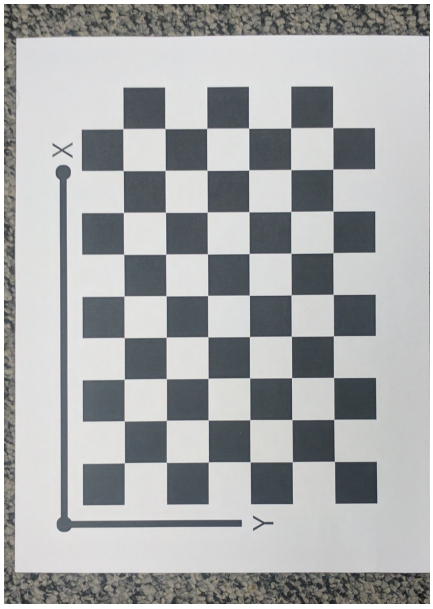
Homework 1: AutoCalib

Srinidhi Sreenath
Robotics Graduate Student
University of Maryland College Park
Email: ssreenat@terpmail.umd.edu

***Abstract*—Estimating the camera intrinsic, extrinsic parameters and distortion coefficients using Zhengyou Zhang's method [1]. The calibration target image is a checkerboard pattern with each square size of 21.5 mm. First, the camera intrinsic matrix (K) is approximated and using that, the extrinsic parameters Rotation (R) and translation (t) are estimated. Then, using these as initial estimates, non linear optimization is done to minimize the geometric error and refine the intrinsic matrix parameters and the distortion coefficients.**

I. Initial Parameter Estimation

A set of 11 checkerboard images are provided with various orientations. A sample image is shown below.



The first step is to find the pixel coordinates of the chessboard corners in each image. Points are found using the *cv2.findChessboardCorners* function. The patternsize parameter is (9,6) which is the number of inner corners that are to be detected. A total of 54 corner points are found for each image.

Next, I specified a 3D coordinate system for each of the 54 detected points. In the 3D coordinate system the Z value is 0 for the plane of the chessboard. X and Y coordinates vary by the size of the chessboard square which is given to be 21.5 mm. The 3D coordinate system is as follows:

$$[0,0,0],[21.5,0,0],...[172,0,0],$$
$$[0,21.5,0],[21.5,21.5,0],...[172,21.5,0],$$
$$.$$
$$.$$
$$.$$
$$[0,107.5,0],[21.5,107.5,0],...[172,107.5,0]$$

Now, the correspondence between the 3D coordinates and the 2D pixel coordinates are available. Next is to find the homography matrix between the 3D and 2D corresponding points. This is done with the method of direct linear transform.

Given n corresponding points, each corresponding pair can be used to create a 2x9 matrix as represented below:

$$p_i = \begin{bmatrix} -x_i & -y_i & -1 & 0 & 0 & 0 & x_i x_i' & y_i x_i' & x_i' \\ 0 & 0 & 0 & -x_i & -y_i & -1 & x_i y_i' & y_i y_i' & y_i' \end{bmatrix}$$

Multiple such matrices are stacked to create a matrix P. Then the system $PH = 0$ needs to be solved for H. The system is shown below:

$$PH = \begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1 x_1' & y_1 x_1' & x_1' \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1 y_1' & y_1 y_1' & y_1' \\ -x_2 & -y_2 & -1 & 0 & 0 & 0 & x_2 x_2' & y_2 x_2' & x_2' \\ 0 & 0 & 0 & -x_2 & -y_2 & -1 & x_2 y_2' & y_2 y_2' & y_2' \\ -x_3 & -y_3 & -1 & 0 & 0 & 0 & x_3 x_3' & y_3 x_3' & x_3' \\ 0 & 0 & 0 & -x_3 & -y_3 & -1 & x_3 y_3' & y_3 y_3' & y_3' \\ -x_4 & -y_4 & -1 & 0 & 0 & 0 & x_4 x_4' & y_4 x_4' & x_4' \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & x_4 y_4' & y_4 y_4' & y_4' \end{bmatrix} \begin{bmatrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \\ h6 \\ h7 \\ h8 \\ h9 \end{bmatrix} = 0$$

Singular value decomposition (SVD) of P is done i.e $P = USV^T$. The last singular vector of V as the solution to H.

Once the Homography matrix is obtained, the matrix V which is in the closed form solution is updated as follows:

$$\mathbf{v}_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, \\ h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T .$$

where ith column of the homography matrix is $h_i = [h_{i1}, h_{i2}, h_{i3}]^T$

The system can be represented as 2 homogeneous equations in b as follows:

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0} .$$

For n images, stacking n such equations, we get the system $Vb = 0$. This system is solved for b by taking SVD of V matrix and obtaining the right singular vector associated with smallest single value. Once b is obtained, it is essentially the vector

$$b = [B11, B12, B22, B13, B23, B33]^T$$

This is used to solve for the intrinsic parameters of the camera matrix using the below equations:

$$\begin{aligned} v_0 &= (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \\ \lambda &= B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \\ \alpha &= \sqrt{\lambda/B_{11}} \\ \beta &= \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)} \\ \gamma &= -B_{12}\alpha^2\beta/\lambda \\ u_0 &= \gamma v_0/\beta - B_{13}\alpha^2/\lambda . \end{aligned}$$

Then, the intrinsic parameter matrix is basically given by:

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

The output obtained for the above is shown below:

```
u = 751.62880302
v = 1338.63469643
lamda = -0.640110261298
alpha = 2060.73771609
beta = 2045.65339931
gamma = -4.24008638058

Initial estimate of Calibration matrix:

[[ 2.06073772e+03 -4.24008638e+00  7.51628803e+02]
 [ 0.00000000e+00  2.04565340e+03  1.33863470e+03]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00]]
```

As it can be seen here, the principal point obtained is

$$(u_0, v_0) = (751.6288, 1338.6347)mm$$

and focal lengths

$$(f_x, f_y) = (2060.7377, 2045.653)mm$$

and skew factor is -4.2

The rotation matrix R and translation vector t for each image can be obtained from the Intrinsic matrix an = d homography matrix of that image by using the following system.

$$\begin{aligned} \mathbf{r}_1 &= \lambda \mathbf{A}^{-1}\mathbf{h}_1 \\ \mathbf{r}_2 &= \lambda \mathbf{A}^{-1}\mathbf{h}_2 \\ \mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\ \mathbf{t} &= \lambda \mathbf{A}^{-1}\mathbf{h}_3 \end{aligned}$$

with $\lambda = 1/\left\|A^{-1}h_1\right\| = 1/\left\|A^{-1}h_2\right\|$

$$R = [r_1, r_2, r_3]$$

The initial distortion is assumed to be zero and therefore

$$k_c = [0, 0]^T$$

II. Non-linear Geometric Error Minimization

First, I calculate the re-projection error before optimizing the Intrinsic parameter matrix. For each pixel coordinate in each image the error is calculated as follows:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} - \mathbf{A} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

where (u, v) is the detected corner pixel coordinate, A in intrinsic matrix, [R — t] is the transformation matrix and X, Y are 3D coordinate points corresponding to the pixel coords. The error is obtained by taking the l2 norm of the resulting vector. The above process is the same as executing the following functional over all points and all images.

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{m}_{ij} - \hat{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathtt{M}_j)\|^2 ,$$

The above functional denotes the maximal likelihood estimate which assumes that the image points are corrupted by independent and identically distributed noise.

The errors over all points and images is summed up and the total error is divided by (54*11) to get the mean reprojection error. The mean reprojection error before optimization is obtained as **1.24961151844**

To reduce the error, the functional needs to be minimized for the error while dealing with radial distortion as well. In order to obtain error due to distortion as well, the following variables are calculated:

$$\breve{u} = u + (u - u_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2]$$
$$\breve{v} = v + (v - v_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \ .$$

where k1 and k2 are radial distortion coefficients. (u_0, v_0) is the principal point obtained from intrinsic params. (x, y) is obtained from multiplying the transformation matrix with the 3D coordinates i.e

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = [R|t] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \tag{1}$$

where R and t are rotation and translation matrix for that image.

and (u, v) is obtained as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2}$$

where A is the intrinsic parameter matrix.

The error vector is then calculated as follows:

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} - \begin{bmatrix} \widetilde{u} \\ \widetilde{v} \end{bmatrix} \tag{3}$$

where (x_p, y_p) are the pixel coordinates obtained initially thru *cv2.findChessboardCorners*. The error is obtained by taking the l2 norm of the above resultant vector.

As previously done, the mean reprojection error is calculated take the summation over all points in all images the total summation is divided by (54*11). This error is what is to be minimized. This is done in python with the help of **scipy.optimize.least_squares** with the cost function returning the mean reprojection error taking the radial distortion coeffs into consideration.

Only the intrinsic params and radial distortion coeffs are considered to minimize the error. Once the error is minimized, the result is shown as follows:

```
Mean Reprojection error before optimization:
1.24961151844

Calibration matrix after optimization:

[[ 2.96319410e+03 -1.81920284e+03  9.27485977e+02]
 [ 0.00000000e+00  1.42506080e+03  1.39356469e+03]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00]]

Distortion coefficients after optimization:
0.0158888379148, -0.0289728808506

Mean Reprojection error after optimization:
1.1942793426
```

After optimization, the principal point obtained is

$$(u_0, v_0) = (927.486, 1393.5647)mm$$

and focal lengths

$$(f_x, f_y) = (2963.1941, 1425.0608)mm$$

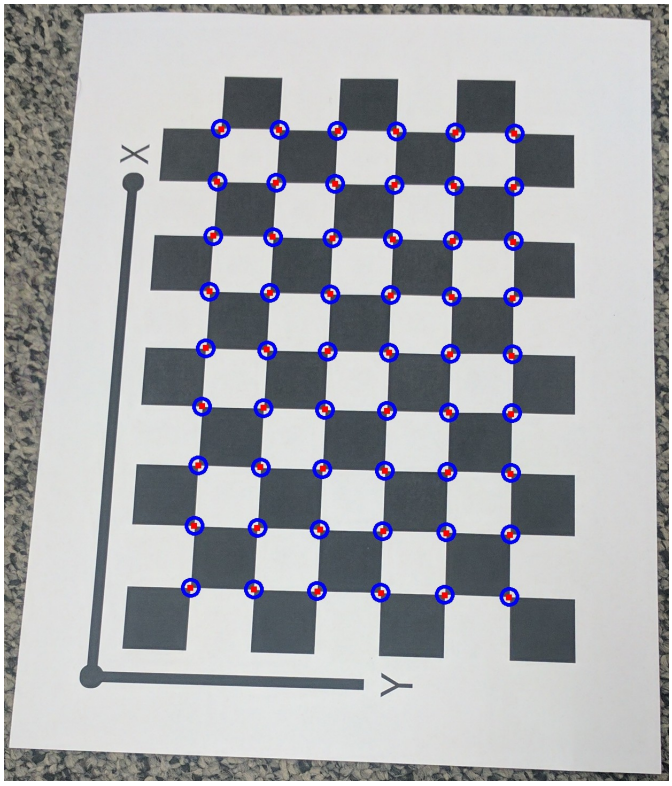
and skew factor is -1.8192

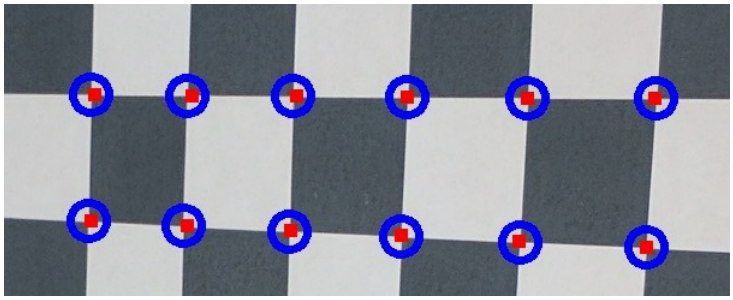
The distortion coeffs after optimization are

$$(k_1, k_2) = (0.0158888379148, -0.0289728808506)$$

The mean reprojection error after optimization is **1.1942793426**

The reprojected points and the original detected corners are plotted on the original images as shown below. The original detected corners are represented by blue circles and the reprojected points are denoted by red squares.





The rest of the output images are in a folder named *Output* in the submission directory after running the code.

REFERENCES

[1] Zhengyou Zhang, *A Flexible New Technique for Camera Calibration*, Microsoft Research, Redmond, WA, 1998.