CMSC733 - Homework 1
AutoCalib

Saurabh Palande
Masters in Robotics
University of Maryland, College Park
UID: 118133959
Email: spalande@umd.edu

***Abstract*—In this homework the Camera Calibration technique proposed by Zhang is implemented. 13 images of Checkerboard pattern (9x6 - excluding boundary) are used for this calibration. Using equations from Zhang's paper, intrinsic and extrinsic parameters are estimated and these parameters are optimized using a non-linear geometric error minimization. The results obtained from the above approach are analysed and re- projection error and final K matrix are found.**

I. Camera Calibration

The following steps are followed to calibrate the camera based on the Checkerboard images.

A. Checkerboard Corner Detection

The checkerboard pattern used for this camera calibration technique is 9x6 excluding the borders and is shown in Figure 1. OpenCV inbuilt function [cv2.findChessboardCorners] is used for finding the corners for checkerboard pattern. The output is shown in Figure 2.

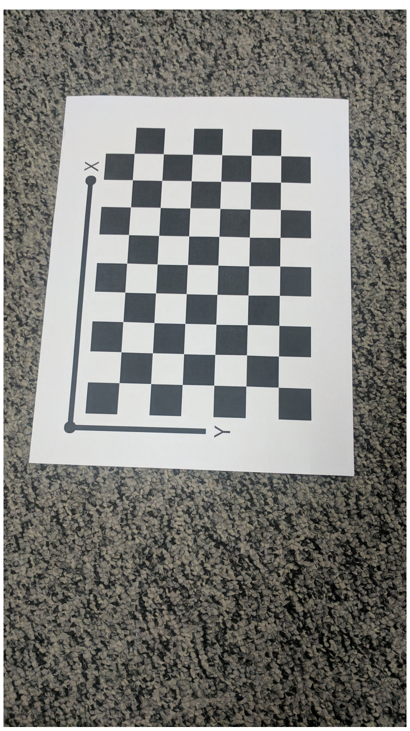

Fig. 1: Input image

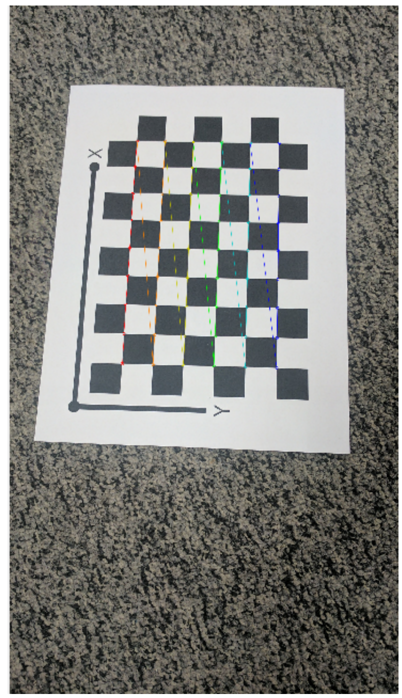

Fig. 2: Corners on input image

B. Initial Parameter Estimation

The first step is to get a good initial estimate of the parameters so that we can feed them into the non-linear optimizer. The following steps are followed to find the initial estimate.

1) Finding Homography matrices: The relationship between a 3D point M and its image projection m is given by $sm = A[Rt]M$ where s is an arbitrary scale factor, (R, t), called the extrinsic parameters, is the rotation and translation which relates the world coordinate system to the camera coordinate system, and A, called the camera intrinsic matrix, is given by

$$A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

A model point M and its image m is related by a homography H:

$sm = HM$ with $H = A[r1r2t]$

The homography matrices for all the images are calculated using cv2.findHomography.

2) Finding V: Let

$$B = A^{-T} A^{-1} \tag{1}$$

Where B is symmetric, defined by a 6D vector

$$b = \begin{bmatrix} B12 & B22 & B13 & B23 & B33 \end{bmatrix}^T \tag{2}$$

Now,

$$h^T{}_i B h_j = v^T{}_{ij} b \tag{3}$$

$v_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T$

$$\begin{bmatrix} v_1 2^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \tag{4}$$

If n images of the model plane are observed, by stacking n such equations, we have

$$Vb = 0 \tag{5}$$

where V is a 2n × 6 matrix.

3) Finding Camera Intrinsic Matrix: The solution to (3) is well known as the eigenvector of $V^T V$ associated with the smallest eigenvalue (equivalently, the right singular vector of V associated with the smallest singular value). Once b is estimated, we can compute all camera intrinsic matrix as follows:

$$v_0 = (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - {B_{12}}^2) \tag{6}$$

$$\lambda = B_{33} - [B^2{}_{13} + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \tag{7}$$

$$\alpha = \sqrt{\frac{\lambda}{B_{11}}} \tag{8}$$

$$\beta = \sqrt{\frac{\lambda B_{11}}{B_{11}B_{22} - B^2{}_{12}}} \tag{9}$$

$$\gamma = -B_{12}\alpha^2\beta/\lambda \tag{10}$$

$$u_0 = \gamma v_0/\beta - B_{13}\alpha^2/\lambda. \tag{11}$$

The initial estimate of the intrinsic matrix is shown in Figure 3.

```
********Initial Intrinsic matrix**********
array([[ 2.05278817e+03, -3.69529711e-01,  7.63061089e+02],
       [ 0.00000000e+00,  2.03663415e+03,  1.35261456e+03],
       [ 0.00000000e+00,  0.00000000e+00,  1.00000000e+00]])
The error before optimization is  0.7646486123322417
```

Fig. 3: Initial estimate

4) Finding Extrinsic Matrix: Once A is known, the extrinsic parameters for each image are readily computed suing the following equations.

$$r_1 = \lambda A^{-1} h_1 \tag{12}$$

$$r_1 = \lambda A^{-1} h_2 \tag{13}$$

$$r_3 = r_1 r_2 \tag{14}$$

$$t = \lambda A^{-1} h_3 \tag{15}$$

$$\lambda = 1/||A^{-1} h_1|| \tag{16}$$

C. Optimization and Computing K_c

The next step after computing the extrinsic and intrinsic parameters is to do optimization. Intially, value of radial distortion is taken as (0,0). To minimize the reprojection error, scipy.optimize is used.

D. Final output

After optimization,The final camera matrix and radial distortion coefficients are shown in Figure 4:

```
********Final Intrinsic matrix**********
array([[1.25569875e+03, 2.58721178e+03, 8.02609814e+02],
       [0.00000000e+00, 6.92868362e+02, 1.34993802e+03],
       [0.00000000e+00, 0.00000000e+00, 1.00000000e+00]])
k1 0.00048616359264881236
k2 -9.234354607508772e-05
The error after optimization is  0.7365468812059792
```

Fig. 4: Final Matrix

Using the parameters obtained above,the images are rectified by finding the correct homogrpahy and then the rectified corner points are reprojected on the rectified image.The output after rectification is shown below in the Apendix.

References

[1] https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr98-71.pdf

[2] https://cmsc733.github.io/2022/hw/hw1/

II. Apendix

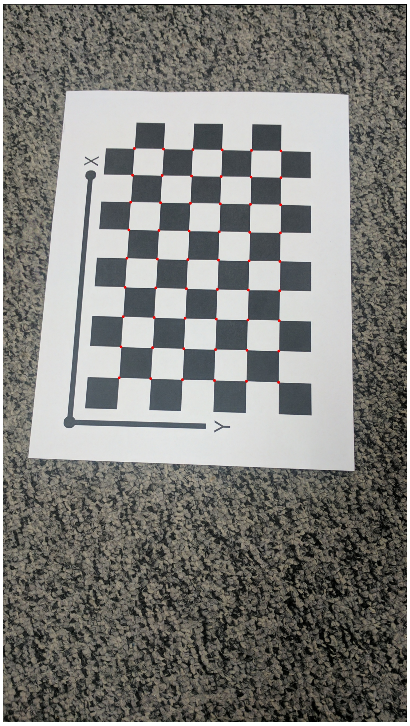

Fig. 5: Rectified Image

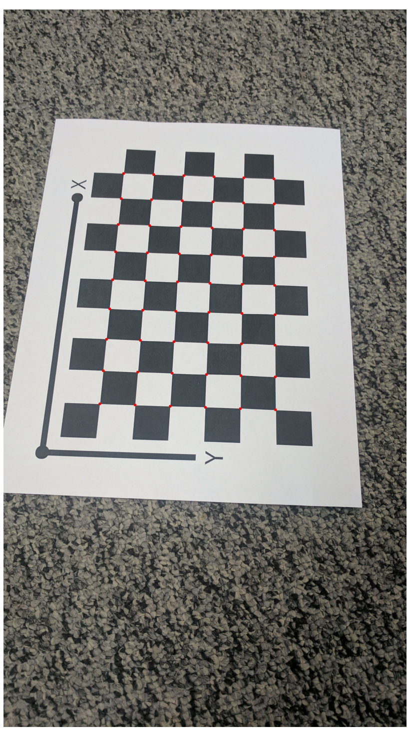

Fig. 6: Rectified Image

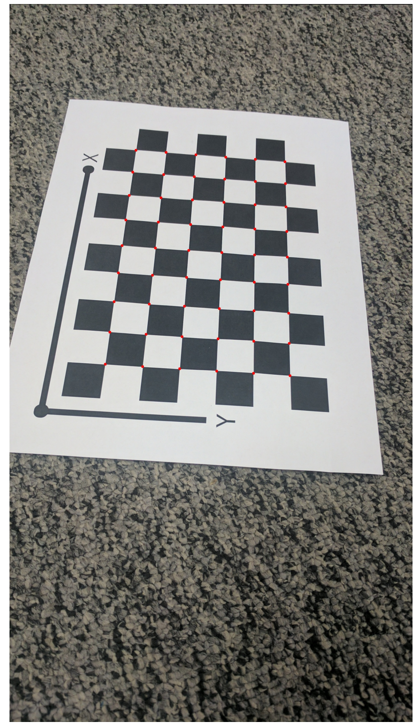

Fig. 7: Rectified Image

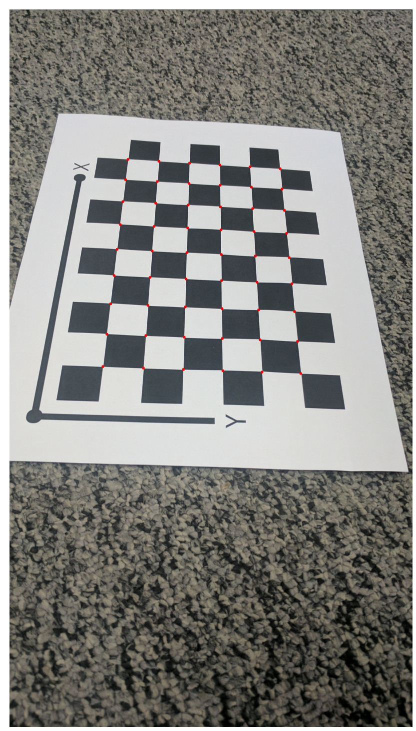

Fig. 8: Rectified Image

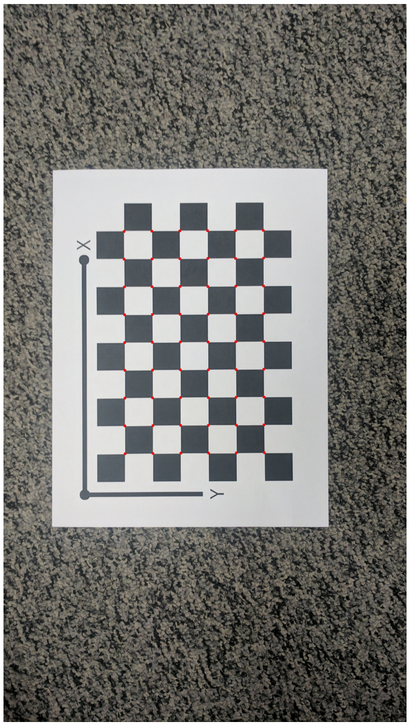

Fig. 9: Rectified Image

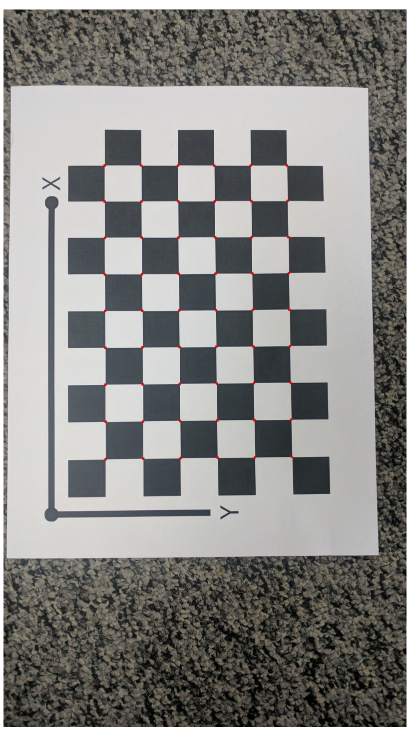

Fig. 10: Rectified Image

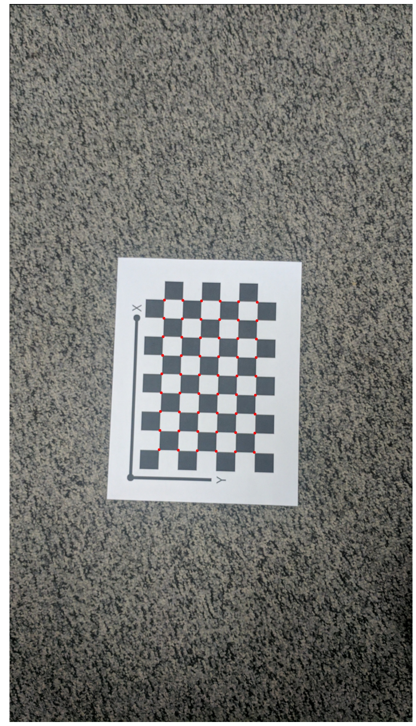

Fig. 11: Rectified Image

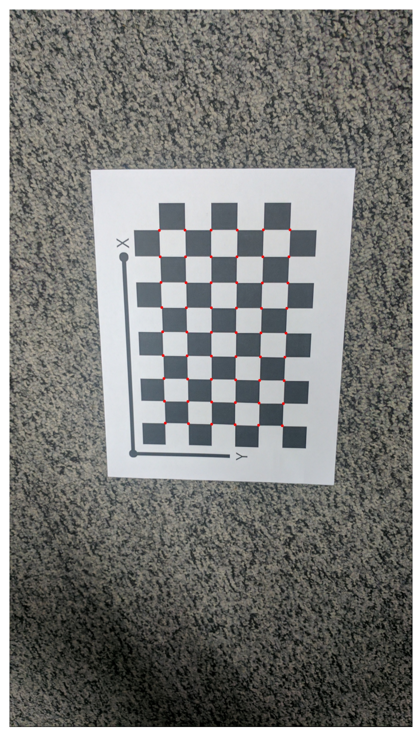

Fig. 12: Rectified Image

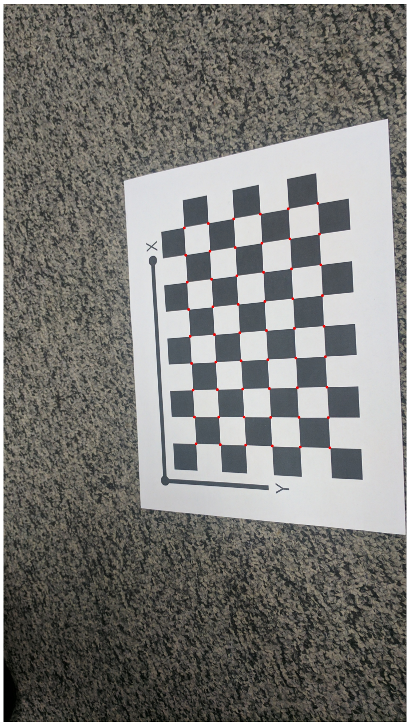

Fig. 13: Rectified Image

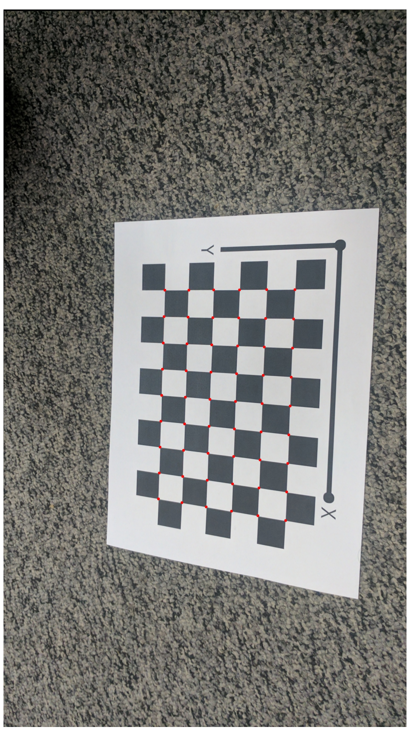

Fig. 14: Rectified Image

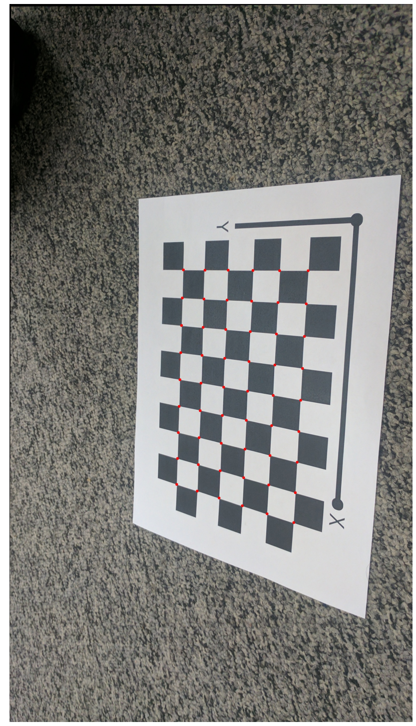

Fig. 15: Rectified Image

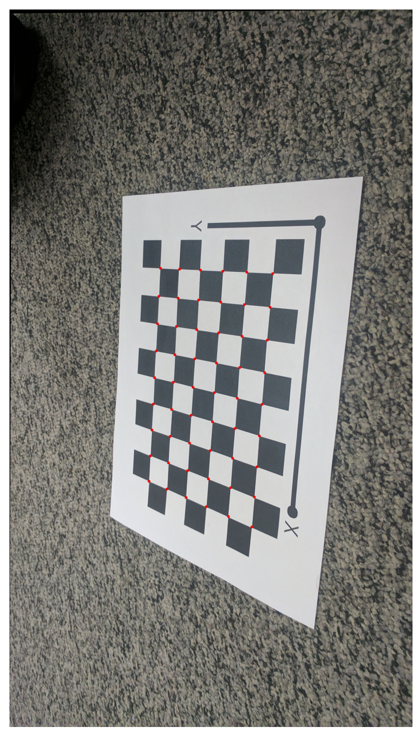

Fig. 16: Rectified Image

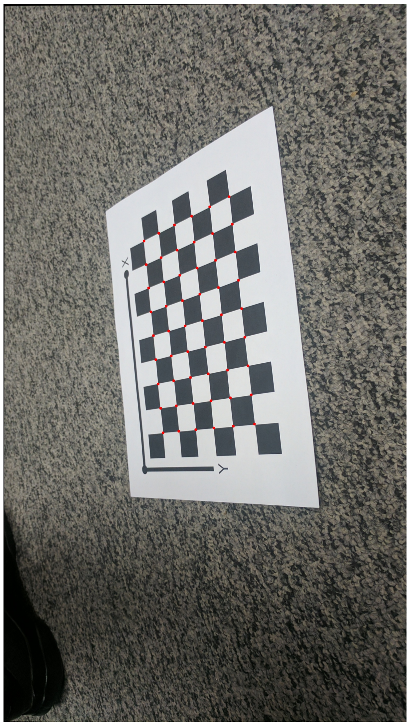

Fig. 17: Rectified Image