Homework 1
Autocalib

Abhijit Mahalle
University of Maryland
College Park, MD

***Abstract*—The report describes in brief the implementation of camera calibration method as described by Zhengyou Zhang in his paper "A Flexible New Technique for Camera Calibration" [1].**

I. Introduction

Camera calibration involves estimating focal length, distortion coefficients, and principle points for a given camera. These are called intrinsic parameters and an intrinsic matrix(A) is given by:

$$.\begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_o \\ 0 & 0 & 1 \end{pmatrix}$$

Where α and β are the scale factors in image u and v axes, and γ is the parameter describing skewness of the two image axes [1]. The principal points is given by (u_0, v_0). However, due to the non-ideal nature of the camera lens, there can be distortion in the image. We will be assuming that we have radial distortion. It is symmetric in nature and ideal image points are distorted along radial directions from the distortion center [2]. The distortion parameters are given by k_1 and k_2. We will try to estimate the radial distortion parameters as well.

II. Data

As will be discussed further, we first need to compute a homography matrix H. So we need a minimum of four points per image to compute the homography. Each H matrix will result in two equations. The matrix B can have a maximum of 6 DoF. Hence, we need at least three images to estimate K.

A checkerboard pattern has been used to estimate camera intrinsic parameters. The pattern was printed on an A4 paper and the size of each square was 21.5mm. Thirteen images taken from a Google Pixel XL phone with focus locked were used to calibrate the camera

A. Image points

The OpenCV function *findChessboardCorners* is used to detect the checkerboard corners. The obtained corners are row by row, left to right in every row [3]. To increase the detection accuracy, the OpenCV function *cornerSubPix* is used. It helps in getting the sub-pixel accurate location of corners [4].

B. World points

For each corner detected in the previous section, we need a corresponding world point. Keeping in mind the sequence of the image points, a mesh grid was obtained depending on the number of rows and columns in the checkerboard and was multiplied by the square size.

III. Initial Parameter

A. Estimating K

As mentioned in [1], if $Z = 0$ for the world points, then we can write $sm = HM$, where M are the world points and m are the image points. Since $Z = 0$, M is a 3×1 matrix representing homogeneous world co-ordinates. Therefore, H is a 3×3 matrix. H can be further written as $H = A[r1, r2, t]$ where A is the camera intrinsic matrix, r1 and r2 are rotational vectors and t is transnational vector.

1) Estimating H: A homography matrix was computed for all the images between the world points and their corresponding image points. After this step, we will get a total of thirteen H matrices.

B. Estimating B

As mentioned in [1], B matrix is given by $B = A^{-T}A^{-1}$. Since A is an upper triangular matrix, we get B as a symmetric matrix. Therefore, we need to estimate $.\begin{pmatrix} B_{11} & B_{12} & B_{22} & B_{13} & B_{23} & B_{33} \end{pmatrix}$ Let's call it b. After following the steps mentioned in [1], the estimated b is:

$$.\begin{pmatrix} 1.50143137e-07 \\ 8.28746405e-11 \\ 1.52412974e-07 \\ -1.14436180e-04 \\ -2.05856619e-04 \\ 9.99999972e-01 \end{pmatrix}$$

C. Estimating A

After computing b, matrix A can be found as explained in [1]. The computed A matrix is:

$$.\begin{pmatrix} 2.05637851e+03 & -1.12657749e+00 & 7.61435270e+02 \\ 0.00000000e+00 & 2.04100889e+03 & 1.35023620e+03 \\ 0.00000000e+00 & 0.00000000e+00 & 1.00000000e+00 \end{pmatrix}$$

D. Estimating Camera Extrinsic

Once A is known, the extrinsic parameters for each image is readily computed [1].

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

$$r_2 = \lambda A^{-1} h_2$$

Since r_1, r_2 and r_3 are orthogonal, we can write

$$r_3 = r_1 \times r_2$$

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

Where h_1, h_2 and h_3 are column vectors of computed H. Since r_1 and r_2 are unit vectors, λ is given by $\lambda = \frac{1}{\|A^{-1}h_1\|} = \frac{1}{\|A^{-1}h_2\|}$ There will be a $[r_1, r_2, r_3, t]$ for all the thirteen images.

E. Approximate Distortion

The radial distortion parameters, k_1 and k_2 are both initialized as 0.

IV. Non-linear Optimization

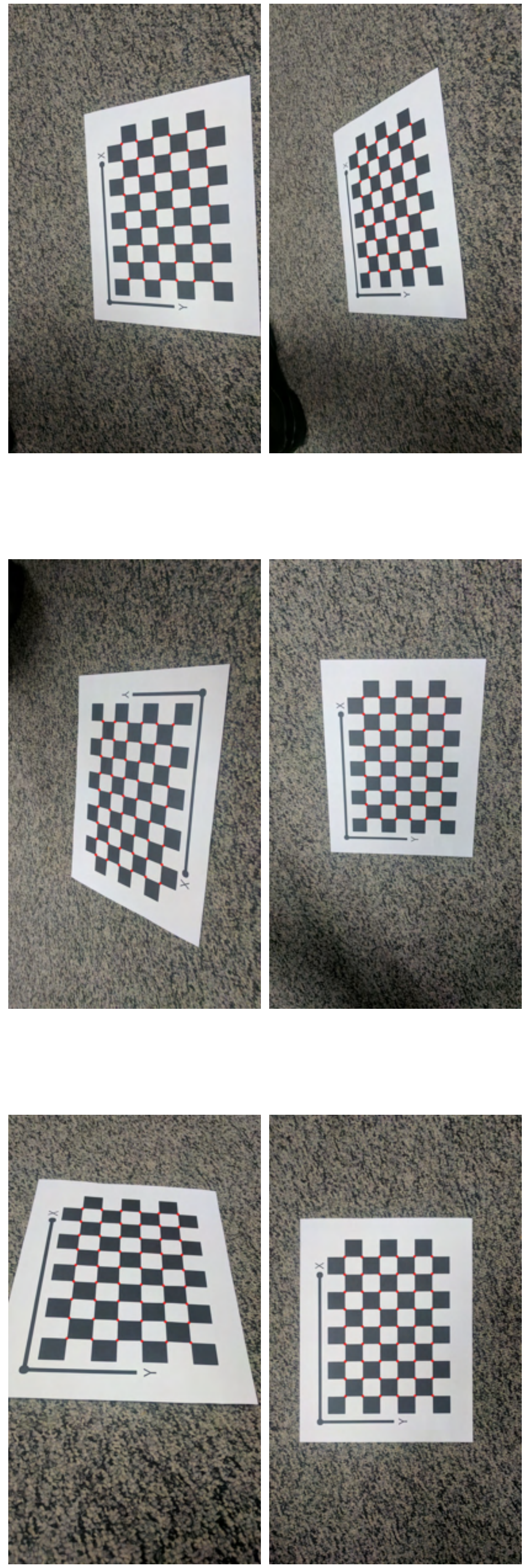
Once we get the initial estimates for the parameters, we try to minimize the re-projection error given by $\|m_{ij} - \tilde{m}(A, k_1, k_2, R_i, t_i, M_j)\|^2$ for all the points on all images. here, i is the image number and j is the point number. m_{ij} is the j^{th} observed point on image i. M_j is the j^{th} world point. $\tilde{m}$ is the reprojected point. *scipy.optimize.least_squares* function has been used to minimize the re-projection error. The loss function is written as per [2].

V. Results

before optimization, the reprojection error was 0.6976324290906062 and it reduced to 0.6818732170461967 after optimization. The camera matrix obtained is:

$$.\begin{pmatrix} 2.05637515e+03 & -1.12802412e+00 & 7.61440334e+02 \\ 0.00000000e+00 & 2.04100059e+03 & 1.35024940e+03 \\ 0.00000000e+00 & 0.00000000e+00 & 1.00000000e+00 \end{pmatrix}$$

Refer fig.1 for reprojection of corners on rectified images.



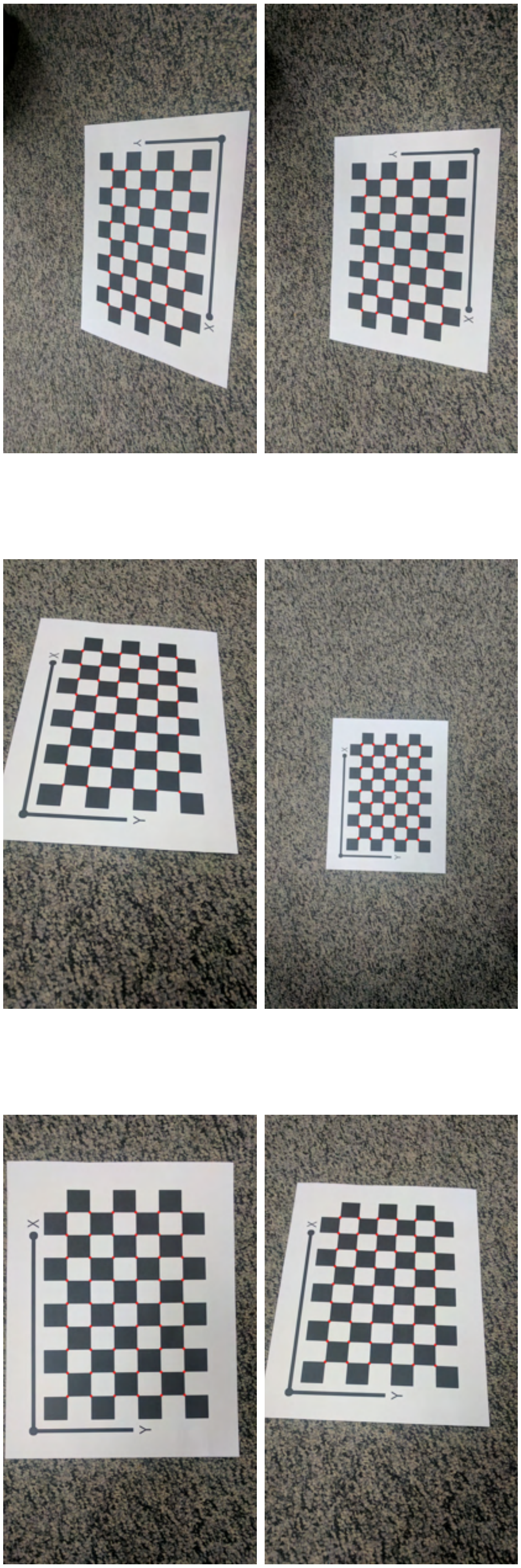
References

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

[2] https://www.cs.rutgers.edu/ẽlgammal/classes/cs534/lectures/CameraCalibration-book-chapter.pdf

[3] https://docs.opencv.org/master/d9/d0c/group__calib3d.html#ga93efa9b0aa890de240ca32b11253dd4a

[4] https://docs.opencv.org/master/dd/d1a/group__imgproc__feature.html#ga354e0d7c86d0d9da75de9b9701a9a87e



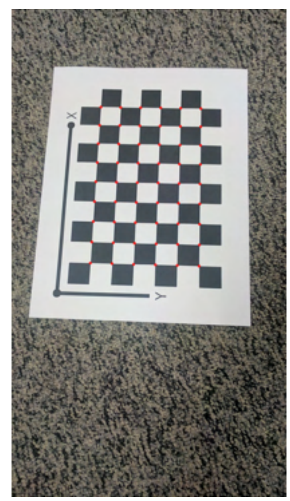


Fig. 1. Reprojection of corners on rectified images