HW1 : AutoCalib

Venkateshkrishna
Masters in Robotics
Worcester Polytechnic Institute
Worcester, MA 01609
vparsuram@wpi.edu

***Abstract*—The goal of this project is to implement Zhang's calibration, which is a calibration method to estimate the camera's intrinsic parameters while also computing extrinsic parameters.**

I. Introduction

The calibration process involves the estimation of the intrinsic parameters which include the K-matrix and the distortion and extrinsic parameters which includes the rotation matrix R and translation vector t. A nonlinear optimization technique minimizes projection errors and find the intrinsic parmaters: K matrix and distortion coefficients.

II. Methodology

The following subsections include:

1) Initial Parameter Estimation
2) Non-linear Geometric error minimization

A. Initial Parameter Estimation

This involves 3 steps:

1) Estimating K matrix
2) Estimating Extrinsics
3) Setting appropriate distortion

1) Estimating K matrix: Initially, corners in the image are identified using the `cv2.findChessboardCorners()` function, yielding both world and image points. Subsequently, the homography between them is determined through Singular Value Decomposition. This results in a series of Homography matrices. These matrices are then employed to compute the V matrix using the provided equations:

$$b = \begin{bmatrix} B_{11} \\ B_{12} \\ B_{22} \\ B_{13} \\ B_{23} \\ B_{33} \end{bmatrix}$$

$$v_{ij} = \begin{bmatrix} h_{i1}h_{j1} & h_{i1}h_{j2} + h_{i2}h_{j1} & h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} & h_{i3}h_{j2} + h_{i2}h_{j3} & h_{i3}h_{j3} \end{bmatrix}$$

$$\begin{pmatrix} v_{12}^T \\ (v_{11} - v_{22}) \end{pmatrix} \cdot b = 0.$$

The b vector is solved for using SVD. Once we have the b vector, the K matrix is obtained using these equations.

$$v_0 = \frac{B_{12}B_{13} - B_{11}B_{23}}{B_{11}B_{22} - B_{12}^2}$$

$$\lambda = B_{33} - \frac{B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})}{B_{11}}$$

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

$$\beta = \frac{\lambda B_{11}}{B_{11}B_{22} - B_{12}^2}$$

$$\gamma = -\frac{B_{12}\alpha^2\beta}{\lambda}$$

$$u_0 = \frac{\gamma v_0}{\beta} - \frac{B_{13}\alpha^2}{\lambda}$$

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

2) Estimating Extrinsics: Once we have the intrinsics the extrinsics are computed using:

$$r_1 = \lambda K^{-1} h_1$$
$$r_2 = \lambda K^{-1} h_2$$
$$r_3 = r_1 \times r_2$$
$$t = \lambda K^{-1} h_3$$

3) Setting appropriate distortion: We assume that the camera has minimal distortion and set $k = \begin{bmatrix} 0 & 0 \end{bmatrix}^\top$.

B. Non-linear Geometric error minimization

The geometric projection error is given by:

$$\sum_{i=1}^{N} \sum_{j=1}^{M} \| x_{i,j} - \hat{x}_{i,j}(K, R_i, t_i, X_j, k) \|$$

We use scipy.optimize to minimize the geometric error by converging to the optimal K matrix and k values. The values estimated earlier are given as initial guesses to this function. Essentially, what we are trying to do is:

$$\text{argmin}_{f_x,f_y,c_x,c_y,k_1,k_2} \sum_{i=1}^{N} \sum_{j=1}^{M} \|x_{i,j} - \hat{x}_{i,j}(K, R_i, t_i, X_j, k)\|$$

III. Results

The K matrix before and after optimization are:

$$K_{\text{before}} = \begin{bmatrix} 2081.75670 & 10.8605700 & 788.666042 \\ 0.00000000 & 2054.56055 & 1393.42102 \\ 0.00000000 & 0.00000000 & 1.00000000 \end{bmatrix}$$

$$K_{\text{after}} = \begin{bmatrix} 2081.75426 & 10.8591247 & 788.673355 \\ 0.00000000 & 2054.55426 & 1393.43592 \\ 0.00000000 & 0.00000000 & 1.00000000 \end{bmatrix}$$

The distortion values obtained are:

$$k = \begin{bmatrix} 0.0115352934928 95827 \\ -0.07656073898987957 \end{bmatrix}$$

The average projection error before and after optimization are:

Error before optimization	Error after optimization
0.697632389619173	0.683228330697036

The corners detected on some the images can be seen in Fig 1 through Fig 4.

The re projected corners on the undisturbed images can be seen in Fig 5 though Fig 17.

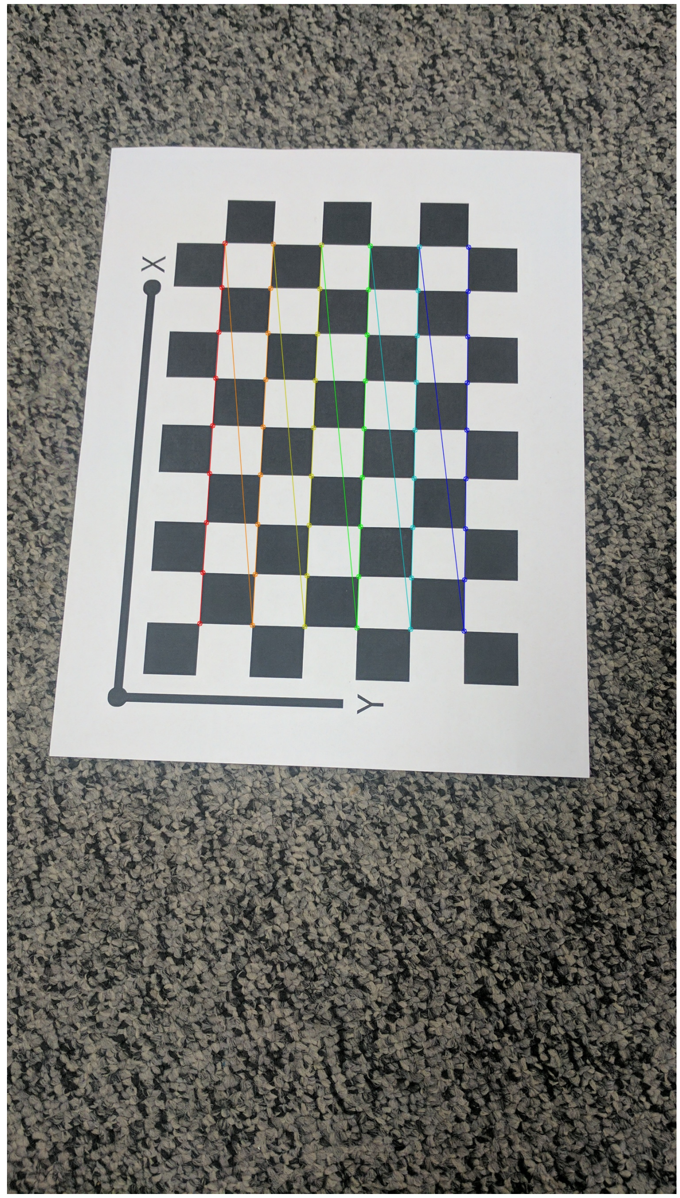

Fig. 1: Corners detected on Image 1

References

[1] Zhang's Calibration: link

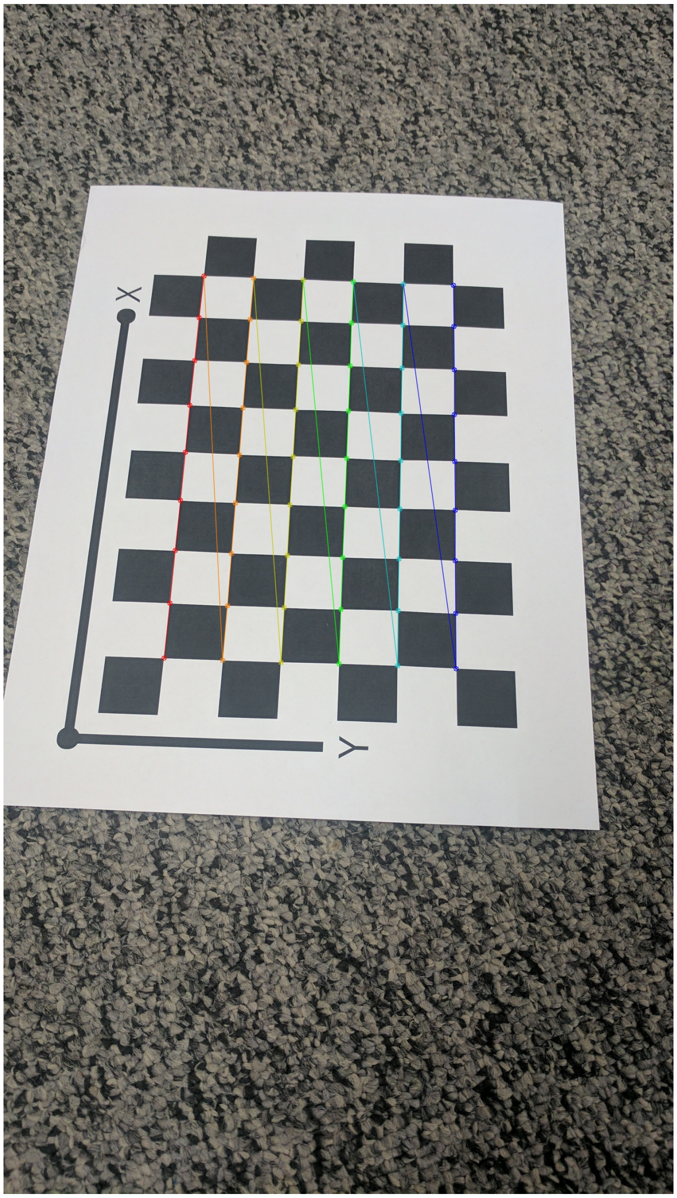

Fig. 2: Corners detected on Image 2

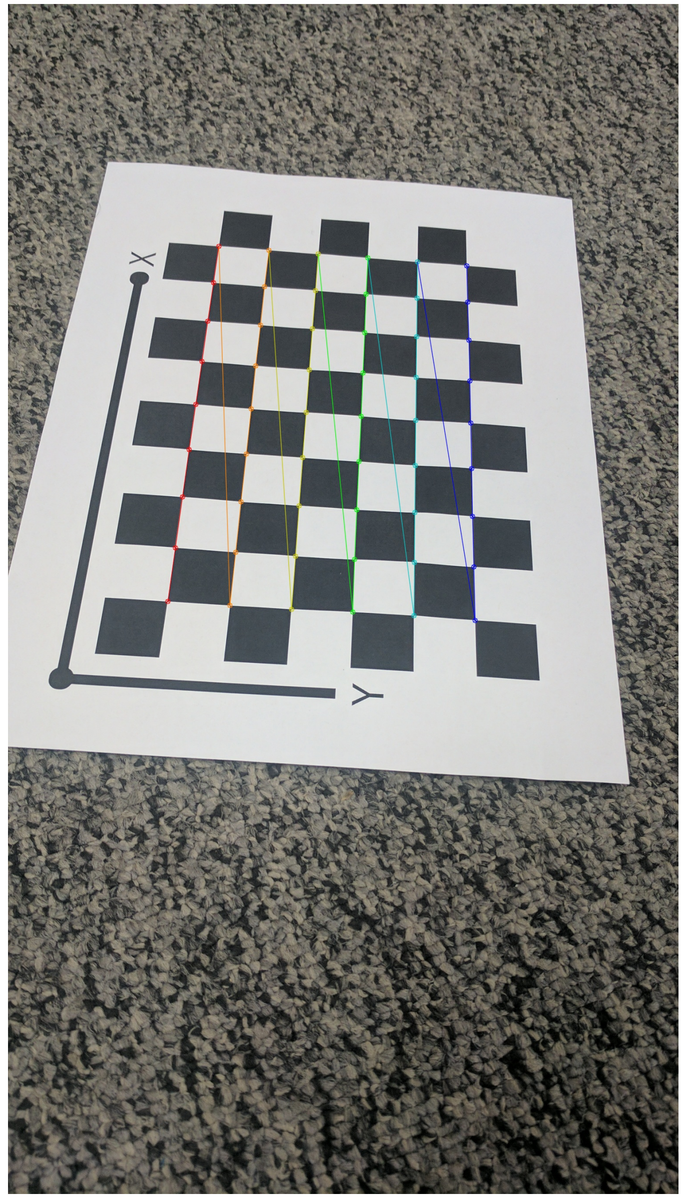

Fig. 3: Corners detected on Image 3

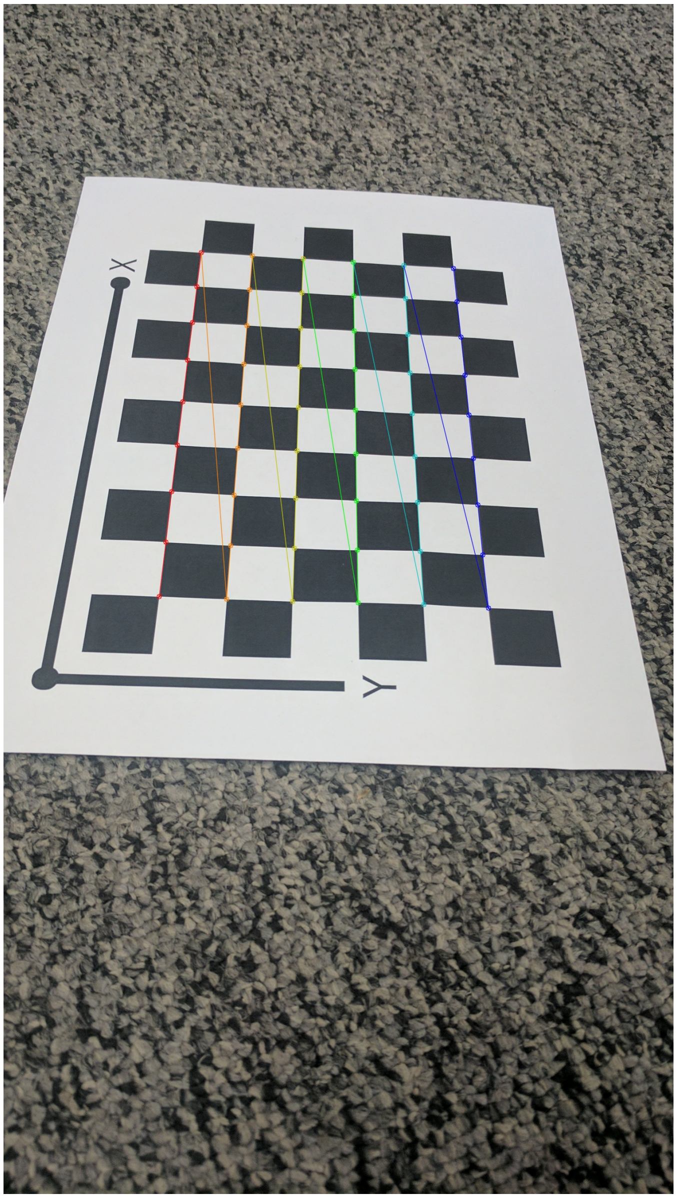

Fig. 4: Corners detected on Image 4

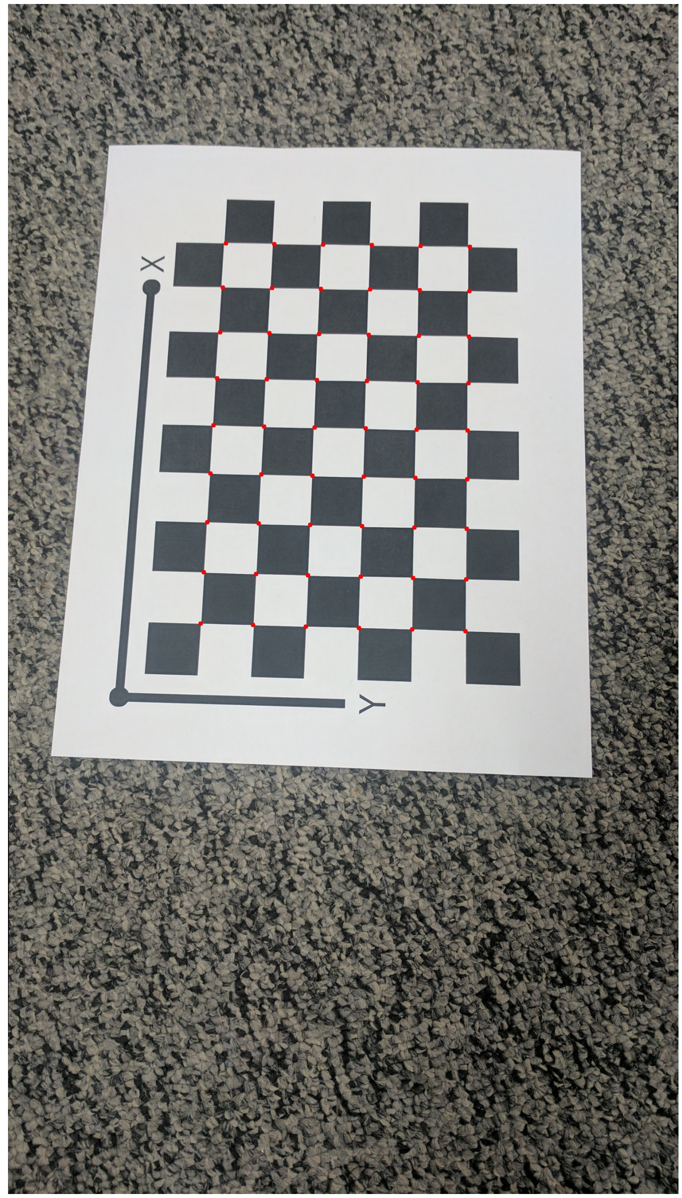

Fig. 5: Reprojected corners on Undistorted Image 1

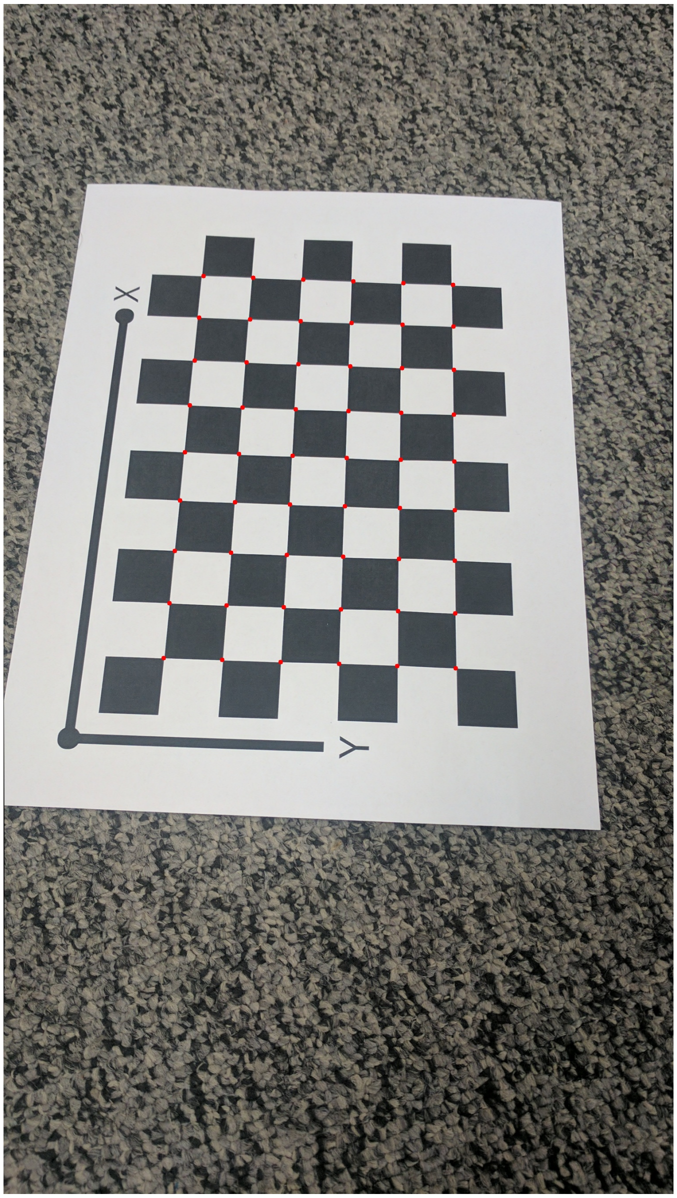

Fig. 6: Reprojected corners on Undistorted Image 2

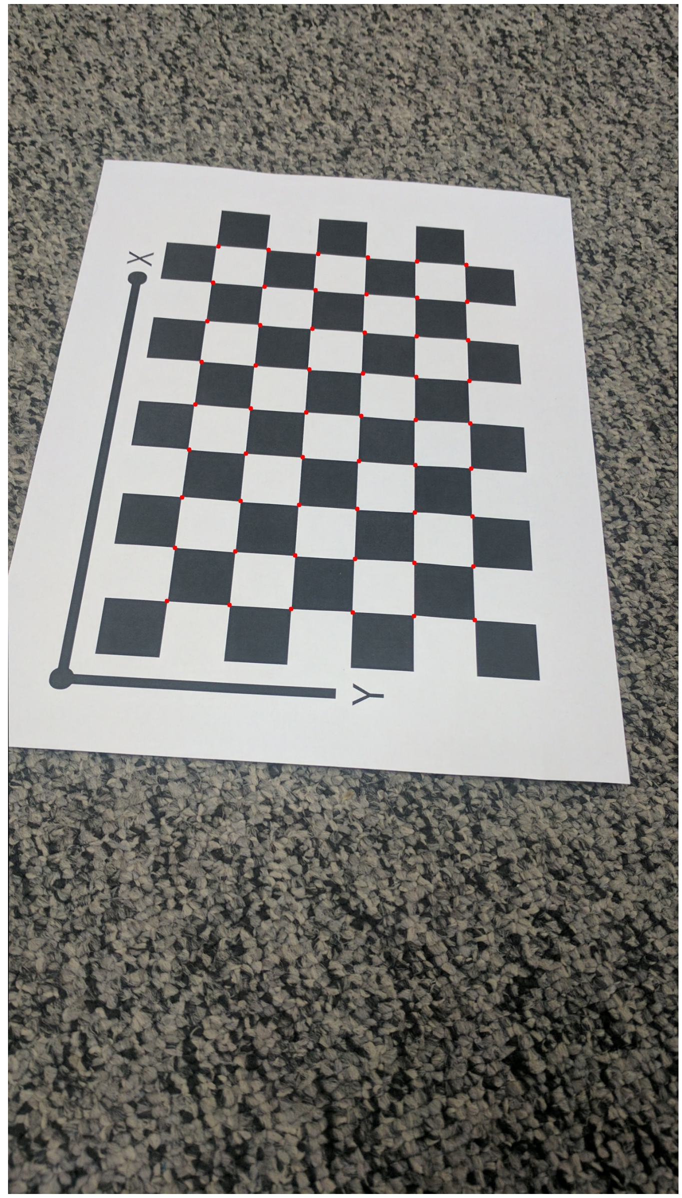

Fig. 7: Reprojected corners on Undistorted Image 3

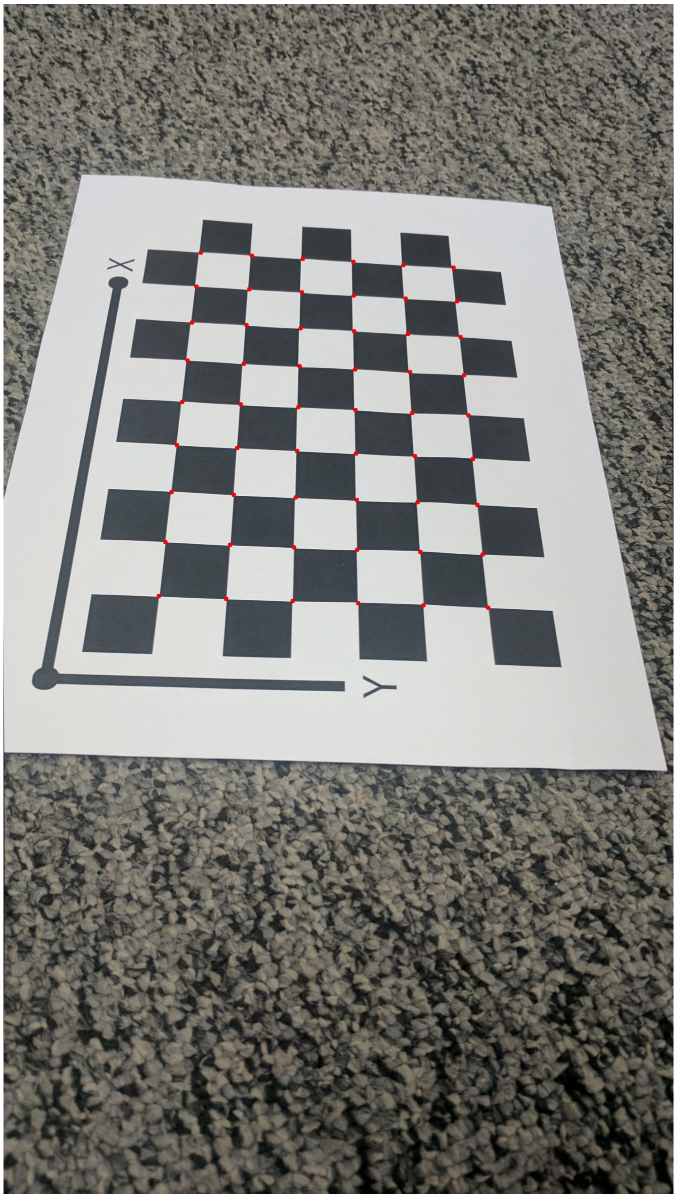

Fig. 8: Reprojected corners on Undistorted Image 4

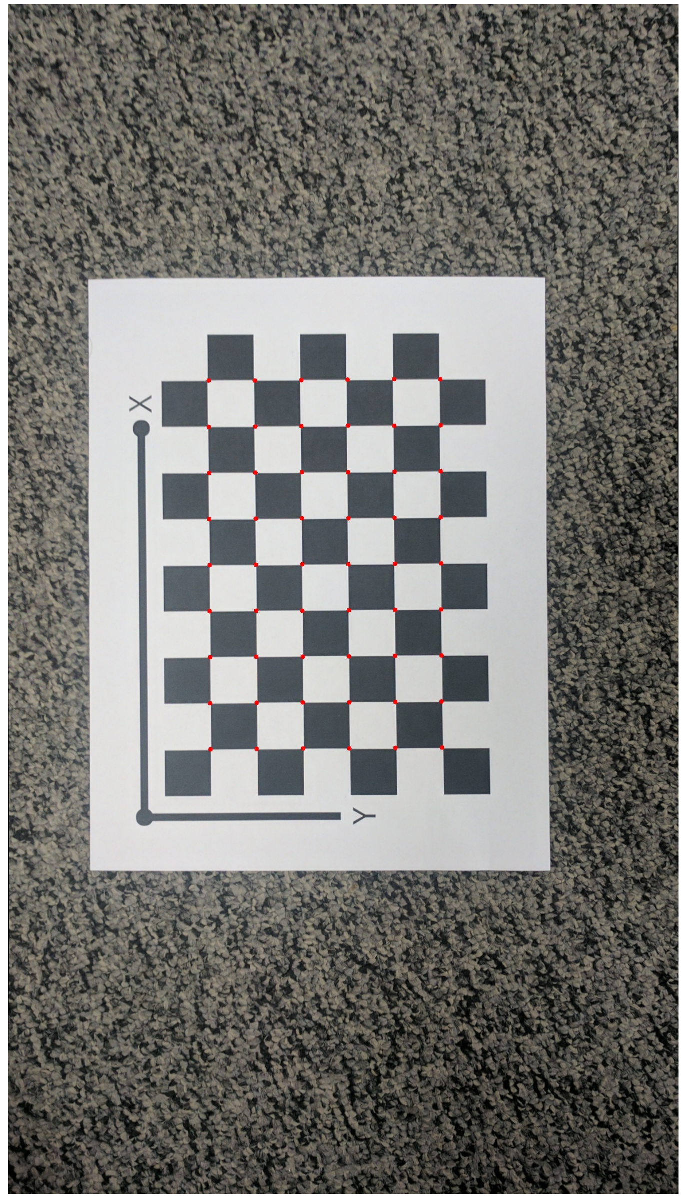

Fig. 9: Reprojected corners on Undistorted Image 5

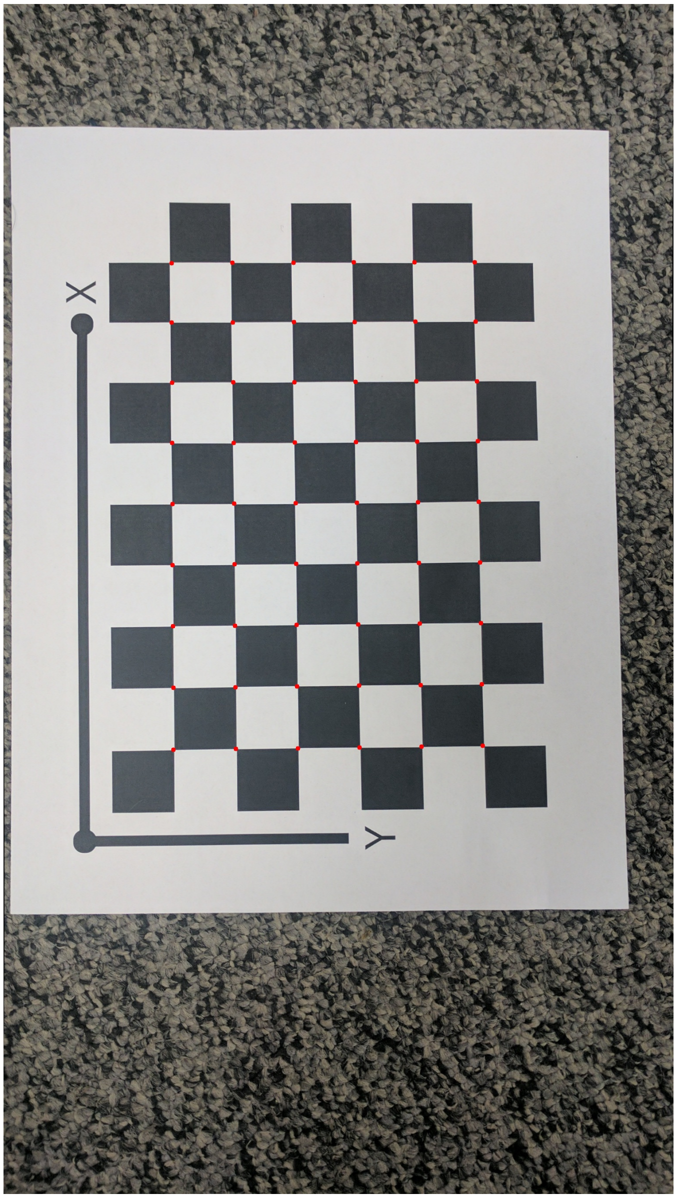

Fig. 10: Reprojected corners on Undistorted Image 6

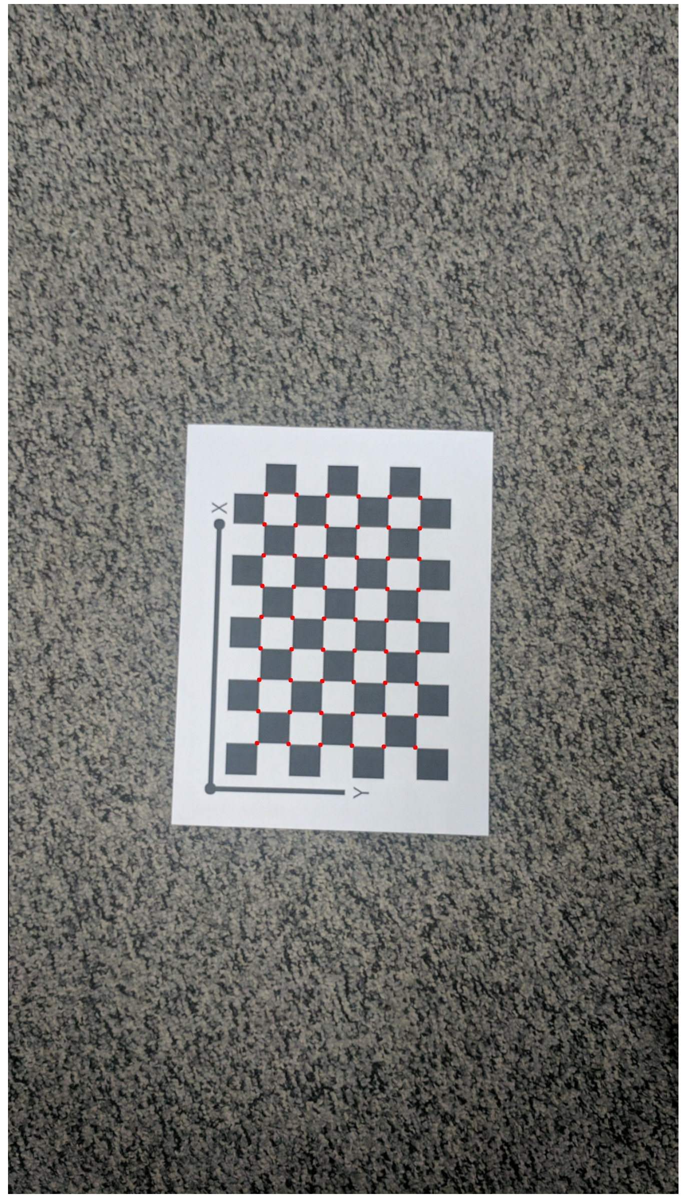

Fig. 11: Reprojected corners on Undistorted Image 7

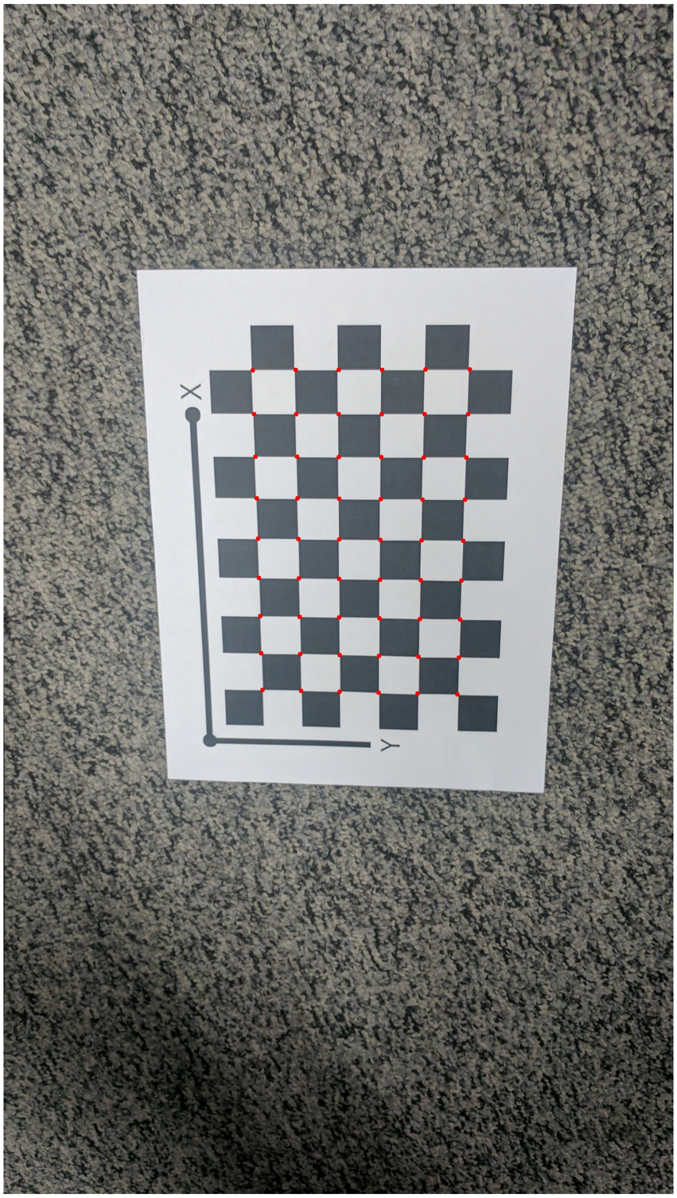

Fig. 12: Reprojected corners on Undistorted Image 8

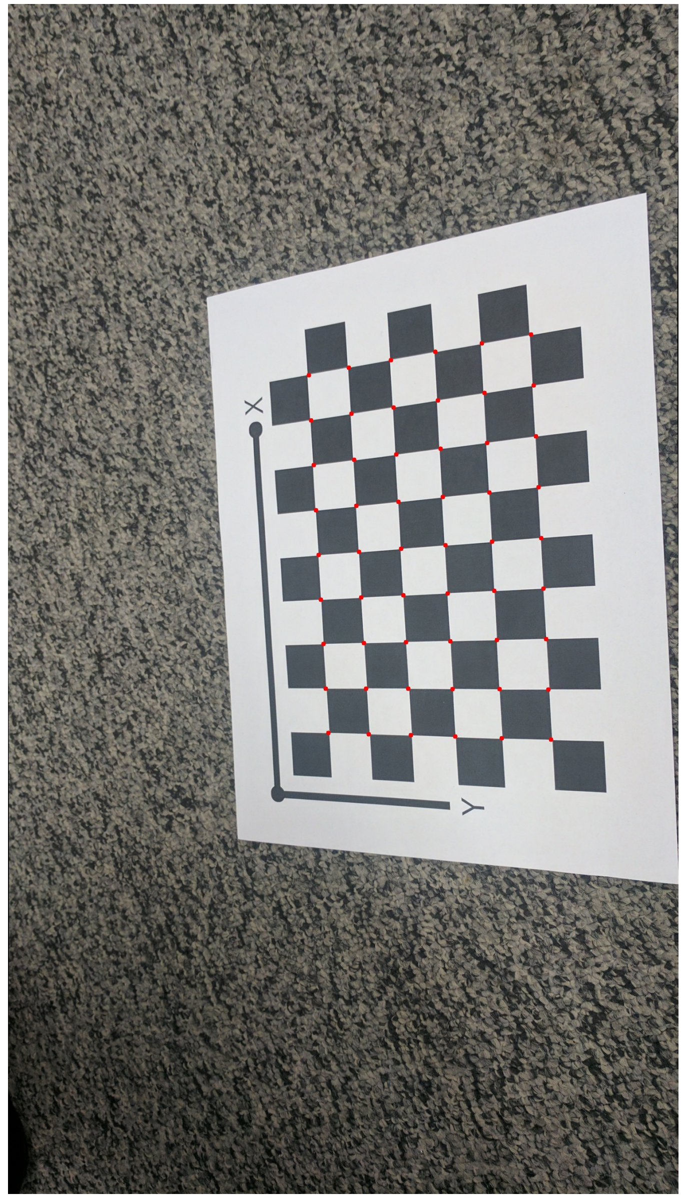

Fig. 13: Reprojected corners on Undistorted Image 9

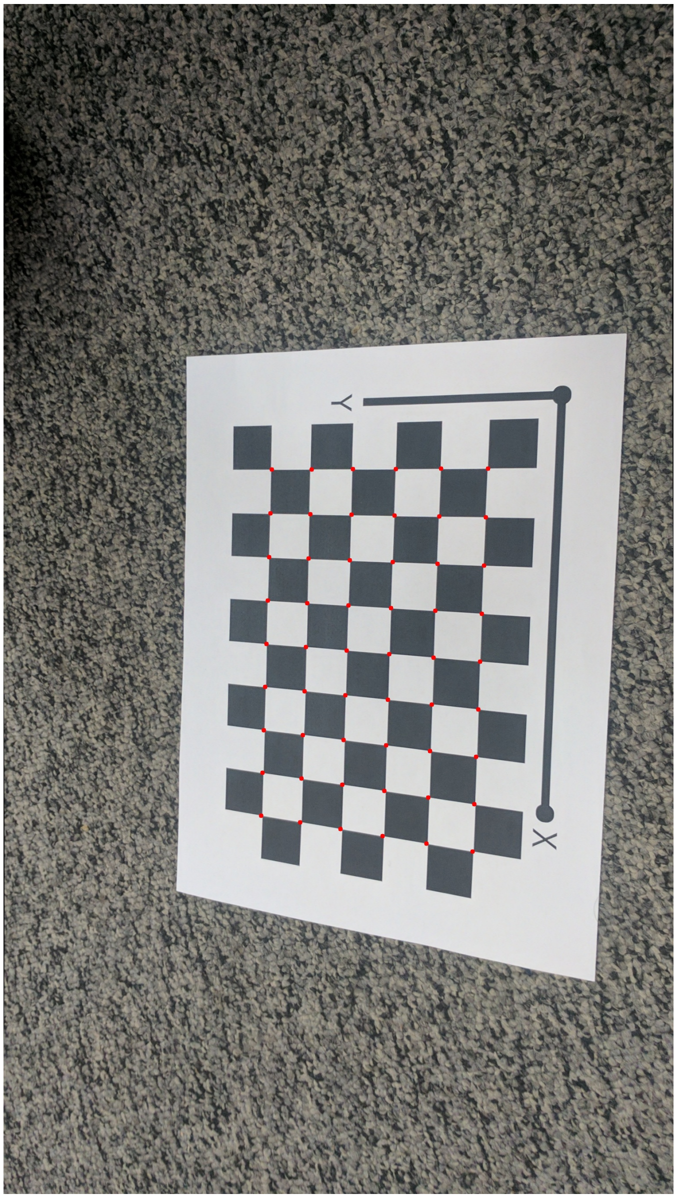

Fig. 14: Reprojected corners on Undistorted Image 10

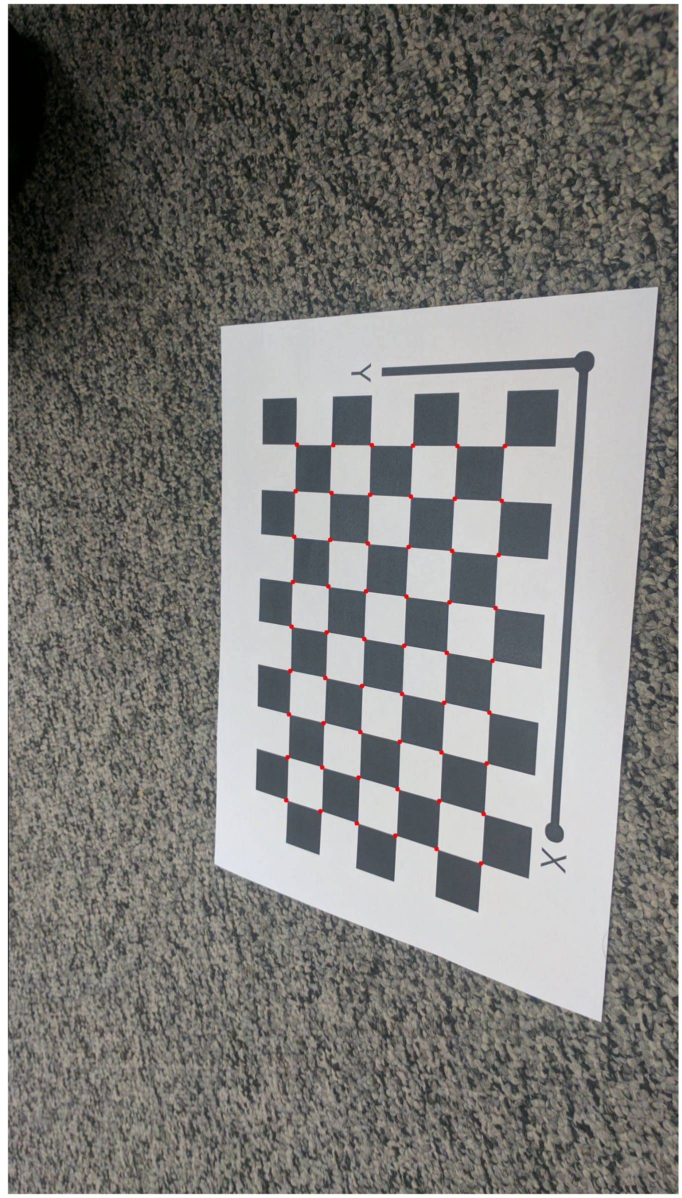

Fig. 15: Reprojected corners on Undistorted Image 11

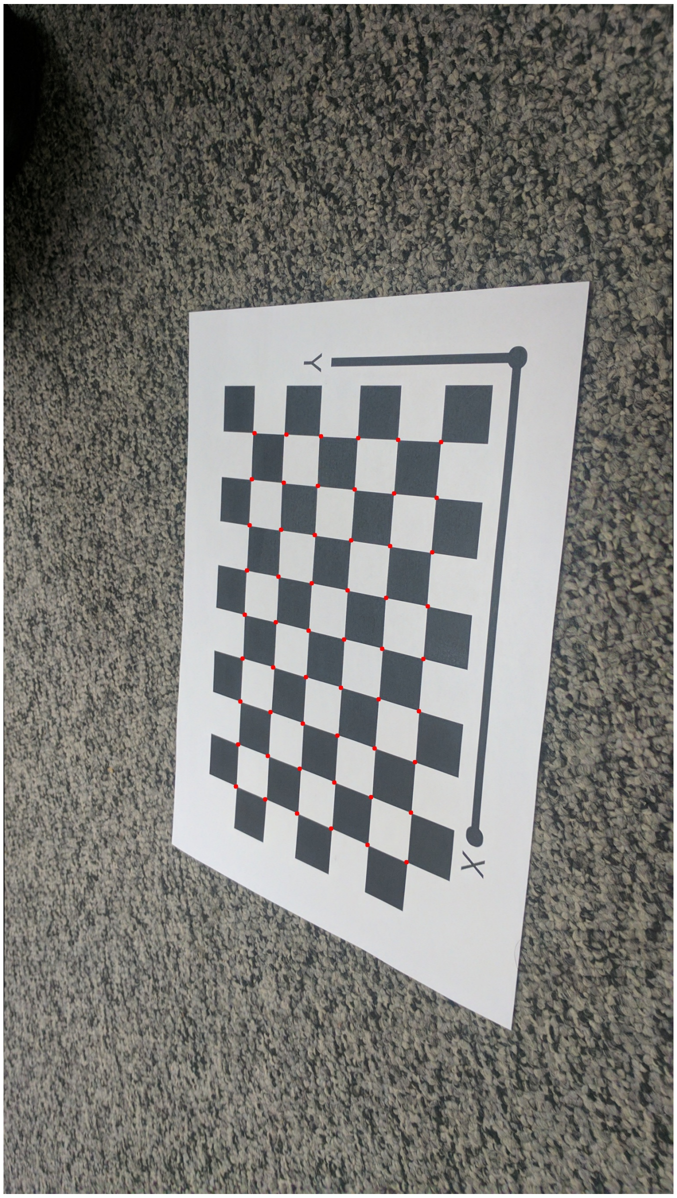

Fig. 16: Reprojected corners on Undistorted Image 12

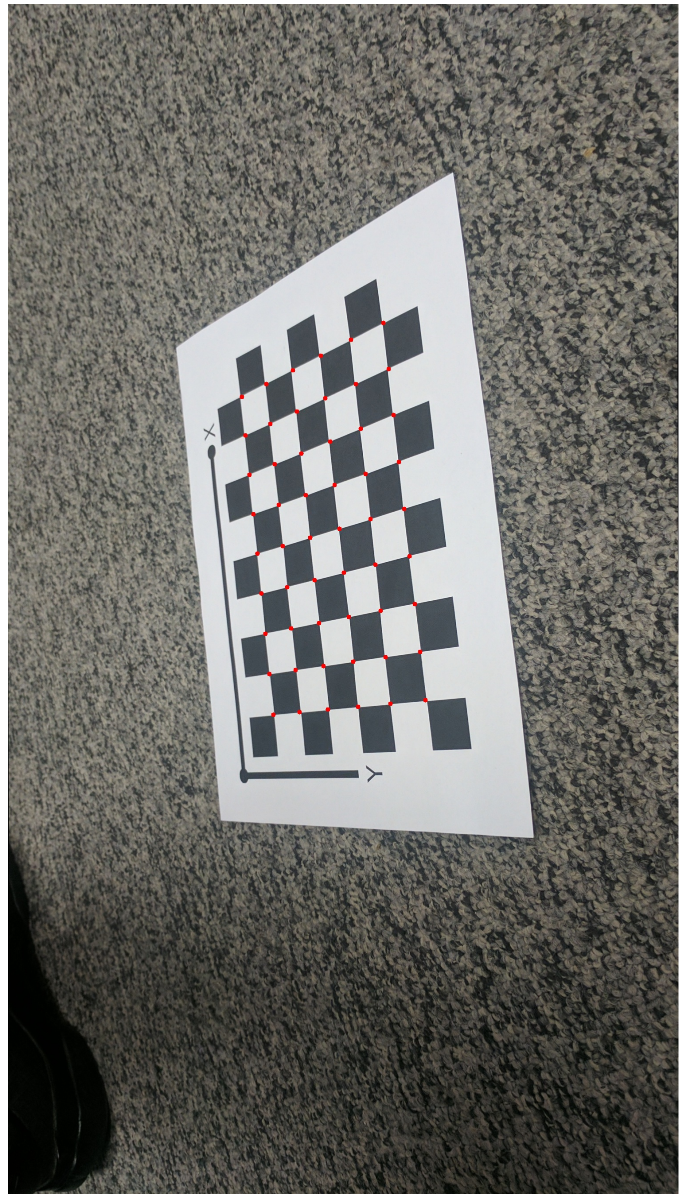

Fig. 17: Reprojected corners on Undistorted Image 13