matrix, distortion coefficients, and rotation and translation vectors for each calibration image. The function takes as input the object points and the image points detected in step 5, as well as the size of the calibration images.

7. The images are then reprojected using the OpenCV function *projectPoints()*. The points in the reprojected image are then compared with the original object points and the error gives the reprojection error.

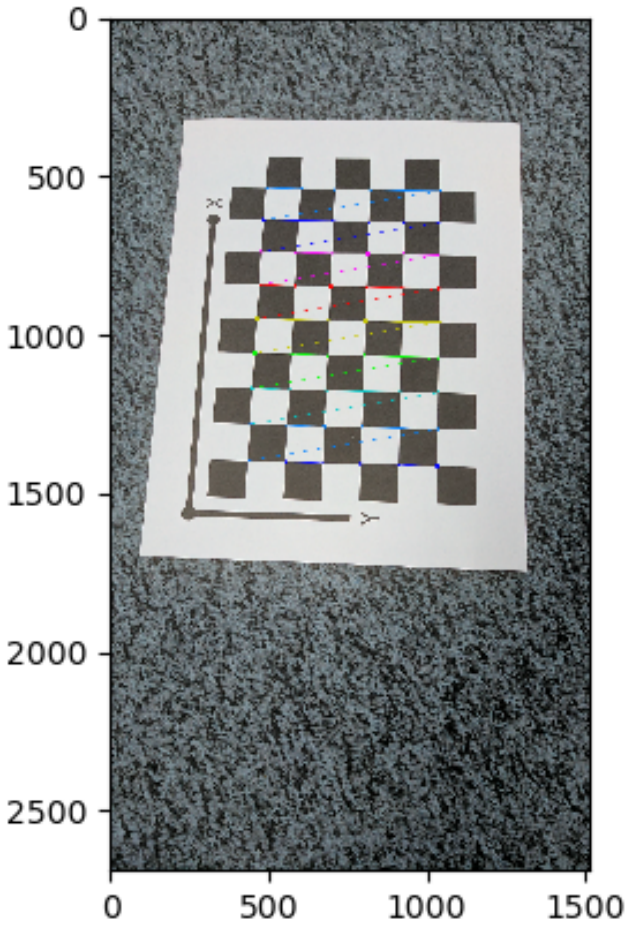


Figure 4: Detected chessboard corners

## 2.2 Reprojection Error and K matrix

The result of the Reprojection error for each given image is shown in Figure 5.
The result of the K matrix for the camera is shown in Figure 6.

```
Error:  0.07811860604254106
Error:  0.09300490101596924
Error:  0.11927474143340673
Error:  0.1434668335059415
Error:  0.06785202612852255
Error:  0.07945822777751176
Error:  0.11452972761669802
Error:  0.06683274545547946
Error:  0.07528112226845364
Error:  0.08243326538824611
Error:  0.11240184156012407
Error:  0.12624306572562138
Error:  0.1230613007443954
Mean error:  0.09861218497407008
```

Figure 5: Reprojection error for all the 13 images

```
K matrix:
 [[2.04039471e+03 0.00000000e+00 7.64587598e+02]
 [0.00000000e+00 2.03217467e+03 1.35929491e+03]
 [0.00000000e+00 0.00000000e+00 1.00000000e+00]]
```

Figure 6: K matrix for the camera

## 2.3 Ways to improve K matrix accuracy

The matrix is composed of the internal camera parameters which include the focal length of the camera lens in the x and y direction, Image center coordinates, and Skew parameter.

Possible steps to increase the accuracy of the Intrinsic matrix are:

1. Increasing the number of calibration images i.e. more images with different orientations and positions can improve the accuracy of the calibration.

2. Images that cover the entire field of view of the camera and images of objects at different distances.

3. Improving the quality of the calibration pattern. Using a high-quality calibration pattern with accurate, evenly spaced corners.

4. Ensuring that the calibration pattern is printed on a flat surface and is not distorted in any way.

5. Increasing the size of the calibration pattern improves the accuracy of the calibration.

6. Removing outlier points from the captured image. Usually removing the edge points in the image due to distortion (e.g. radial distortion).

## 2.4 Problems Faced and Solutions Implemented

The challenge in this section was the formation of the image points vector and ensuring the storing of corners was in the same sequence. To ensure this, each image corner was validated manually when storing in the vector to ensure the correct sequence as the object points.