# Homework 1
# Auto Calibration

Lih-Narn Wang
Master of Engineering in Robotics
University of Maryland, College Park
Email: ytcdavid@terpmail.umd.edu

## I. Initial Parameters Estimation

In this section, I will give a detailed explanation of how to estimate the intrinsic matrix, extrinsic matrix, and the distortion parameters to have a good initial guess to do a non-linear optimization.

### A. Find Homography

The first thing is to create a set of 3D points as our model points. Based on the paper, the 3D points should be on the plane: $Z = 0$. We are using a (9 x 6) chess board as our calibration images, so I created a set of mesh points based on the grid with a scale equals to 21.5. After that, I got a ndarray of model points: (54, 3). Secondly, I used *cv2.findChessboardCorners* along with *cv2.cornerSubPix* to find and refine the chess board points on the images. After that, because I have 13 images, I got a ndarray of image points: (13, 54, 2). Finally, I used least-square method to find a homography from model points to image points for each images. After this, I ended up with 13 homographies.

*1) Normalization:* Though the paper [1] didn't state that I have to do normalization before finding the Homographies, the other paper [2] suggest that I can do normalization to find a better perspective transformation. One is to treat the **x, y** components the same, and normalized both axis together. The other is to treat them as different components, and normalized two axis separately. So after this small step, I have three sets of model points and image points. One is the original data points and others are from two different normalization. The difference of them will be presented and discussed in the **Result** section.

### B. Estimate Intrinsic Matrix

After knowing the Homographies between model points and image points, we can extract Intrinsic matrix and Extrinsic matrix from them. By using the two constraints provided by the property that the vectors in a rotation matrix have orthonormality, we can easily derive the formulas in the papers[1][2]. Using these formulas, we can extract Intrinsic matrix from the Homographies. After this step, I ended up having only one Intrinsic matrix with dimension: (3, 3)

### C. Estimate Extrinsic Matrix

With the Intrinsic Matrix, I can easily extract the Extrinsic matrix from the Homographis. One thing worth our attension is that the Extrinsic matrix is composed by a rotation matrix and a translation matrix, so there will be 13 different Extrinsic matrix for 13 images. After this step, I ended up with a set of Extrinsic matrix with dimension: (13, 3, 4)

*1) Approximate the Rotation Matrix:* Because of noise, the rotation matrix encoded in the extrinsic matrix may not be valid. From the paper[1], I approximated the valid rotation matrix. The difference of original and approximated extrinsic matrix will be presented and discussed in the **Result** section.

### D. Distortion

Because there are distortion in the images, so we should undistort the image to minimize the reprojection error. However, we can only estimate the coefficients of the distortion by non-linear operations, so the paper suggest us to make a guess for initialization and optimize it in the last step. From the paper[1], I'll only consider the first two coefficients. And based on the instructions, I gave an initial guess that $k_d = [0, 0]$. The formula in the paper is very confusing, so I used the formula provided by MATLAB[3]. The equations from my understanding are:

$u_{dis} = u_{undis} + (u_{undis} - u_0)[k_1(x^2 + y^2) + k2(x^2) + y^2)^2]$
$v_{dis} = v_{undis} + (v_{undis} - v_0)[k_1(x^2 + y^2) + k2(x^2) + y^2)^2]$
$x = u_{undis}/f_x, y = v_{undis}/f_y$

In my case, $u_{undis}, v_{undis}$ is the projected points of model points. $u_0, v_0$ is the parameter in the intrinsic matrix, which is the center of the distortion. $x, y$ are projected points of model points normalized by their focal length, so they are dimensionless.

## II. Non-linear Geometric Error Minimization

Based on the parameters I got in the previous steps, I had a pretty good initialization of the camera model. Now, I can use iterative technique to minimize the reprojection error. One thing worth notice is the parameters to optimize. There are 9 parameters in the rotation matrix, but only 3 degree of freedoms, so I transformed my rotation matrix in the extrinsic matrix into 3 parameters based on the method described in [2]. After this, I ended up with 85 parameters, 5 from intrinsic matrix, 2 from the distortion, and $(3 + 3) * 13$ from the extrinsic matrix. I used *scipy.optimize.minimize* to minimize the geometric error, and the results are shown in the next section.

## III. Results

In this section, I'll present the re-projection error without the approximation of rotation matrix, the re-projection error before the optimization, and the re-projection error after the optimization of three sets of points I mentioned above. Also, I'll present the Intrinsic matrix and the distortion coefficients. Finally, I'll present the best result of undistort images with re-projection points marked in red dots. I'll use **A** to represent the results of original data points, **B** to represent the results of normalized data points(treat x,y as the same axis), and **C** to represent the results of normalized data points(normalized x,y axis separately)

### *A. Re-projection error*

TABLE I: Re-projetion error of each model
(mean pixel distance)

| Data Set | W/O Approximation | W/ Approximation | Optimization |
|---|---|---|---|
| **A** | 0.698239 | 1.959240 | 0.583367 |
| **B** | 0.698239 | 1.917707 | 0.583373 |
| **C** | 0.714143 | 1.818168 | 0.583373 |

From the results, we can see that the normalization of points didn't have positive effect before the approximation of the rotation matrix, but it does help after the approximation. And the error after the approximation is high compare to the one before the approximation. It makes sense because we applied constrained on the matrix we found, so the error goes up. Lastly, the optimization works great. Because our initialization of three cases are similar, they all converged to a very similar Intrinsic matrix and distortion coefficients, which I believed is the global minimum. The error after the optimization may come from the error of corner detection (chess board corners on the images).

### *B. Intrinsic Matrix & Distortion Coefficients*

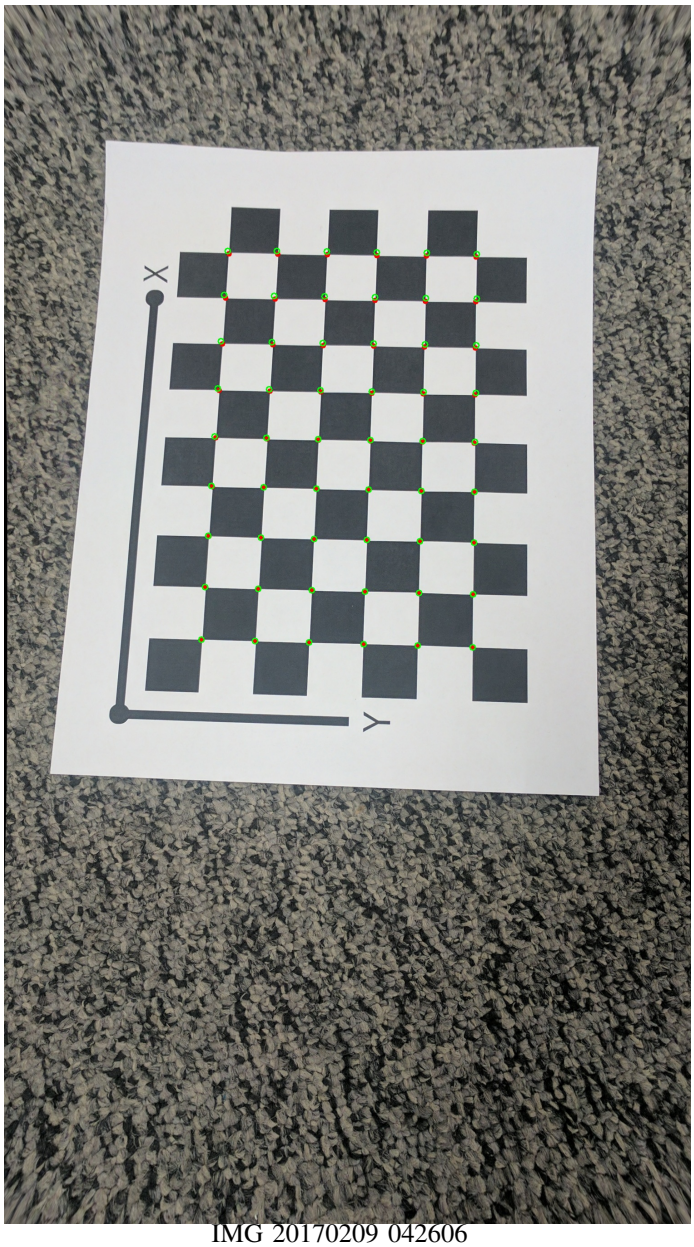
I actually got 3 different Intrinsic Matrix and 3 different distortion coefficients. But due to the reason I mentioned above, they all look very similar. As a consequence, I'll just present one of them.

$$Intrinsic = \begin{bmatrix} 2048.663 & -1.87399 & 758.663 \\ 0 & 2040.876 & 134.509 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distoriton = [0.173268, -0.754788]$$

### *C. Rectified Images*

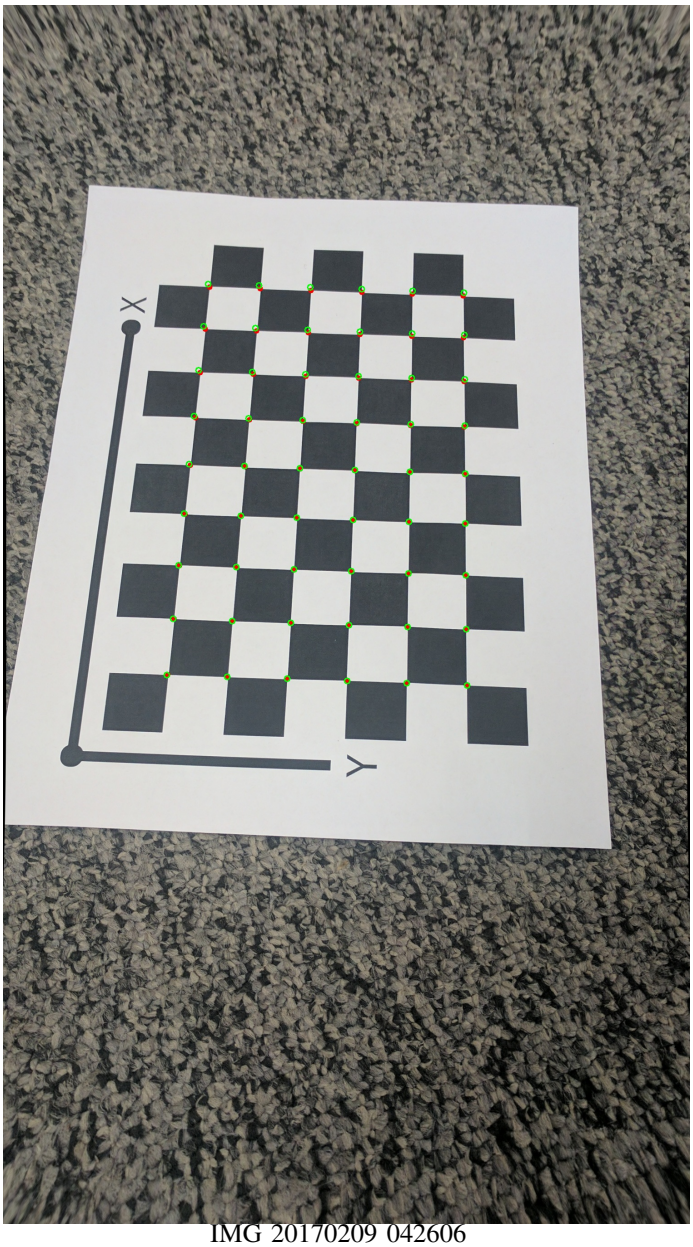
The green circles are the original projected points, and the red dots are the undistort projected points.



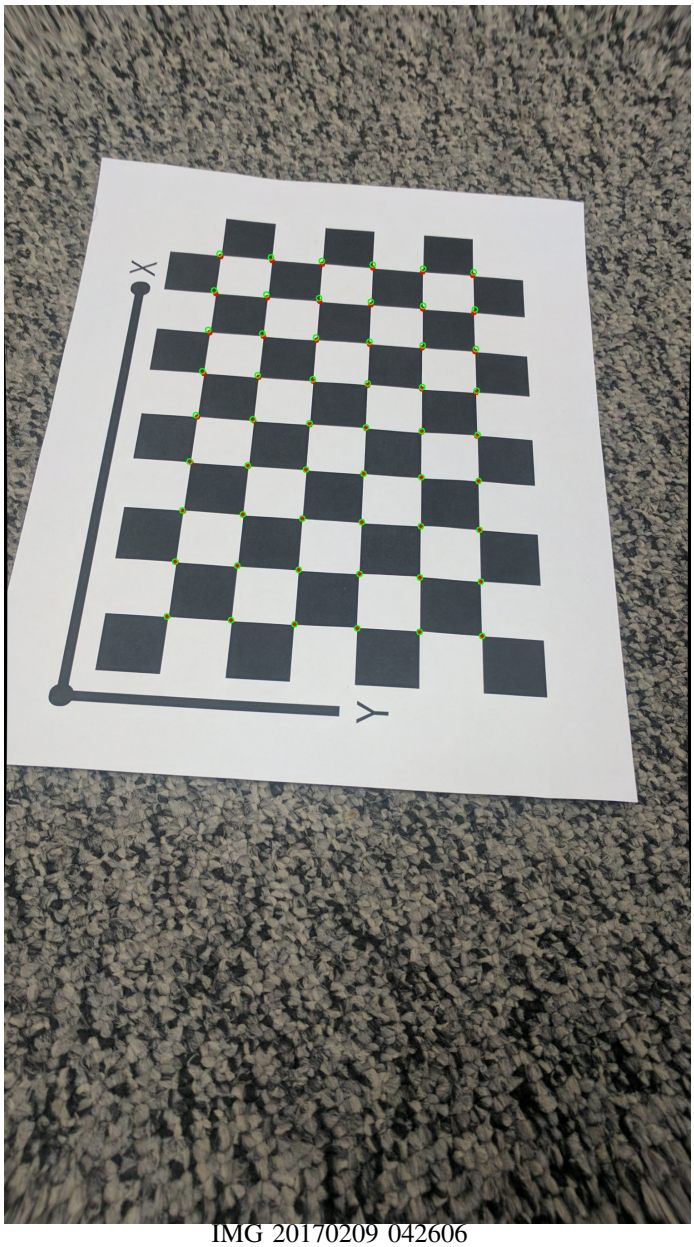

IMG 20170209 042606

## References

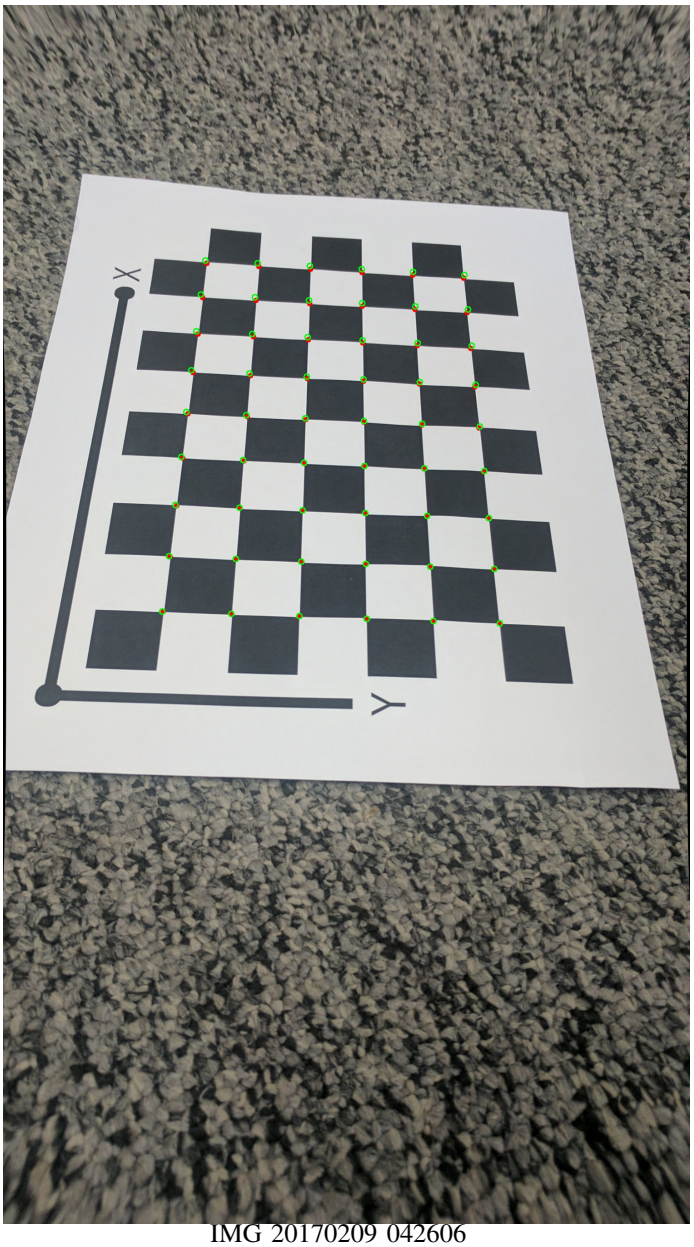
[1] Zhengyou Zhang, "A Flexible New Technique for Camera Calibration"
[2] Wilhelm Burger, "Zhang's Camera Calibration Algorithm: In-Depth Tutorial and Implementation"
[3] MATLAB,https://www.mathworks.com/help/vision/ug/camera-calibration.html
[4] CMSC733, https://cmsc733.github.io/2019/hw/hw1/



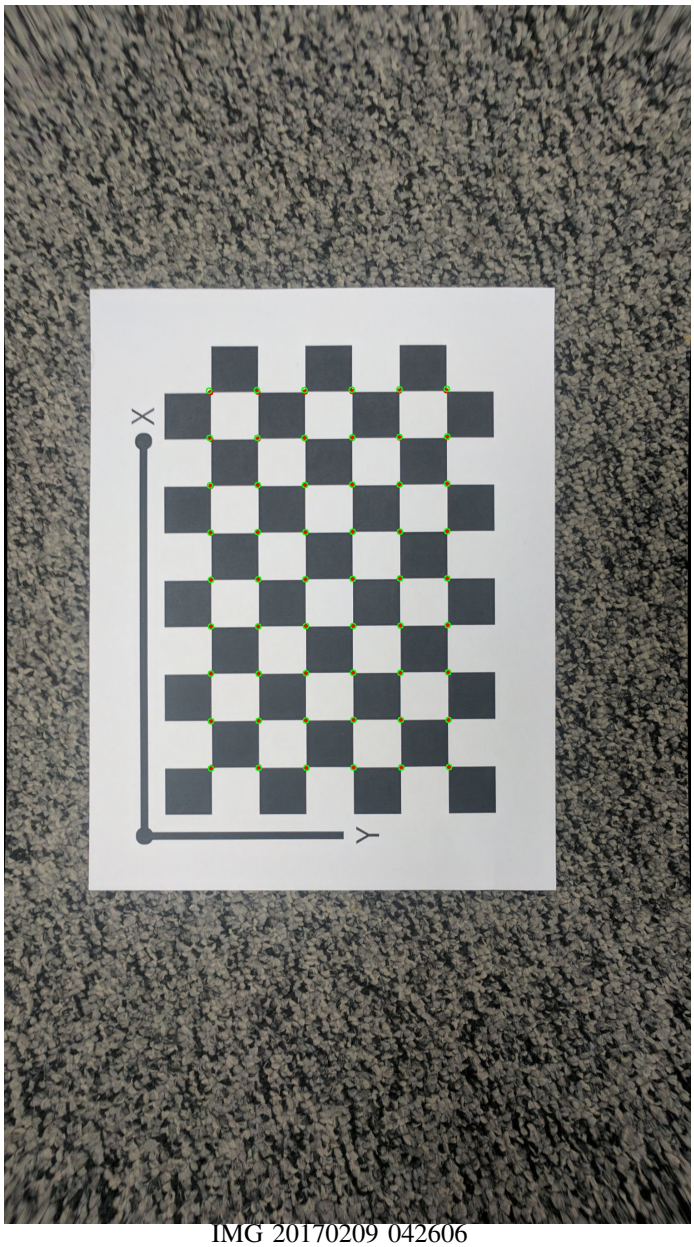

IMG 20170209 042606



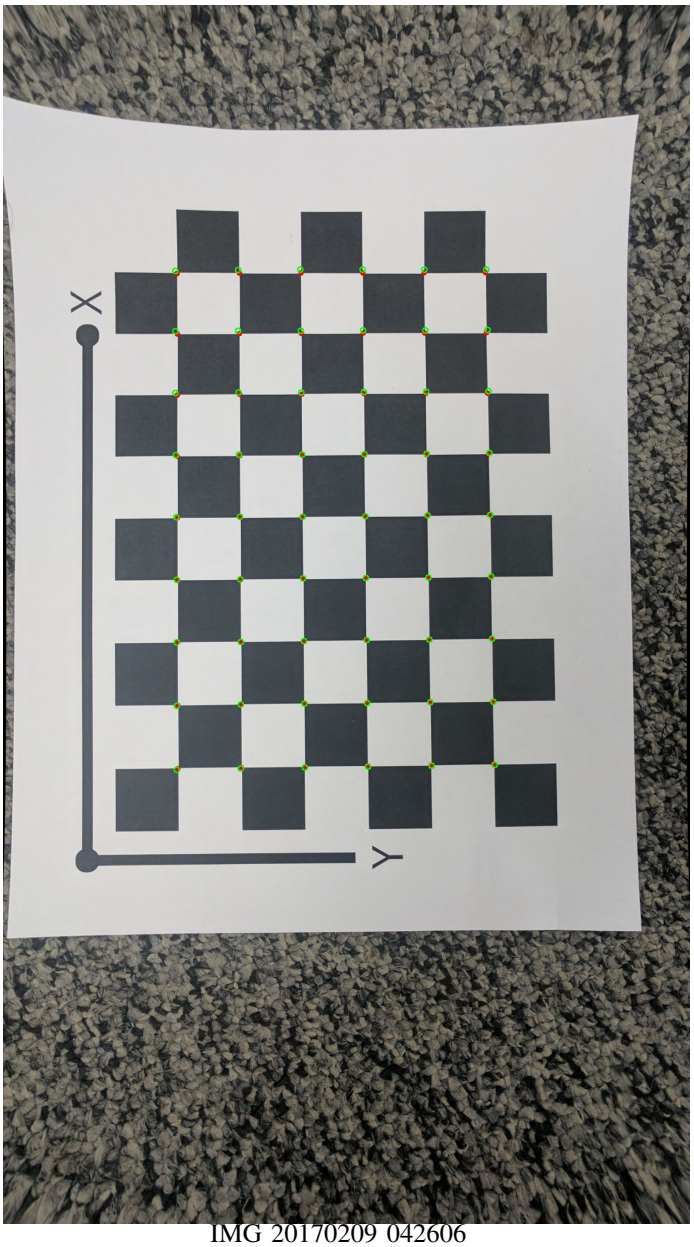

IMG 20170209 042606



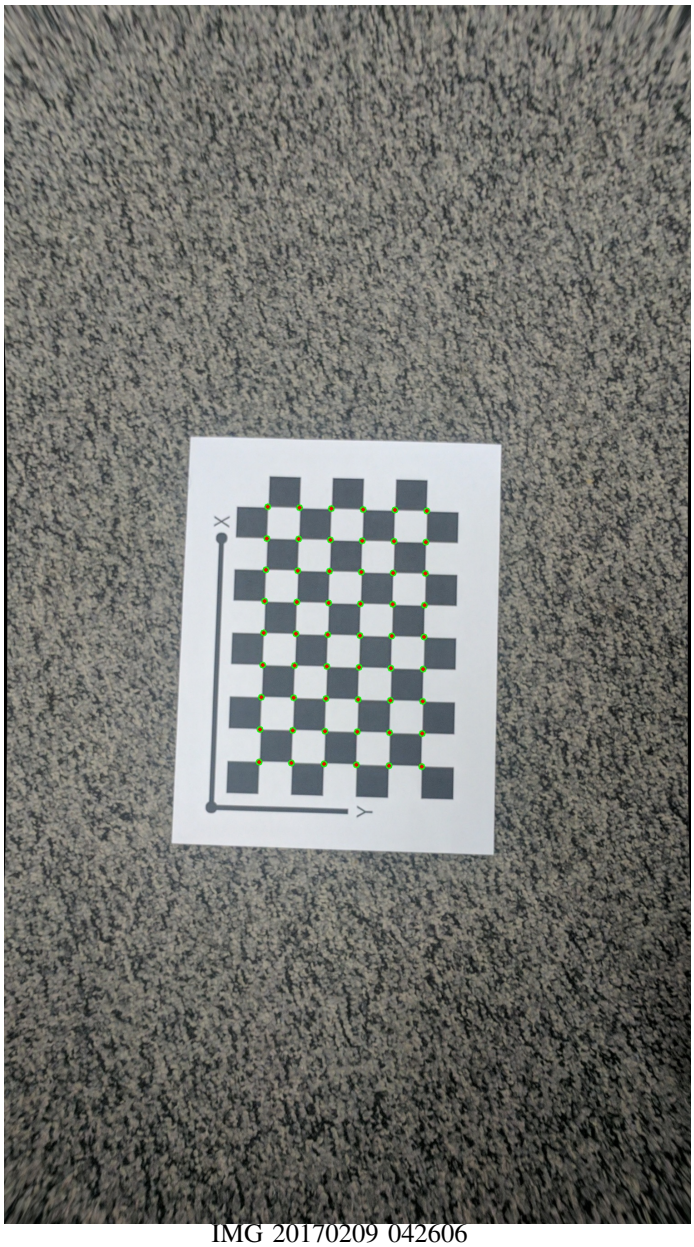

IMG 20170209 042606



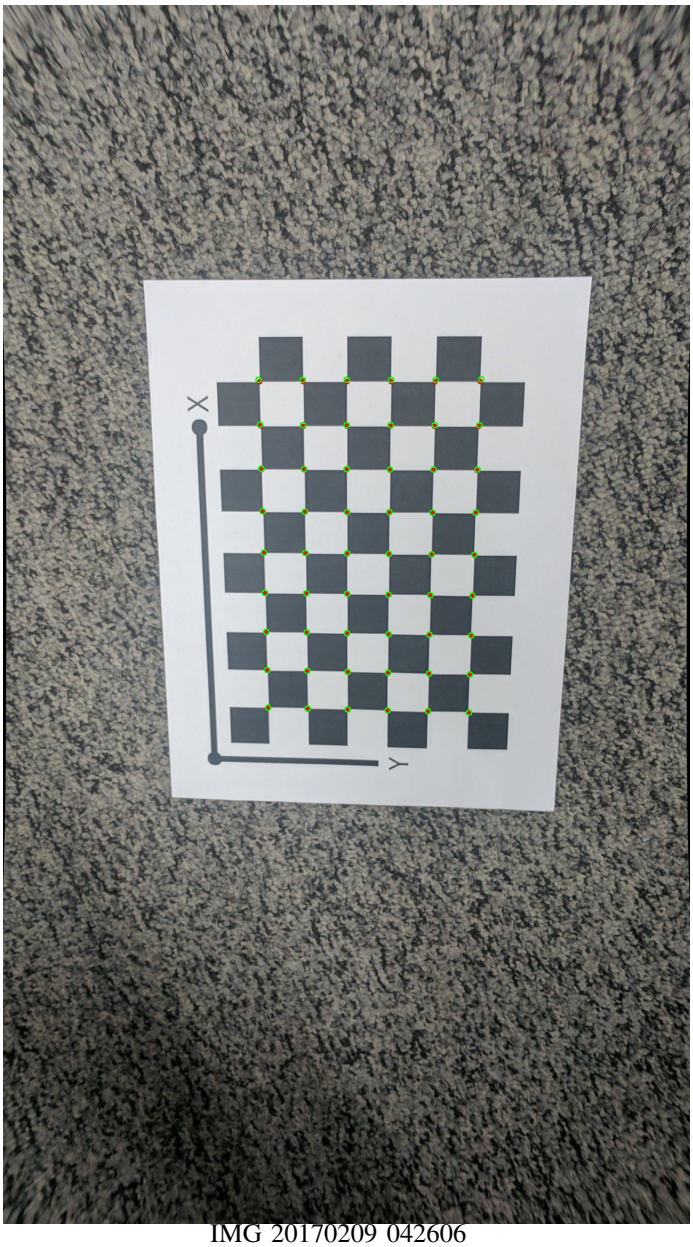

IMG 20170209 042606



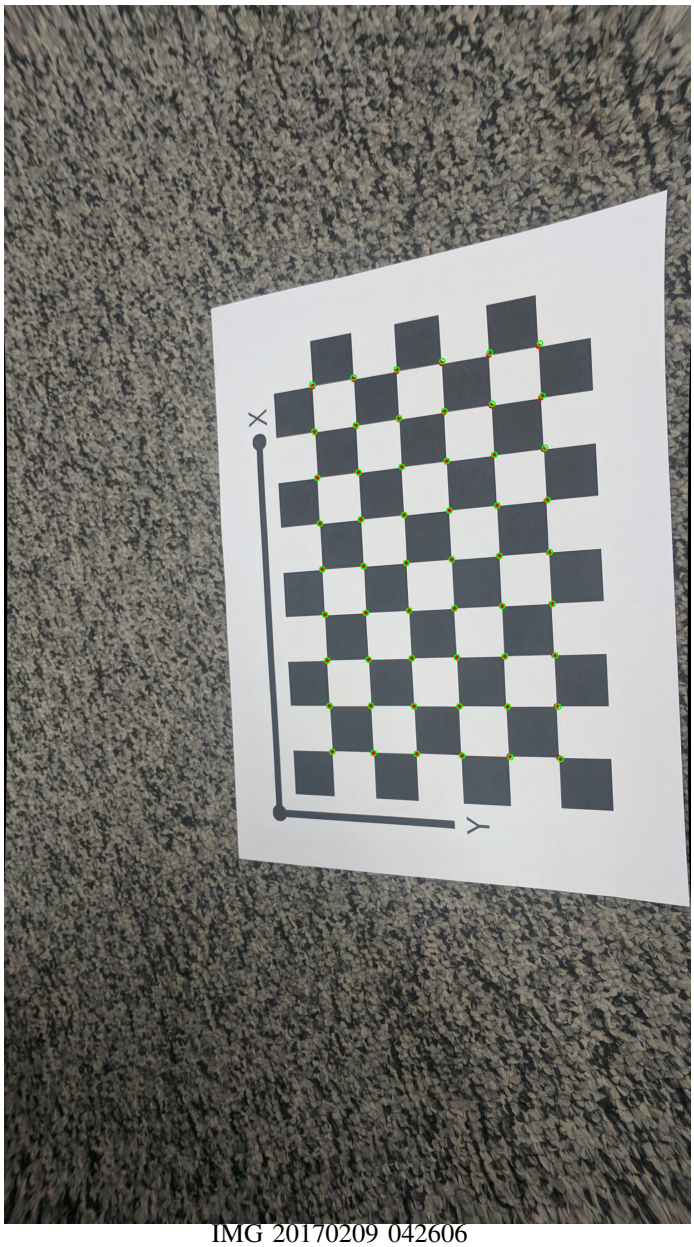

IMG 20170209 042606



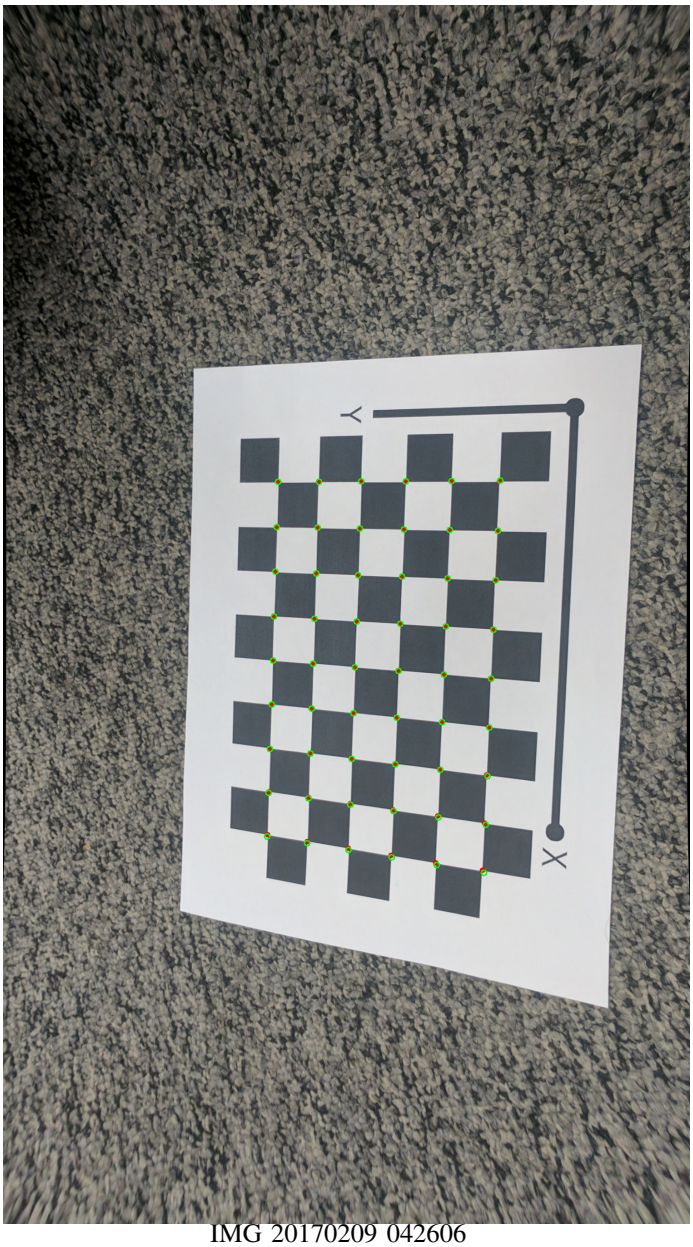

IMG 20170209 042606



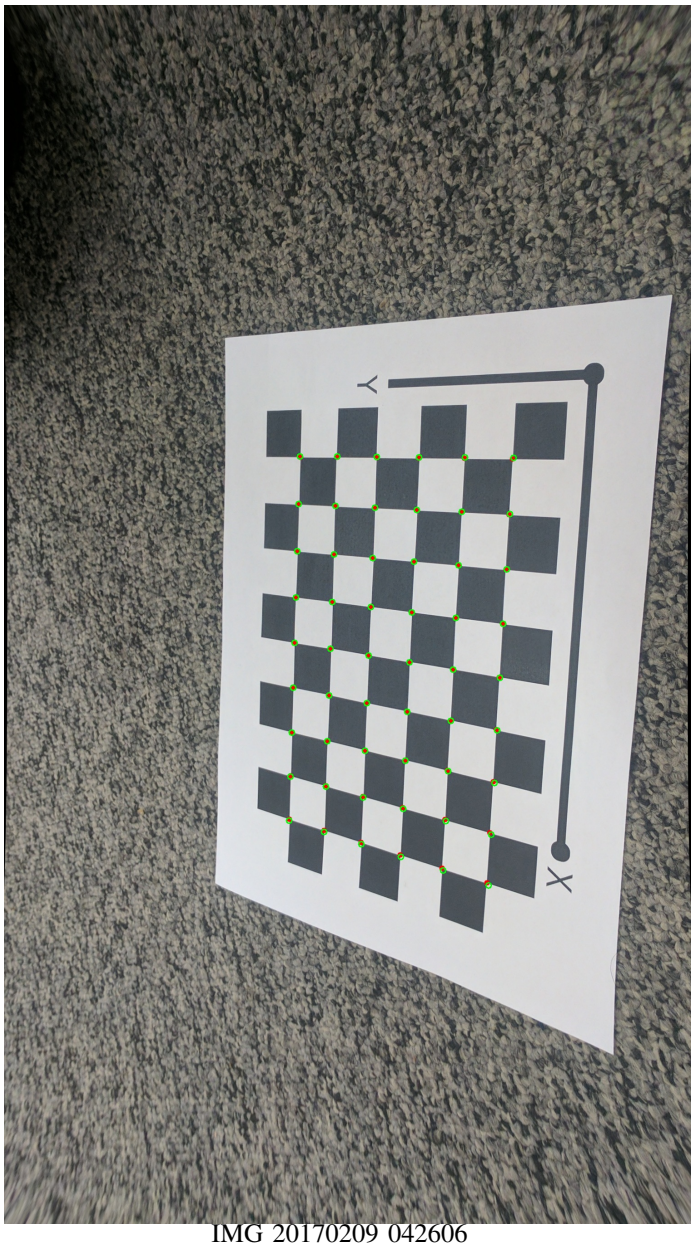

IMG 20170209 042606



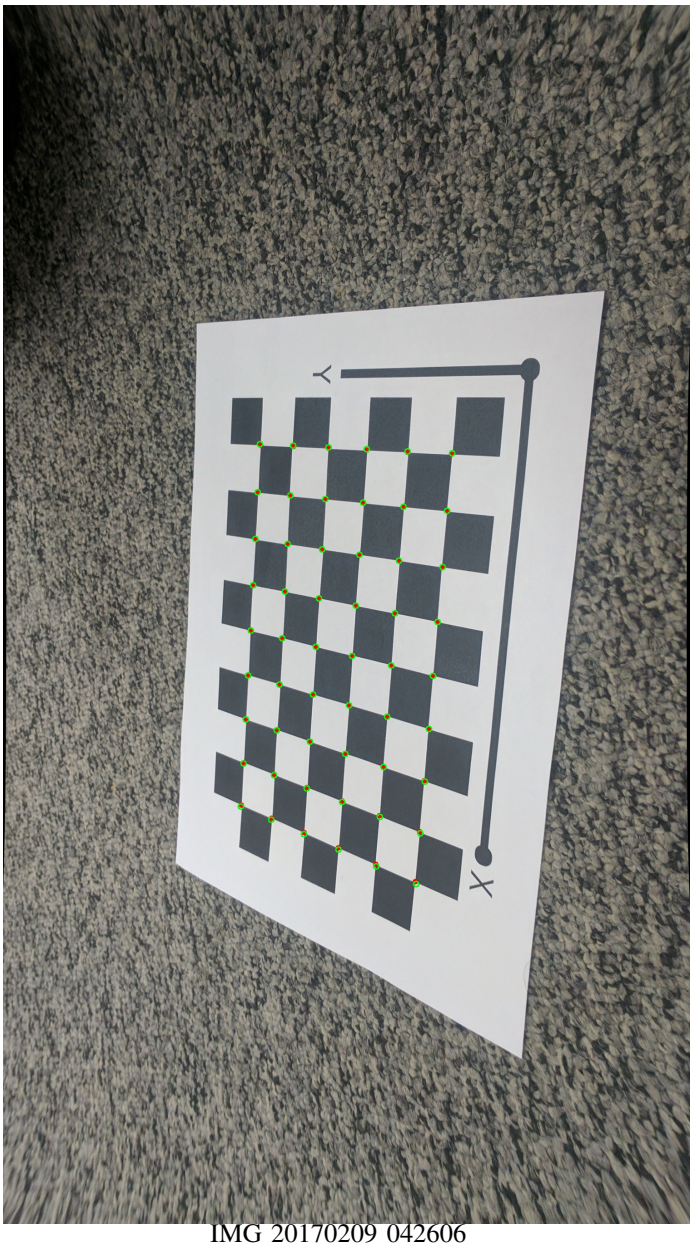

IMG 20170209 042606



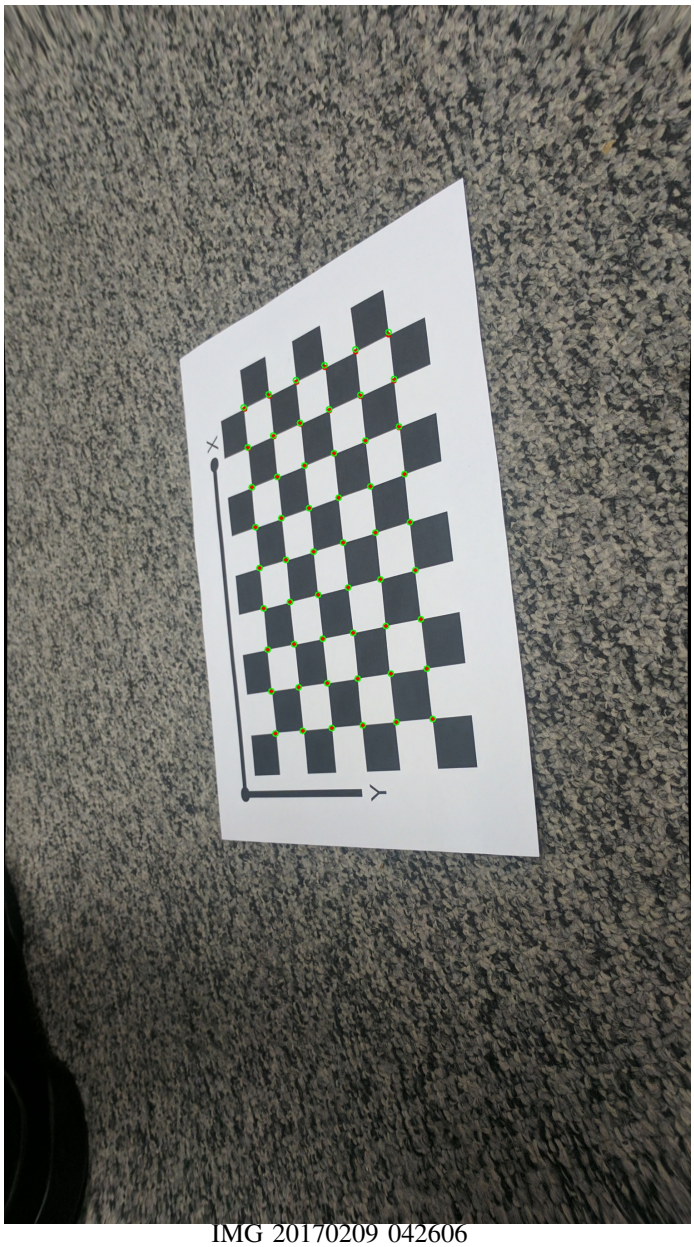

IMG 20170209 042606




IMG 20170209 042606




IMG 20170209 042606




IMG 20170209 042606