Homework 1: AutoCalibration

Abhinav Modi
Masters of Engineering in Robotics
University of Maryland, College Park
Email: abhi1625@umd.edu *Using 2 Late day*

I. Introduction

Camera Calibration is one of the most time consuming but significant process for any computer vision research involving 3D geometry, of which robotics research is a very prime example. An automatic way to perform efficient and robust camera calibration was provided by Zhengyou Zhang of Microsoft**?**. In this report is summarized the implementation of Zhang's technique for calibration.

II. Data

Zhang's implementation requires a calibration target to estimate the camera intrinsics. For this project 13 image of a checkerboard were provided and are present in the data folder.

III. The Calibration Approach

The first step is to find a good initial estimate of the Intrinsic parameters and the distortion coefficients. This is done in the following steps:

1) Find corner points in the checkerboard for all the givenn images.
2) Compute the homographies between the world coordinate points(the checkerboard corners in the original board of which the images are taken), and the corners found in each image.
3) Compute the Intrinsic parameters by solving for $\mathbf{B} = A^T A$, where A is the intrinsic camera matrix. The initial intrinsic camera matrix computed for this project is

$$\begin{bmatrix} 2034.7497, & 0, & 772.7044 \\ 0.0000, & 2017.9031, & 1360.9095 \\ 0.0000, & 0.0000, & 1.0000 \end{bmatrix} \quad (1)$$

4) This intrinsic matrix is used to estimate the extrinsic matrices R and t for each image.
5) All the above steps are performed assuming there is no distortion of any sorts in the image. Now to calculate the distortion coefficients we use K_c = [0,0] as an initial guess.
6) Once we have an initial guess for each parameter A, R, t, K_c we perform Non-linear Geometric Error Minimization to refine them and obtain a better estimate of the camera intrinsic parameters and the distortion coefficients. This was done using $scipy.optimize.least_squares$ function of the scipy library. The following values of the intrinsic camera matrix and distortioin coefficients were obtained:

$$A = \begin{bmatrix} 2063.3798, & 0.0000, & 763.1675 \\ 0.0000, & 2046.5943, & 1381.0201 \\ 0.0000, & 0.0000, & 1 \end{bmatrix} \quad (2)$$

$$K_c = \begin{bmatrix} 0.0581, & -0.2572 \end{bmatrix} \quad (3)$$

IV. Discussion and Conclusion

Once the non-linear minimization is performed the Intrinsic camera matrix obtained is much more refined. Geometrically speaking this optimization is based on minimizing the euclidean distance between the corner point in the image, and the corner point obtained after reprojecting the world coordinate point, using the intrinsic camera matrix A, extrinsic camera matrix R, t and the distortion coefficients K_c. After optimization there is still some error left because this optimization gives a local minima. This is why a good initial guess is necessary. To evaluate the accuracy of the calibration matrices obtained we compute the **reprojection error**. This is done by calculating the absolute norm between what we got with our transformation and the corner finding algorithm. To find the average error we calculate the arithmetical mean of the errors calculate for all the calibration images. This value comes out to be **1.7436**.

As can be seen in the following image, we can clearly see the difference in the original point and the point once the undistortion has been performed. The blue circle was the original point and the pink circle os the point once the calibration has been perofrmed. These calibration matrices are then used to rectify the images. The output rectified images are shown below:

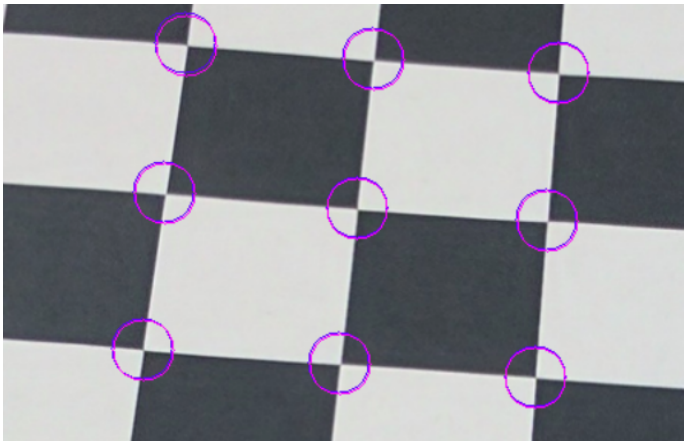

Figure 1: Difference in the rectified and the original image corners

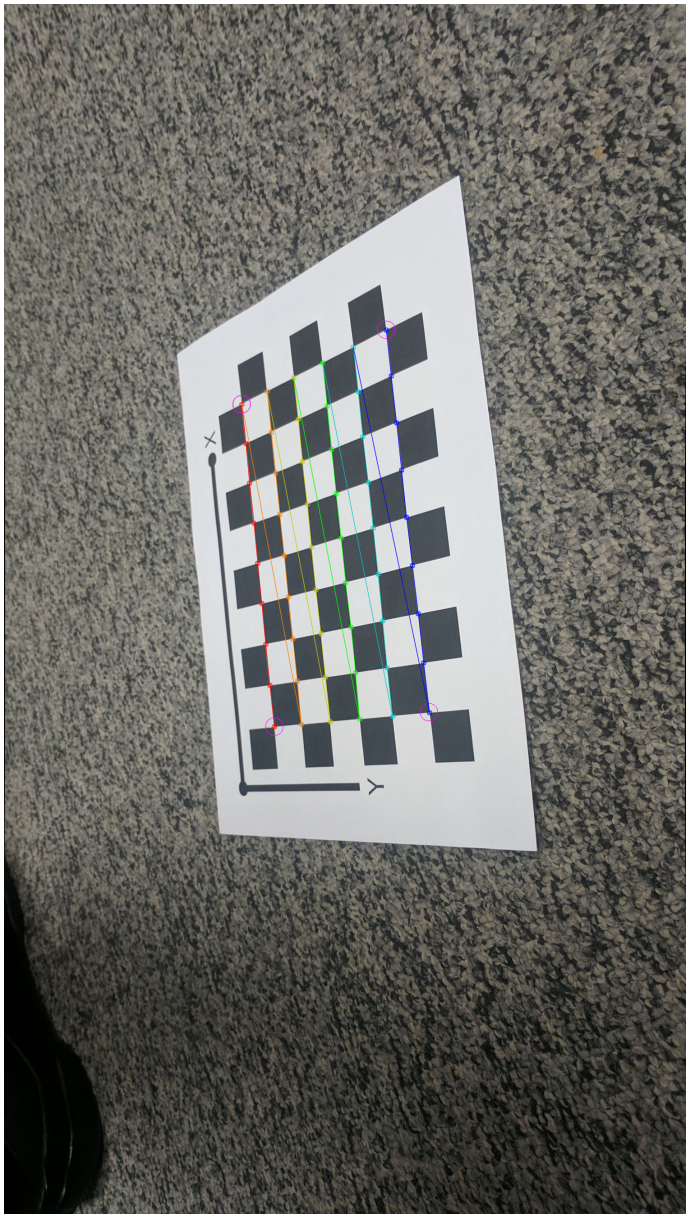

Figure 2: Rectified output of image 1

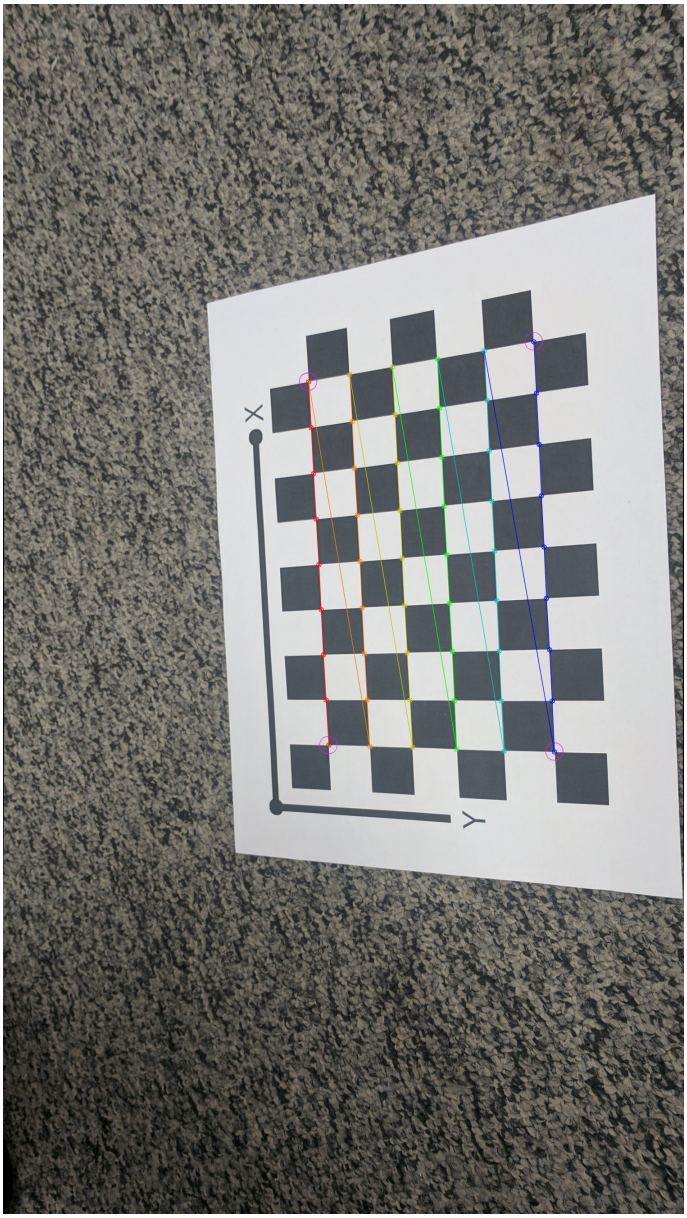

Figure 3: Rectified output of image 2

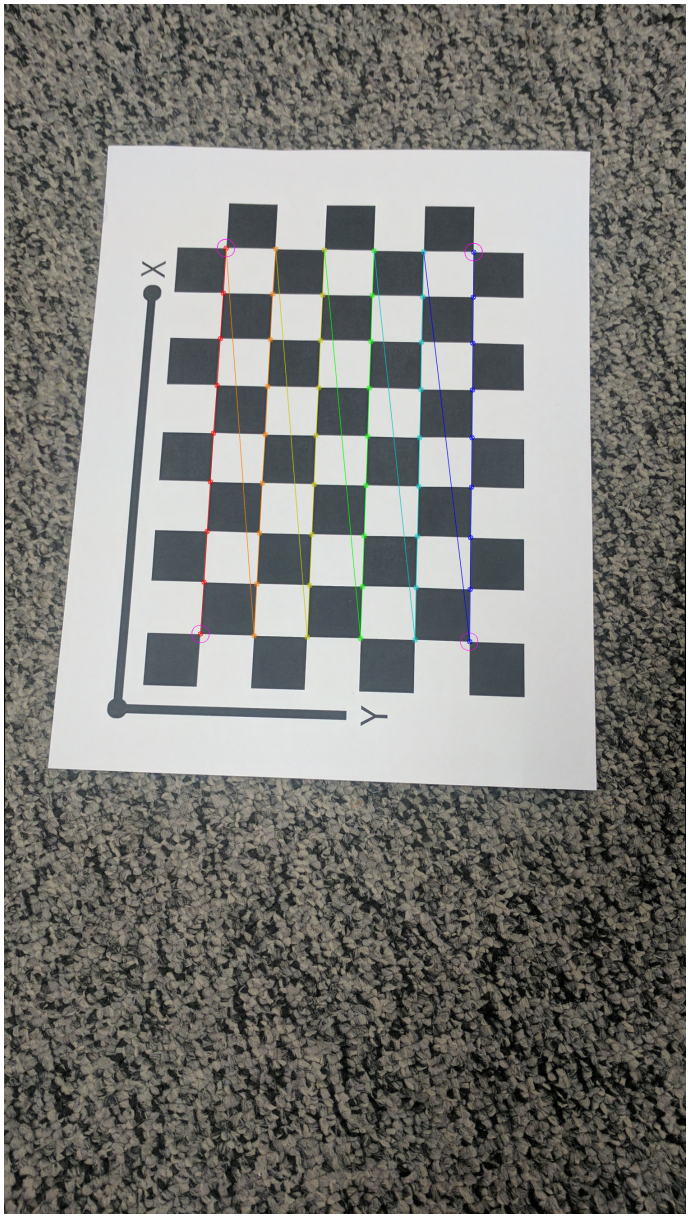

Figure 4: Rectified output of image 3

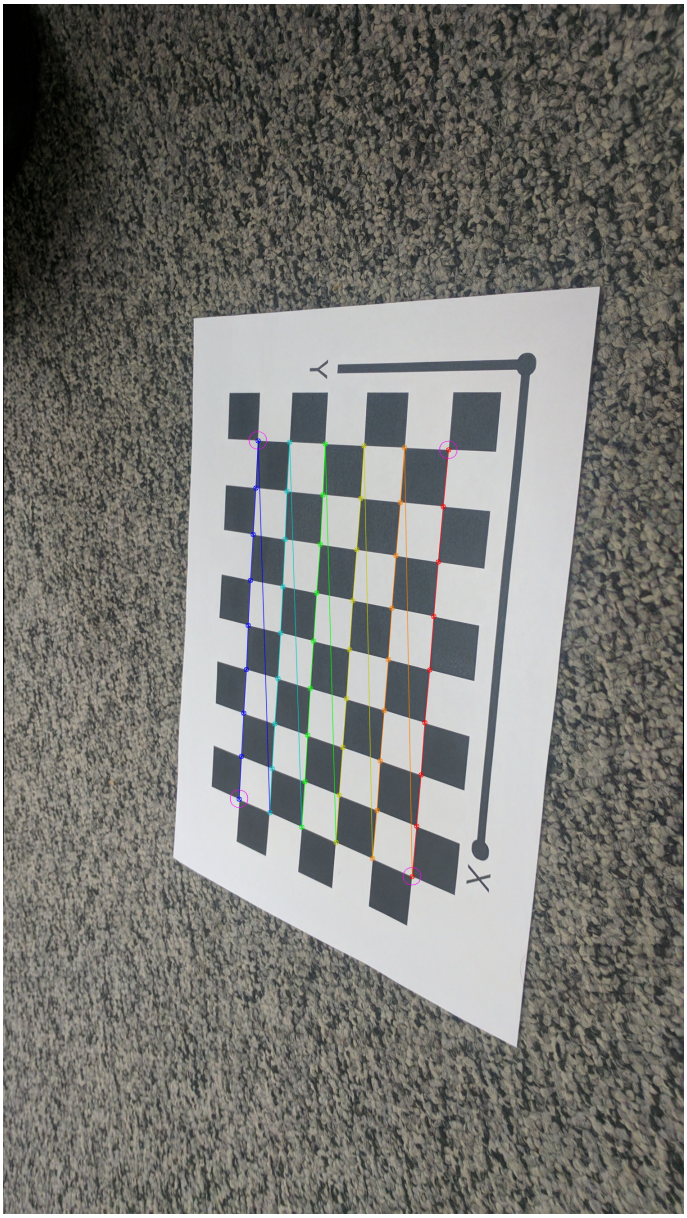

Figure 5: Rectified output of image 4

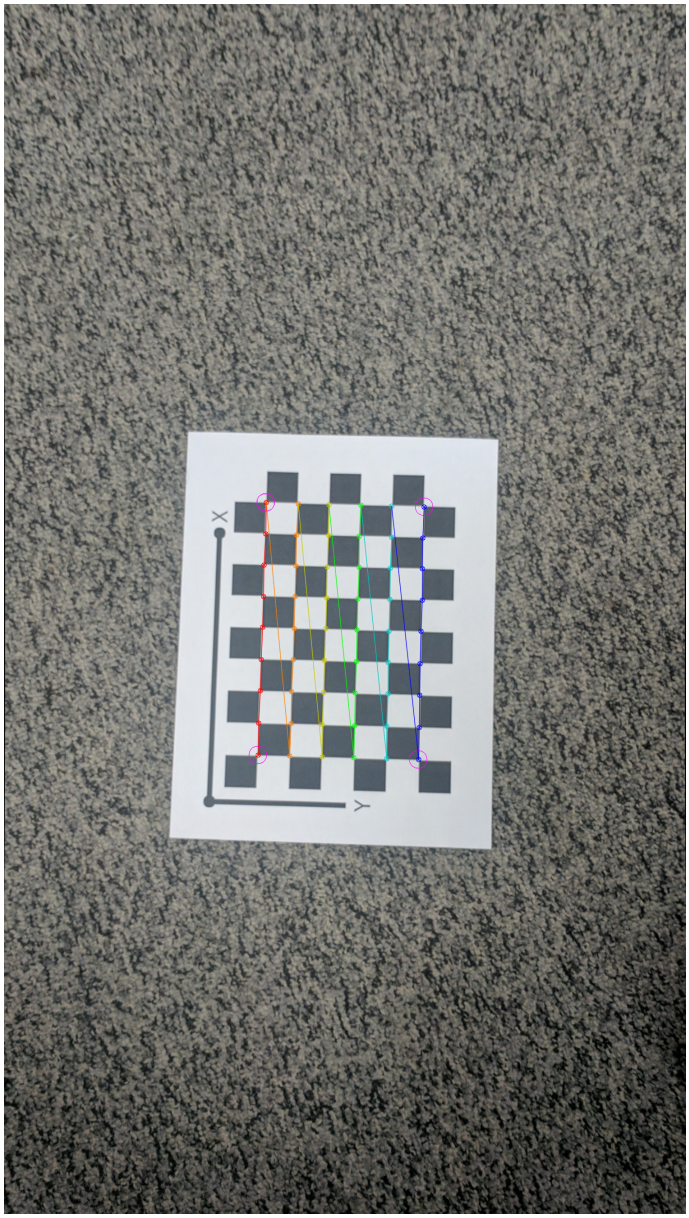

Figure 6: Rectified output of image 5

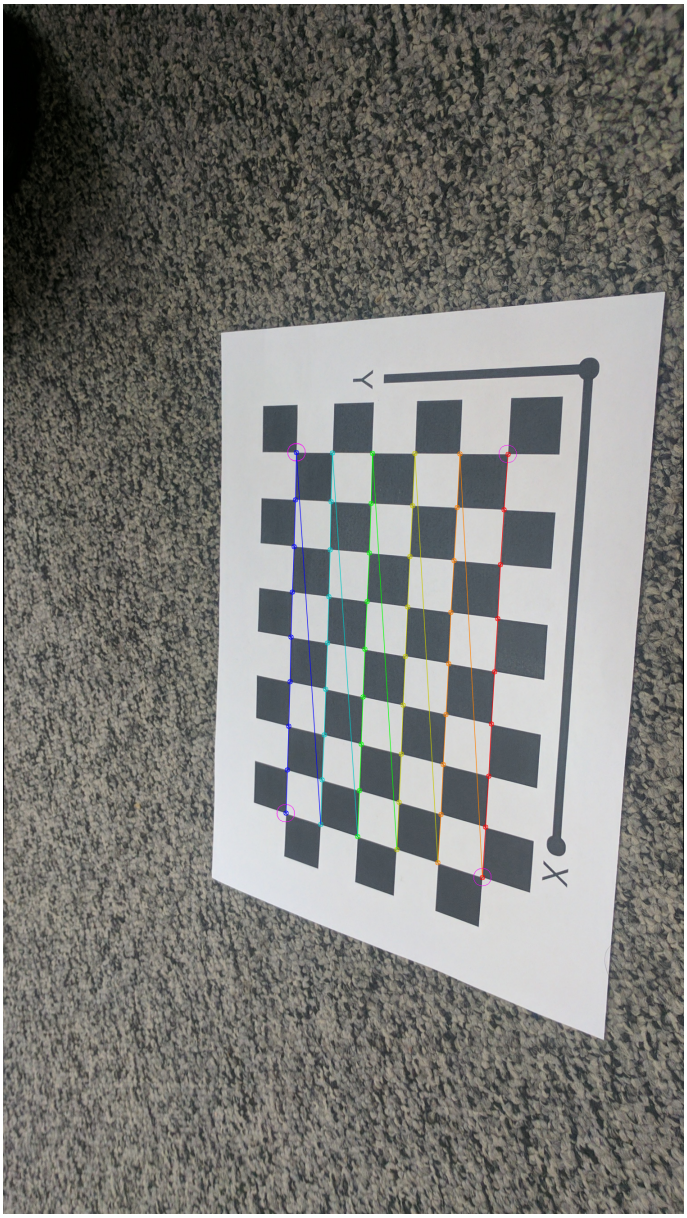

Figure 7: Rectified output of image 6

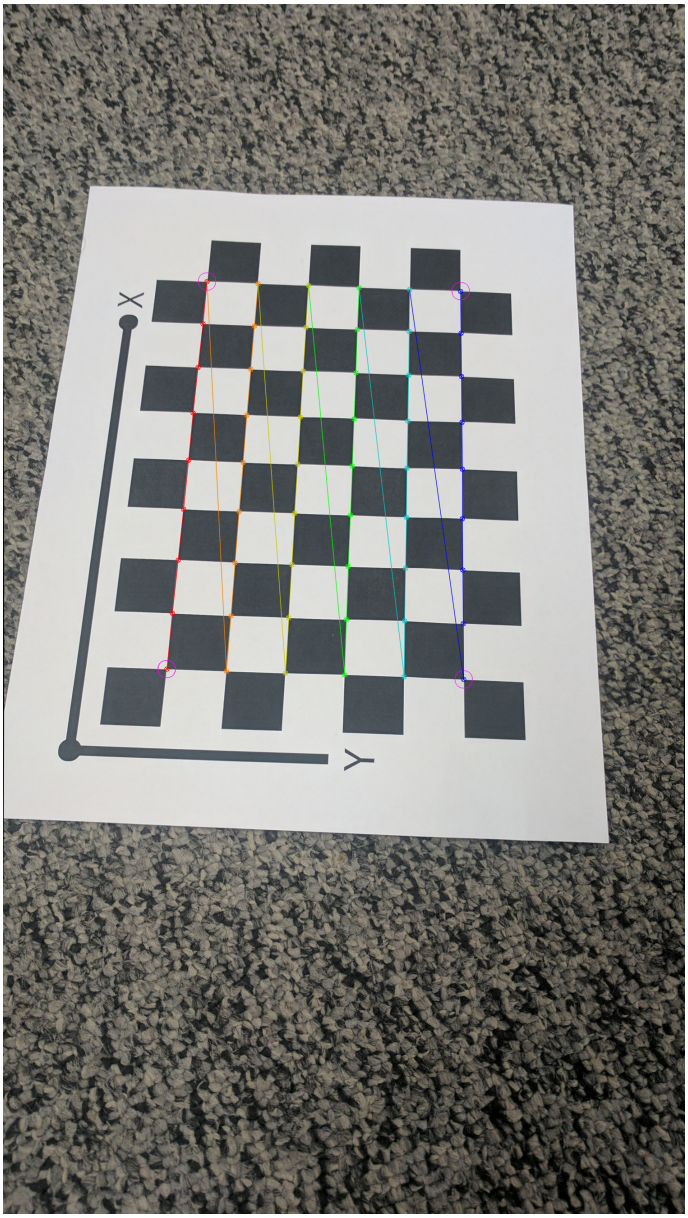

Figure 8: Rectified output of image 7

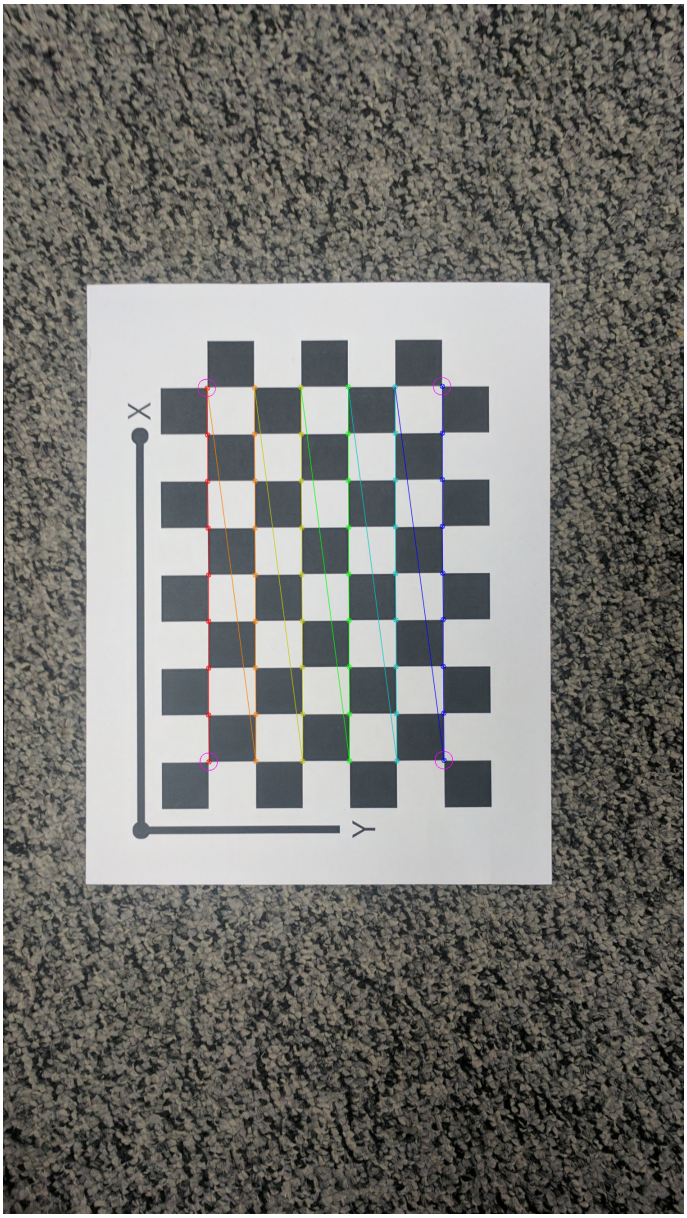

Figure 9: Rectified output of image 8

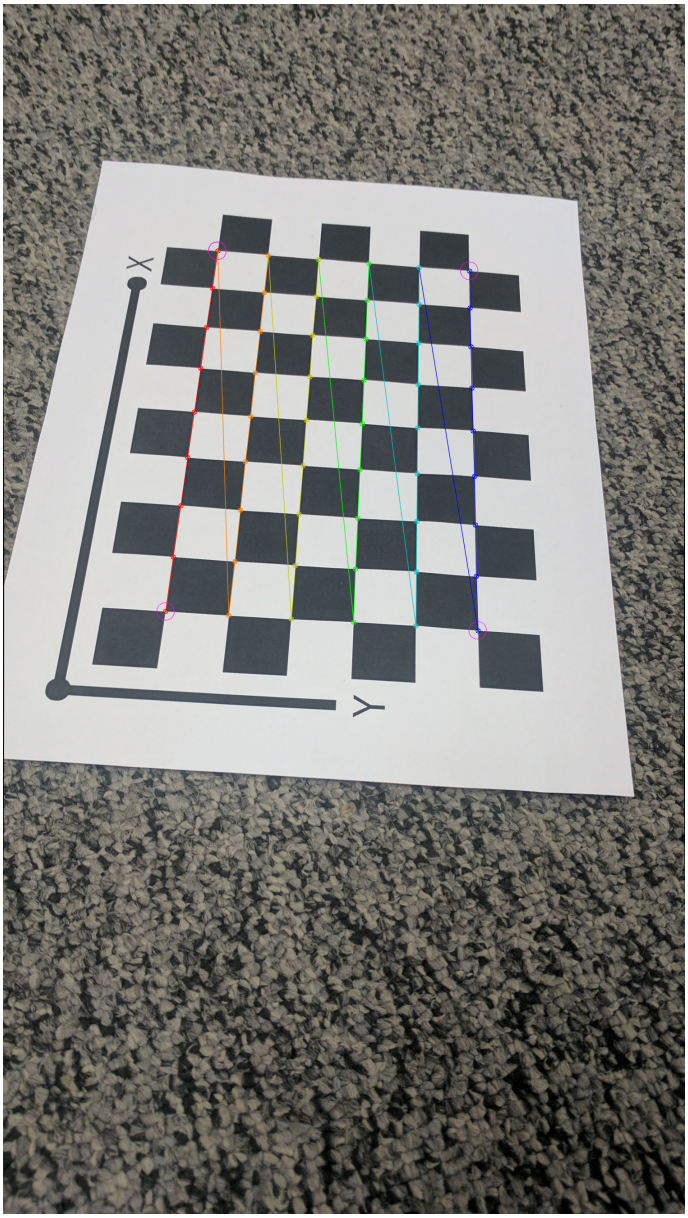

Figure 10: Rectified output of image 9

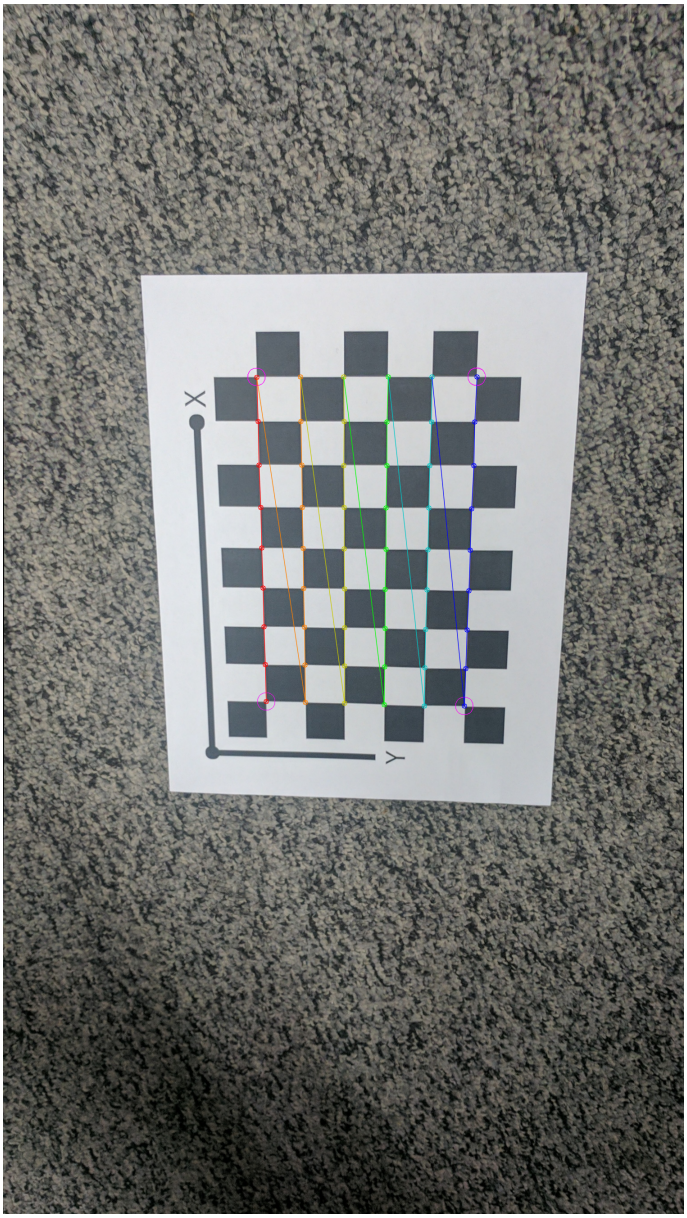

Figure 11: Rectified output of image 10

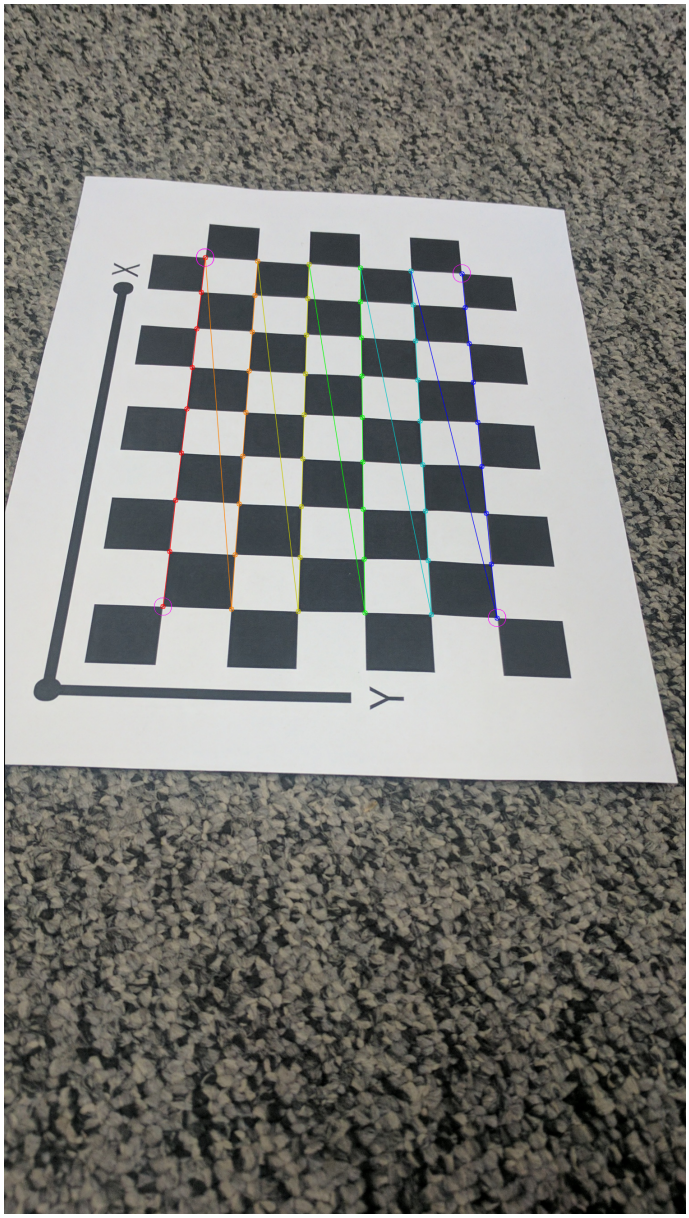

Figure 12: Rectified output of image 11

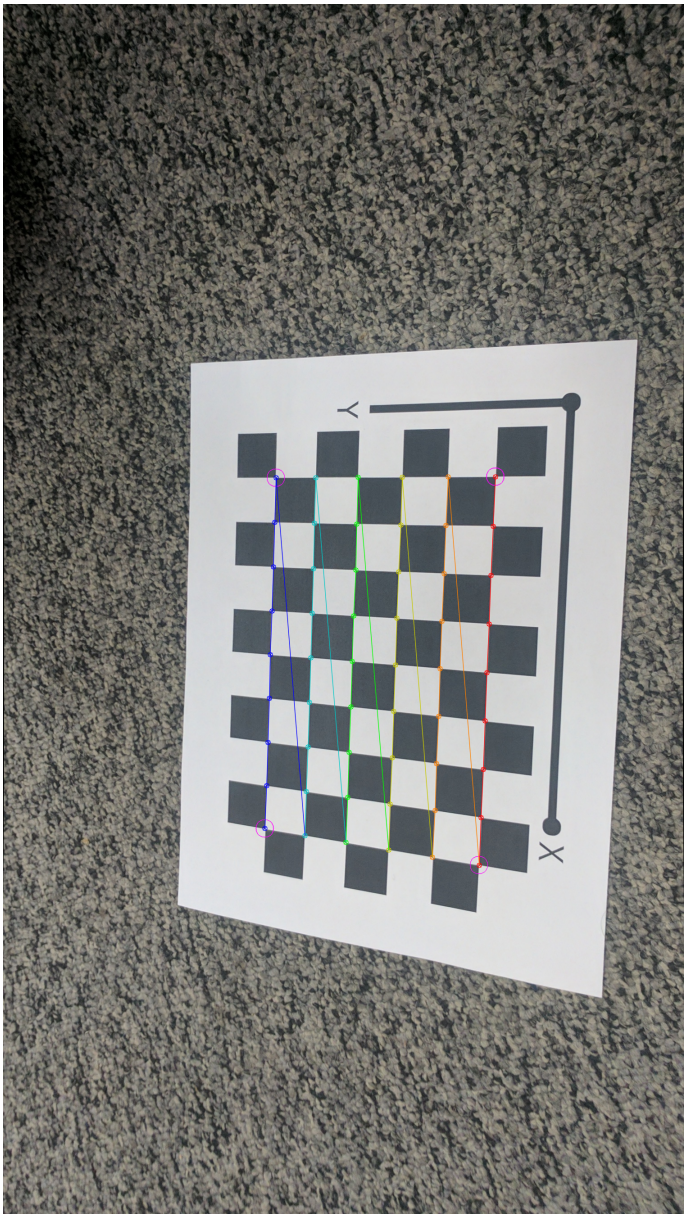

Figure 13: Rectified output of image 12

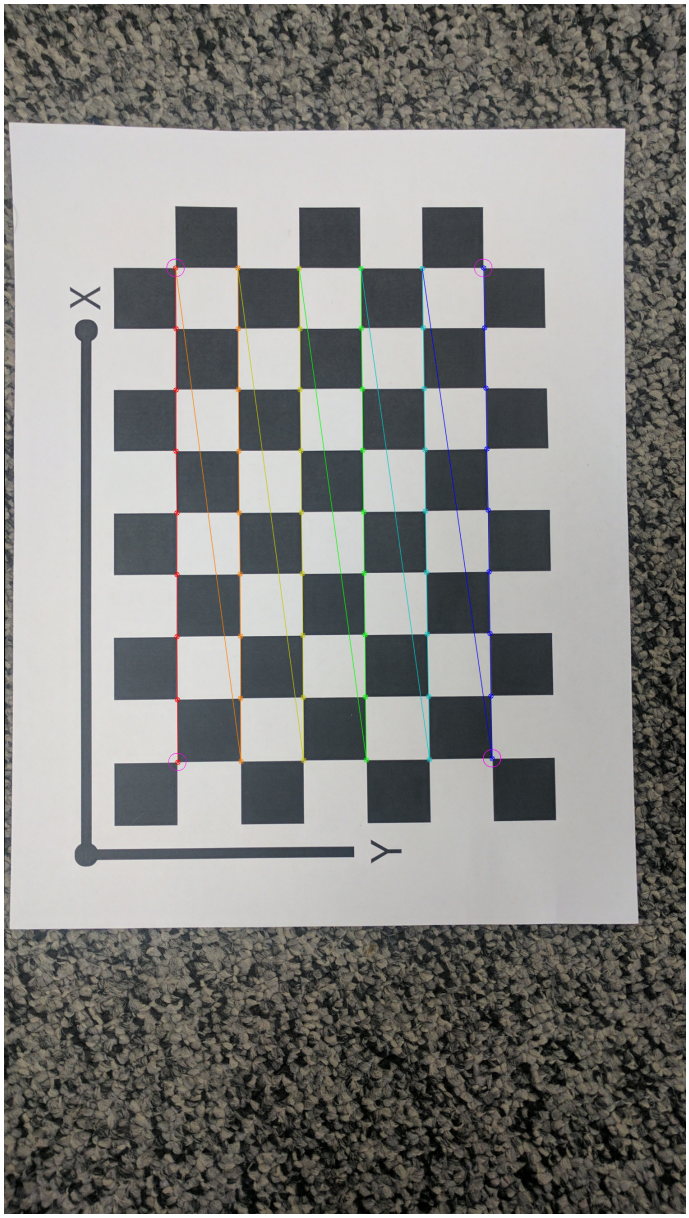

Figure 14: Rectified output of image 13