Assignment 1

Name: Saraansh Tandon
Roll No.: 20171007

Note:

- All the images are present in the folder 'Camera_calibration_data' which is present in the same directory as the src folder.
- Link to images folder: https://drive.google.com/drive/folders/11IV3US398tHnVHyZLU-RWWHQ1449Sw1V?usp=sharing

Direct Linear Transform

1. In this part we were supposed to find the parameters of the camera and projection using the DLT method.
   a. Firstly a function called 'get_projection_matrix' was defined which took the world coordinates and the projected coordinates as parameters and returned the Projection Matrix for that image. This was done by creating a 40x12 matrix 'M' for 20 points and performing SVD on it. The Projection Matrix 'P' could then be found out by taking the singular vector corresponding to the smallest singular value (solution to the optimization problem) and reshaping it into a 3x4 matrix.
   b. Next the Projection Matrix was decomposed using the function 'get_params', to obtain the projection center, rotation matrix and calibration matrix. These parameters were then reported.
   Projection Matrix ->
   [[0.00278872 -0.00027259 -0.00107257 -0.69451402]
    [0.00058862 -0.00301548 0.0006177 -0.71946595]

```
 [ 0.00000051 -0.00000017  0.00000053 -0.00045058]]
```

Projection Center ->

```
 [407.86273265 -70.70895504 430.90265668]
```

Rotation Matrix ->

```
[[-0.72399544 -0.00002326  0.68980475]
 [-0.15889327 -0.97310334 -0.16680175]
 [ 0.67125519 -0.23036904  0.70451869]]
```

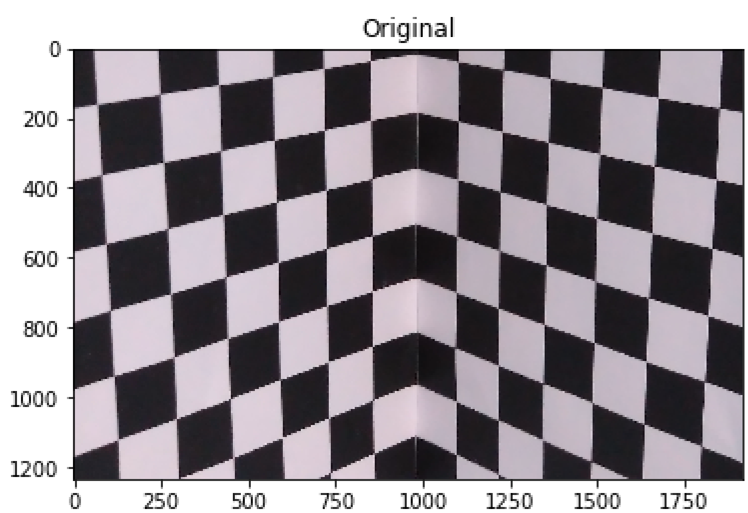
Calibration Matrix ->

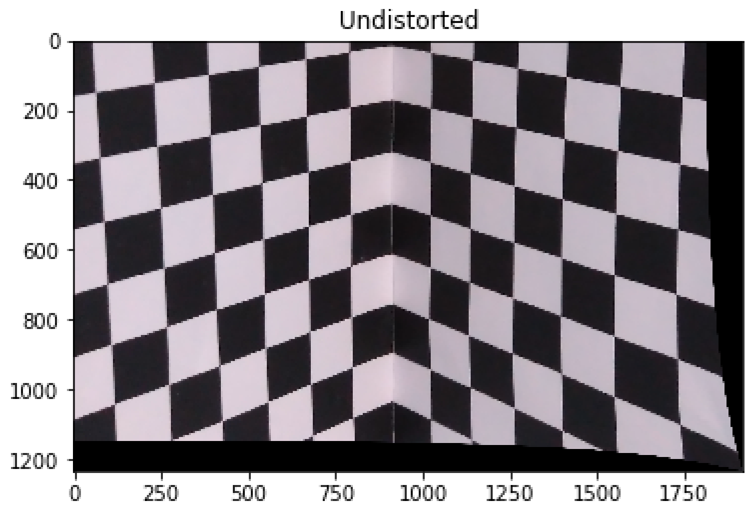
```
[[3634.85968162    1.39036898 1553.46706162]
 [  -0.         3607.10601233 2009.16702769]
 [  -0.            0.            1.        ]]
```

2. In this part we were supposed to implement the RANSAC variant of the calibration method.
    a. The method used in part 1 requires only 6 points to solve the problem and we have 20 at our disposal.
    b. So we perform multiple iterations(50) of the solution of part 1 using only 6 randomly sampled points in each iteration for obtaining the projection matrix. Then we use that projection matrix to project all 20 points and compare the results with the manually obtained ground truth(Squared error with a threshold of 15).
    c. In the end we report the projection matrix which gives the best results.
    d. **Best Projection Matrix ->**

```
[[ 0.00277593 -0.00027391 -0.00110957 -0.69441903]
 [ 0.00057603 -0.00303519  0.00058221 -0.71955758]
 [ 0.00000051 -0.00000018  0.00000052 -0.00045169]]
```

    e. We observe that the projection matrices obtained in the first and second part are very similar in the magnitude of their elements.
    f. We also observe that the number of matches between the ground truth and the projected points ranges from very low(even 0 in some cases) to very high(all correctly matched). This may mean that not all points are equally capable of representing the projections.
3. In this part we were asked to undistort the image and then perform the above calibration again.
    a. To do this, 10 points in the world coordinates and the image coordinates are annotated.
    b. Next the inbuilt function ‘cv2.calibrateCamera’ is used to find the distortion parameters.
    c. Next these parameters are used to undistort the image using the inbuilt function ‘cv2.undistort’ to get the undistorted image.
    d. **Distortion parameters ->**
    [-4.35001587e-35, -1.18741875e-28, -3.46961550e-38, -7.10548241e-38, -3.24817178e-22]
    e. We observe that the distortion parameters are extremely low, implies there is not much distortion in the image, and hence we conclude that the calibration parameters will also be approximately the same as those for the original image.





Zhang’s Method

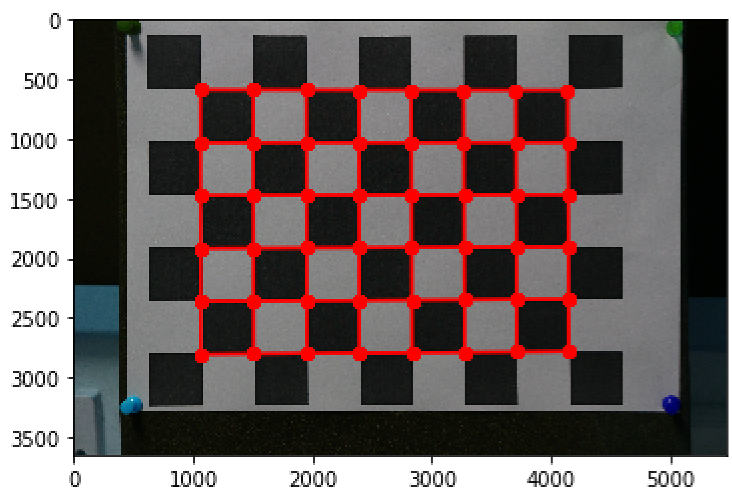
1. In this part we were supposed to perform camera calibration using the Zhang’s Method.
    a. Firstly the world coordinates for the checkerboard pattern are specified as a standard evenly spaced grid created using the meshgrid function.
    b. Next for each image the projected coordinates for all the corners are obtained using the ‘cv2.findChessboardCorners’ function. All of these points are stored along with their corresponding world coordinates.
    c. Next these world and projected coordinates are passed to the inbuilt ‘cv2.calibrateCamera’ function to obtain the calibration matrix.
    d. **Camera Calibration Matrix ->**
    [[13663.48159728 0. 3336.51275357]
     [0. 13681.38882946 1496.57984722]
     [0. 0. 1.]]
2. In this part we were supposed to use the result of the previous part to obtain the projection matrix, and use it to project the world

coordinates and display a wireframe of these projected points overlayed over the original image.

a. Firstly we use the rotation angles to compute the rotation matrix using the function 'get_rot_mat'.
b. Next we use this rotation matrix, the translation vectors and the calibration matrix to obtain the projection matrix as:

   P = [KR|KRC]

   P is also normalized.
c. Next this P is used to project the world coordinates to image points and a wireframe is constructed using these projected points.
d. We observe that although the wireframe is not perfectly aligned with the corners, it is very very close to them and is a very good approximation of the actual image coordinates.

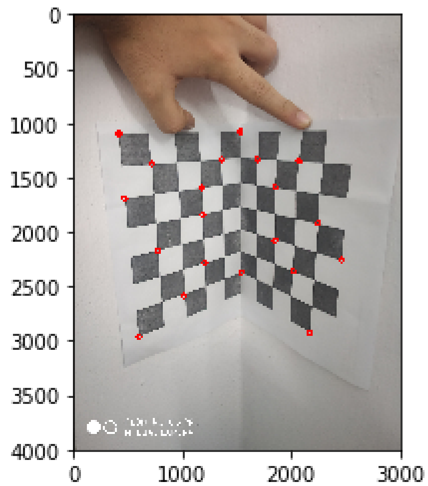


3. In this part we were supposed to report the origin's image based on the projection matrix.
   a. The image of the origin is the last column of the projection matrix.
   b. We observe that the projected image coordinates are again very very close to the actual image coordinates of the origin.
   c. **Image of origin according to projection matrix ->** [1060.06429, 587.72062, 1.0]
      Actual approximate origin in image -> [1076 600]

Here the origin is taken as the most top left corner.

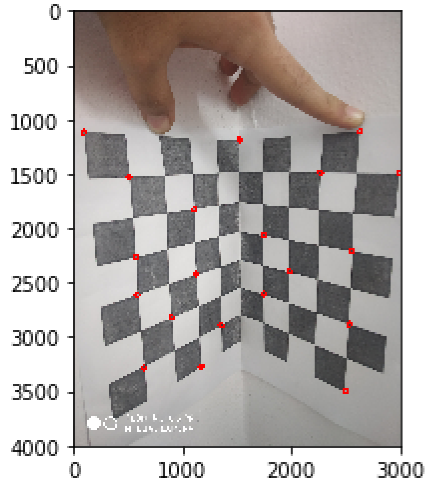
Hands on

1. We take three images with different focus levels and perform DLT on them. For this question the size of each square in the checkerboard is considered to be 25x25.

a. 

Calibration Matrix ->

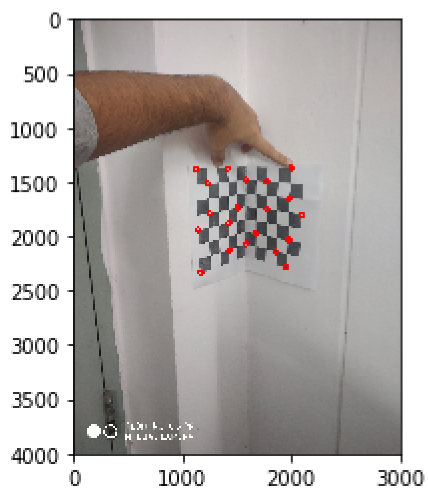
[[2847.58956262 -49.95300026 1476.81436548]
 [-0.   2959.56804575 1949.98773692]
 [-0.   0.   1.   ]]



b.

Calibration Matrix ->

[[2441.91660924   10.27400961 1585.45799169]
 [-0.   2534.40436684 1794.38218469]
 [-0.   0.   1.   ]]



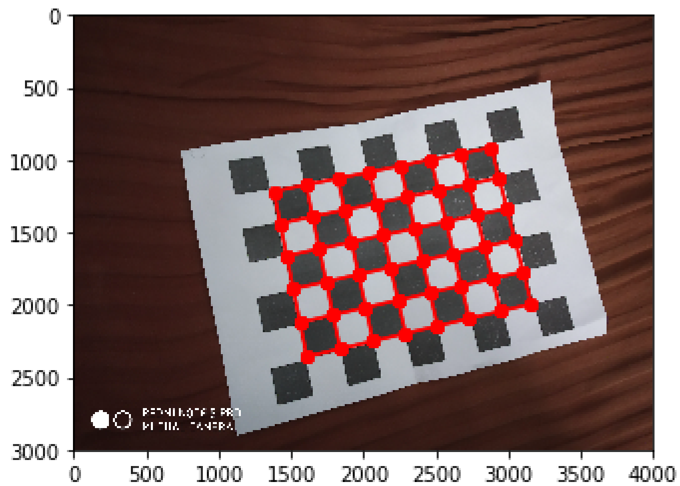
c.

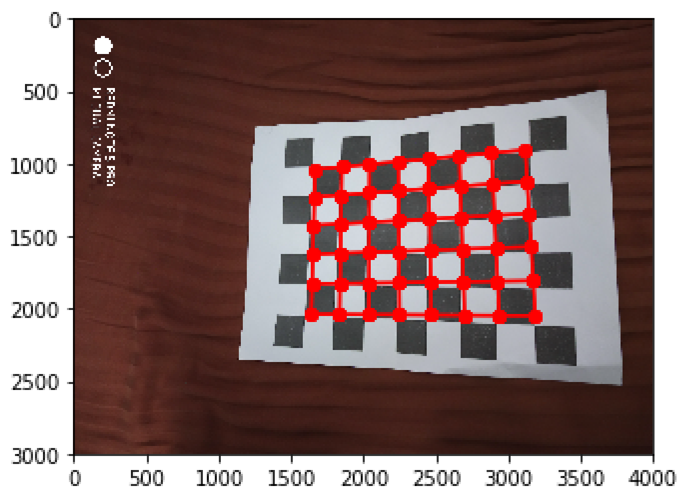
Calibration Matrix ->

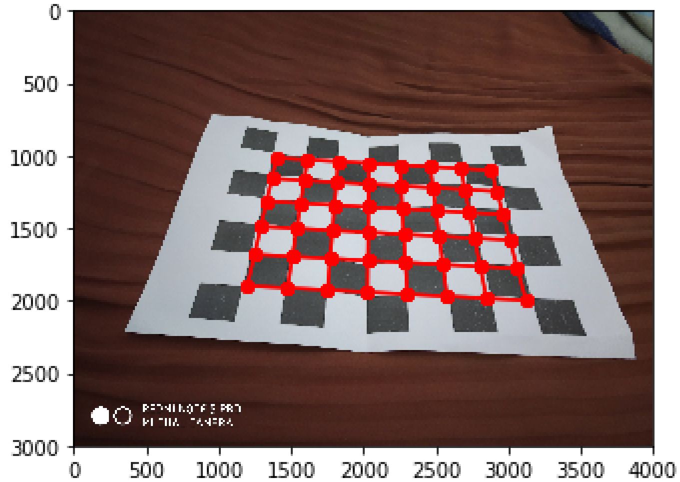
[[1759.23946515 -28.00857334 1419.96932585]
 [-0.   1878.65437893 1736.82215837]
 [-0.   0.   1.   ]]

2. We use 8 images of a checkerboard pattern to perform Zhang's Method of calibration.
    a. **Camera Calibration Matrix ->**

       [[3176.4793653 0. 1946.41907023]
       [0. 3206.8057093 1469.54116455]
       [0. 0. 1.]]






0
500
1000
1500
2000
2500
3000
3500
4000