1 Introduction

In this homework we extract the extrinsic and intrinsic parameters of a camera using Zhang's algorithm. Specifically we do the following,

1. We first use the Canny edge detector to detect edges in the calibration pattern and then use the Hough transform to construct hough lines and detect the corners in the calibration pattern as intersection of the Hough lines.

2. Next, from the detected corners, we calculate the homography between the world coordinates and the image pixel coordinates and use the homography to estimate the intrinsic and extrinsic parameters of the camera using Zhang's algorithm.

3. Lastly, we refine the intrinsic and extrinsic parameters using the Levenberg-Marquardt algorithm.

2 Corner Detection

The corner detection method involves the following steps:

1. First apply the Canny edge detector in MATLAB. The threshold for the detector is 0.8 for the given dataset and 0.7 for my dataset.

2. Next, use the Hough transform to detect the continuous edges in the calibration pattern and identify the vertical and horizontal lines in the calibration pattern. The resolution for ρ used is 0.6 for the given dataset and 0.7 for my dataset for the inbuilt MATLAB function.

3. Lastly we detect the corners of the calibration pattern using homogeneous representation of the detected horizontal and vertical lines.

3 Zhang's Algorithm for Camera Calibration

The following steps are involved the Zhang's algorithm for calculation of intrinsic an extrinsic parameters:

1. To determine the homography H between the image pixel coordinates x_I of the detected corners and the world coordinates x_W of the corners we first need to determine the world coordinates of the corner. The corner coordinates are measured using the ruler where the top left corner has coordinates of (0,0).

2. The homography H satisfies the condition $x_I = Hx_W$ for each image. H is found using linear-least squares estimation as has been discussed in Lecture 11.

3. In the next step, Zhangs algorithm is applied to find the intrinsic parameters. We assume that we have a calibration pattern on the $Z = 0$ plane of the world plane and pictures of calibration pattern are taken from n different viewpoints. Zhang's algorithm is based on the fact that the camera image of an absolute conic is independent of the camera location and orientation and depends only on the intrinsic parameters. The camera image of the absolute conic is given by $w = K^{-T}K^{-1}$.

4. If $H = [\vec{h}_1 \vec{h}_2 \vec{h}_3]$ then we can show that

$$\vec{h}_1^T w \vec{h}_1 = \vec{h}_2^T w \vec{h}_2, \tag{1}$$

and,

$$\vec{h}_1^T w \vec{h}_2 = 0. \tag{2}$$

5. We can reshape matrix w into a vector

$$\vec{b} = \begin{bmatrix} w_{11} & w_{12} & w_{22} & w_{13} & w_{23} & w_{33} \end{bmatrix}^T, \tag{3}$$

and define

$$\vec{V}_{ij} = \begin{bmatrix} h_{i1}h_{j1} & h_{i1}h_{j2} + h_{i2}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1} + h_{i1}h_{j3} & h_{i3}h_{j2} + h_{i2}h_{j3} & h_{i3}h_{j3} \end{bmatrix}^T. \tag{4}$$

6. From this we can derive $\vec{V}_{12}^T \vec{b} = 0$ and $\left(\vec{V}_{11} - \vec{V}_{22}\right)\vec{b} = 0$ so that $V\vec{b} = 0$ where $V = \begin{bmatrix} \vec{V}_{12}^T \\ \left(\vec{V}_{11} - \vec{V}_{22}\right)^T \end{bmatrix}$.

7. For each ith camera position, we can formulate V and stack up V for the total n number of camera positions to give a system of equations $\mathbf{V}\vec{b} = 0$ where $\mathbf{V} = \begin{bmatrix} V_1 \\ V_2 \\ ...V_n \end{bmatrix}$. The system of equations $\mathbf{V}\vec{b} = 0$ can be solved using the Linear-Least squares method discussed in Lecture 11.

8. Once we have calculated $\vec{b}$, we know the image of the absolute conic w. The camera calibration matrix K is found from components of w using,

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

$$x_0 = \frac{w_{12}w_{13} - w_{11}w_{23}}{w_{11}w_{22} - w_{12}^2}, \tag{6}$$

$$\lambda = w_{33} - \frac{w_{13}^2 + x_0\left(w_{12}w_{13} - w_{11}w_{23}\right)}{w_{11}}, \tag{7}$$

$$\alpha_x = \sqrt{\frac{\lambda}{w_{11}}}, \tag{8}$$

$$\alpha_y = \sqrt{\frac{\lambda w_{11}}{w_{11}w_{22} - w_{12}^2}}, \tag{9}$$

$$s = -\frac{w_{12}\alpha_x^2\alpha_y}{\lambda}, \tag{10}$$

$$y_0 = \frac{s x_0}{\alpha_y} - \frac{w_{13}\alpha_x^2}{\lambda}. \tag{11}$$

9. The intrinsic parameter matrix K is the same for all the images but the extrinsic parameters, namely the rotation matrix R and the translation vector $\vec{t}$, need to be calculated for each image individually. R and $\vec{t}$ is calculated using the following equations,

$$R = \begin{bmatrix} \vec{r}_1 & \vec{r}_2 & \vec{r}_3, \end{bmatrix} \tag{12}$$

$$K^{-1} \begin{bmatrix} \vec{h}_1 & \vec{h}_2 & \vec{h}_3 \end{bmatrix} = \begin{bmatrix} \vec{r}_1 & \vec{r}_2 & \vec{t} \end{bmatrix}, \tag{13}$$

$$\begin{aligned} \vec{r}_1 &= \epsilon K^{-1}\vec{h}_1, \\ \vec{r}_2 &= \epsilon K^{-1}\vec{h}_2, \\ \vec{r}_3 &= \vec{r}_1 \times \vec{r}_2, \\ \vec{t} &= \epsilon K^{-1}\vec{h}_3, \end{aligned} \tag{14}$$

where,

$$\epsilon = \frac{1}{||K^{-1}\vec{h}_1||}. \tag{15}$$

4 Refining Calibration Parameters

The algebraic minimization in the Linear-Least Squares solution $\mathbf{V}\vec{b} = 0$ does not provide an intuitive sense of the accuracy of the calculate intrinsic and extrinsic parameters. To get a quantitative estimate of the accuracy of the parameters we calculate the cost as,

$$\sum_i \sum_j ||x_{ij} - K[R_i|\vec{t_i}]x_{M,j}||, \tag{16}$$

where x_{ij} is the location of the jth pixel for the ith image and is actual image point for the point $x_{M,j}$. R_i and $\vec{t_i}$ are the extrinsic parameters for the ith image. To refine the calibration parameters we reduce the cost using the Levenberg-Marquadt (LM) algorithm. The intrinsic and extrinsic parameters of the camera are stacked into $\vec{p} = [K, R_i, t_i | i = 1, 2, ...n]^T$ vector and the LM algorithm is then used to refine these parameters.

Special attention has to be given to the rotation matrix R_i. It has 9 parameters but only 3 DOFs. In any optimization algorithm, the number of variables used to represent an entity must strictly be equal to the number of DOFs of that entity. To get a 3-parameter representation of R_i we use the Rodrgiues representation. The Rodrigues representation is,

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{17}$$

$$\vec{w} = \frac{\phi}{2sin\phi} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}, \tag{18}$$

$$\phi = cos^{-1}\frac{trace(R) - 1}{2}, \tag{19}$$

$$R = I + \frac{sin\phi}{\phi}[\vec{w}]_x + \frac{1 - cos\phi}{\phi^2}[\vec{w}]_x^2, \tag{20}$$

$$[\vec{w}]_x = \begin{bmatrix} 0 & -w_x & w_y \\ w_x & 0 & w_x \\ w_y & w_x & 0 \end{bmatrix}. \tag{21}$$

While calculating R in the LM algorithm we need to make sure that it is orthonormal. This ensured by conditioning the rotation matrix. This is done by finding the best orthonormal match to a given matrix Q by minimizing $||R - Q||_F^2$ where $|| \cdot ||_F$ indicates the Frobenius norm. This is done by carrying out a SVD of Q.

5 Model to Incorporate Radial Distortion

The camera calibration procedure discussed in the previous section is based on the pinhole camera model. However the pinhole camera model fails for short focal-length cameras. Such cameras may image straight lines in world 3D as curves in the image plane. As such we need to account for this radial distortion. A simple model used is,

$$\begin{aligned} \hat{x}_{rad} &= \hat{x} + (\hat{x} - x_0)\,[k_1 r^2 + k_2 r^4], \\ \hat{y}_{rad} &= \hat{y} + (\hat{y} - y_0)\,[k_1 r^2 + k_2 r^4], \end{aligned} \tag{22}$$

where, $\hat{X} = (\hat{x}, \hat{y})$ is the predicted pixel coordinate using the pinhole camera model $\hat{X} = K[R|t]X_W$, $(\hat{x}_{rad}, \hat{y}_{rad})$ is the corrected pixel coordinate after incorporating radial distortion, (x_0, y_0) is the location of the principal point, k_1 and k_2 are model parameters and $r^2 = (x - x_0)^2 + (y - y_0)^2$.

6 Implementation in the Code

The MATLAB code has been implemented in the file **hw8_TejasPant.m**. For this homework, MATLAB has been used instead of Python because I could not get the LM module to work in Python. The following steps are involved in the MATLAB code.

1. We first calculate the world coordinates of the calibration pattern assuming that it is in the $Z = 0$ plane in the world coordinate system. The distance between consecutive corners in the calibration pattern is 25 mm which was determined using a measuring ruler.

2. In the next step we detect the pixel coordinates of the corners in the images of the calibration pattern taken from 40 different viewpoints for Dataset 1 and 20 different viewpoints for Dataset 2. This is done in the function **detectCorners**. In this function the Canny edge detector is used to detect the edges of the calibration pattern and the Hough transform is used to identify the vertical and horizontal lines in the pattern.

3. Next, we calculate the homography that transforms the world coordinates in the $Z = 0$ plane to the image plane using Linear-Least Squares method in function **setupA**. After calculating the homography we now calculate V_{11}, V_{12}, V_{22} required in the Zhang's algorithm using the function **calcVZhangFromHomography**.

4. After setting up **V** for all the images in the dataset we calculate $\vec{b}$ using Linear-Least Squares method.

5. The intrinsic parameters are then calculated from $\vec{b}$ in the function **getIntrinsicParamFromb** and we get the matrix K. The rotation matrix and the translation vector is calculated from K.

6. The extrinsic and intrinsic parameters are then refined using the LM method. The inbuilt MATLAB function **lsqnonlin** is used for the LM method. This function requires a cost function. The cost function is defined using the function **costFunctionLM**.

7. In the last step we carry out the reprojection of the corners using the function **reprojectCorners**.

7 Results

7.1 Intrinsic parameters for given dataset

Intrinsic parameters determined using Linear-Least Squares method,

$$K_{D1} = \begin{bmatrix} 744.6362 & 0.4783 & 232.2006 \\ 0 & 745.0836 & 329.9239 \\ 0 & 0 & 1 \end{bmatrix}. \tag{23}$$

Intrinsic parameters determined using LM method for refining parameters without radial distortion

$$K_{D1} = \begin{bmatrix} 903.5265 & 0.9104 & 334.0564 \\ 0 & 905.2604 & 237.1774 \\ 0 & 0 & 1 \end{bmatrix}. \tag{24}$$

Intrinsic parameters determined using LM method for refining parameters after correcting radial distortion

$$K_{D1} = \begin{bmatrix} 911.8783 & 0.7474 & 340.3366 \\ 0 & 914.2807 & 236.9937 \\ 0 & 0 & 1 \end{bmatrix}. \tag{25}$$

7.2 Extrinsic Parameters for images in given dataset

The corresponding images are at the end of the document.

7.2.1 Image 8 using Linear-Least Square (LLS) and LM Method

$$\begin{bmatrix} R|t \end{bmatrix}_{I8,D1,LLS} = \begin{bmatrix} 0.9201 & 0.0103 & 0.3916 & -21.7580 \\ -0.2071 & 0.8613 & 0.4640 & -175.4626 \\ -0.3325 & -0.5080 & 0.7946 & 627.4433 \end{bmatrix} \tag{26}$$

$$\begin{bmatrix} R|t \end{bmatrix}_{I8,D1,LM} = \begin{bmatrix} 0.9303 & 0.0720 & 0.4028 & -108.0405 \\ -0.2532 & 0.8469 & 0.5393 & -92.7734 \\ -0.3214 & -0.5914 & 0.8005 & 723.5021 \end{bmatrix} \tag{27}$$

7.2.2 Image 12 using Linear-Least Square (LLS) and LM Method

$$\begin{bmatrix} R|t \end{bmatrix}_{I12,D1,LLS} = \begin{bmatrix} 0.8989 & 0.2063 & -0.3865 & -2.8257 \\ -0.0328 & 0.9114 & 0.4103 & -175.9412 \\ 0.4369 & -0.3562 & 0.8260 & 520.3756 \end{bmatrix} \tag{28}$$

$$\begin{bmatrix} R|t \end{bmatrix}_{I12,D1,LM} = \begin{bmatrix} 0.8748 & 0.2306 & -0.4845 & -83.6923 \\ -0.0146 & 0.8469 & 0.4942 & -92.7734 \\ 0.5364 & -0.4374 & 0.7831 & 723.5021 \end{bmatrix} \tag{29}$$

7.2.3 Image 23 using Linear-Least Square (LLS) and LM Method

$$[R|t]_{I23,D1,LLS} = \begin{bmatrix} 0.9751 & -0.1546 & -0.1587 & -11.1342 \\ 0.1413 & 0.9857 & -0.0921 & -172.2514 \\ 0.1706 & 0.0674 & 0.9830 & 421.8506 \end{bmatrix} \tag{30}$$

$$[R|t]_{I23,D1,LM} = \begin{bmatrix} 0.9866 & -0.1703 & -0.2359 & -75.1463 \\ 0.1637 & 0.9944 & -0.0945 & -123.4032 \\ 0.2406 & 0.0819 & 0.9898 & 527.8999 \end{bmatrix} \tag{31}$$

7.2.4 Image 32 using Linear-Least Square (LLS) and LM Method

$$[R|t]_{I32,D1,LLS} = \begin{bmatrix} 0.9698 & -0.2420 & -0.0296 & 5.0029 \\ 0.2383 & 0.9667 & -0.0936 & -184.8763 \\ 0.0512 & 0.0837 & 0.9952 & 450.4250 \end{bmatrix} \tag{32}$$

$$[R|t]_{I32,D1,LM} = \begin{bmatrix} 0.9899 & -0.2555 & -0.0584 & -62.0189 \\ 0.2535 & 0.9887 & -0.1128 & -132.9223 \\ 0.0667 & 0.1081 & 0.9976 & 565.3600 \end{bmatrix} \tag{33}$$

7.3 Intrinsic parameters for my dataset

Intrinsic parameters determined using Linear-Least Squares

$$K_{D2} = \begin{bmatrix} 650.4075 & -2.6376 & 425.1661 \\ 0 & 652.8432 & 314.1275 \\ 0 & 0 & 1 \end{bmatrix} \tag{34}$$

Intrinsic parameters determined using LM method for refining parameters without radial distortion

$$K_{D2} = \begin{bmatrix} 883.4563 & -3.8922 & 326.9158 \\ 0 & 880.2676 & 414.5967 \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

Intrinsic parameters determined using LM method for refining parameters after correcting radial distortion

$$K_{D2} = \begin{bmatrix} 873.1869 & -4.3785 & 319.7347 \\ 0 & 869.4459 & 419.9473 \\ 0 & 0 & 1 \end{bmatrix} \tag{36}$$

7.4 Extrinsic Parameters for images in my dataset

The corresponding images are at the end of the document.

7.4.1 Image 2 using Linear-Least Square (LLS) and LM Method

$$[R|t]_{I2,D2,LLS} = \begin{bmatrix} 0.9145 & -0.0433 & -0.4023 & -151.1128 \\ 0.0097 & 0.9963 & -0.0852 & -79.2746 \\ 0.4045 & 0.0740 & 0.9115 & 588.9099 \end{bmatrix} \tag{37}$$

$$[R|t]_{I2,D2,LM} = \begin{bmatrix} 0.9038 & -0.0041 & -0.5476 & -51.7963 \\ -0.0273 & 0.9974 & -0.0871 & -162.1407 \\ 0.5470 & 0.0912 & 0.9013 & 730.4336 \end{bmatrix} \tag{38}$$

7.4.2 Image 5 using Linear-Least Square (LLS) and LM Method

$$\begin{bmatrix}R|t\end{bmatrix}_{I5,D2,LLS} = \begin{bmatrix} 0.9510 & 0.0238 & 0.3084 & -205.8766 \\ -0.0180 & 0.9996 & -0.0215 & -5.7493 \\ -0.3088 & 0.0149 & 0.9510 & 452.7700 \end{bmatrix} \tag{39}$$

$$\begin{bmatrix}R|t\end{bmatrix}_{I5,D2,LM} = \begin{bmatrix} 0.9428 & 0.0039 & 0.4685 & -139.1032 \\ 0.0075 & 0.9994 & -0.0465 & -83.1545 \\ -0.4685 & 0.0470 & 0.9423 & 640.0377 \end{bmatrix} \tag{40}$$

7.4.3 Image 12 using Linear-Least Square (LLS) and LM Method

$$\begin{bmatrix}R|t\end{bmatrix}_{I12,D2,LLS} = \begin{bmatrix} 0.9737 & 0.0171 & 0.2271 & -259.6932 \\ 0.0240 & 0.9838 & -0.1774 & 20.4336 \\ -0.22655 & 0.1782 & 0.9575 & 512.4753 \end{bmatrix} \tag{41}$$

$$\begin{bmatrix}R|t\end{bmatrix}_{I12,D2,LM} = \begin{bmatrix} 0.9671 & 0.0176 & 0.3613 & -182.7453 \\ 0.0340 & 0.9796 & -0.2827 & -65.0896 \\ -0.3602 & 0.2842 & 0.9468 & 715.1872 \end{bmatrix} \tag{42}$$

7.4.4 Image 16 using Linear-Least Square (LLS) and LM Method

$$\begin{bmatrix}R|t\end{bmatrix}_{I16,D2,LLS} = \begin{bmatrix} 0.9631 & 0.08831 & 0.2541 & -248.3683 \\ -0.0737 & 0.9950 & -0.0663 & 29.9202 \\ -0.2587 & 0.0452 & 0.9648 & 537.5123 \end{bmatrix} \tag{43}$$

$$\begin{bmatrix}R|t\end{bmatrix}_{I16,D2,LM} = \begin{bmatrix} 0.9552 & 0.0770 & 0.4232 & -167.4934 \\ -0.0555 & 0.9962 & -0.1115 & -59.1197 \\ -0.4266 & 0.0978 & 0.9536 & 752.0862 \end{bmatrix} \tag{44}$$

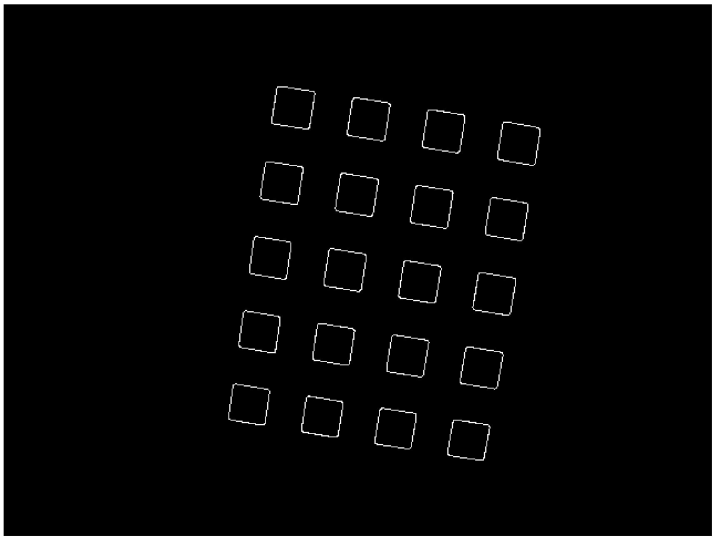

Figure 1: Edges detected using Canny Edge Detector for **Image 5** of **Dataset 1**.

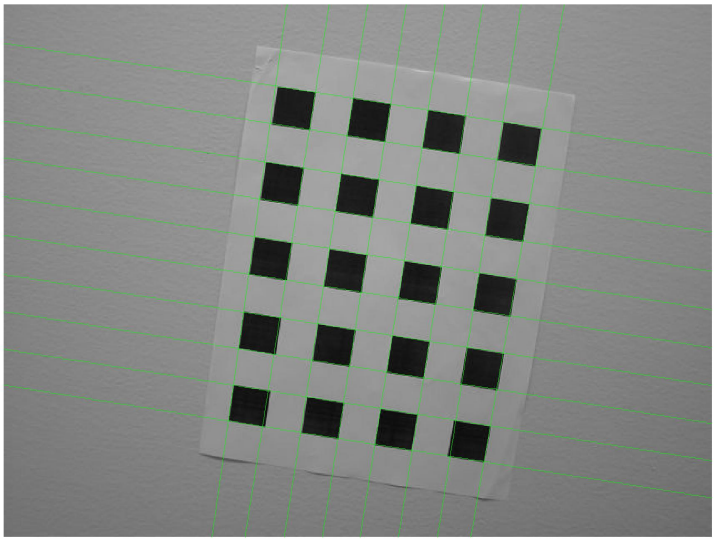

Figure 2: Lines detected using Hough Line Transform for **Image 5** of **Dataset 1**.

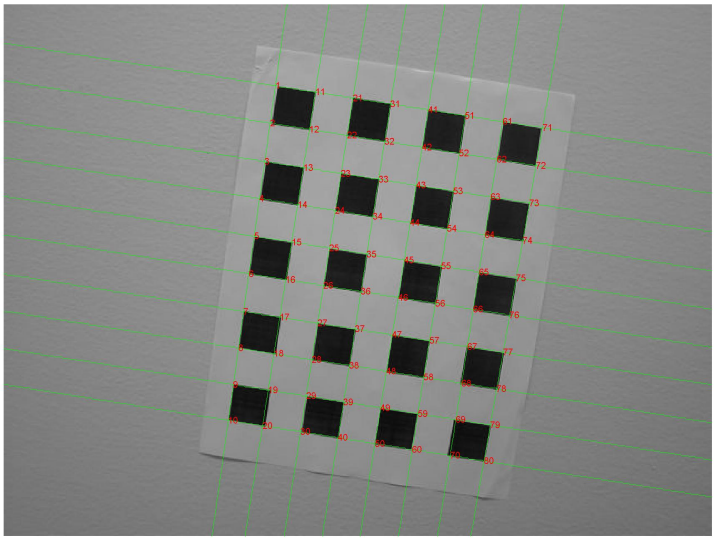

Figure 3: Detected Corners for **Image 5** of **Dataset 1**.

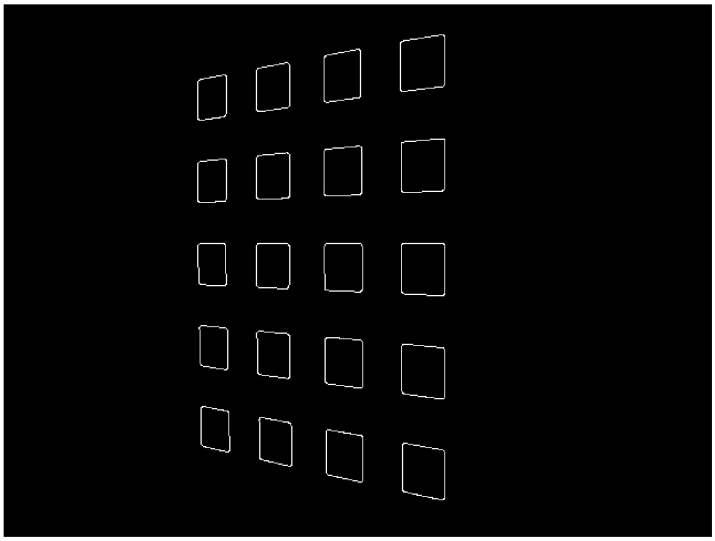

Figure 4: Edges detected using Canny Edge Detector for **Image 20** of **Dataset 1**.

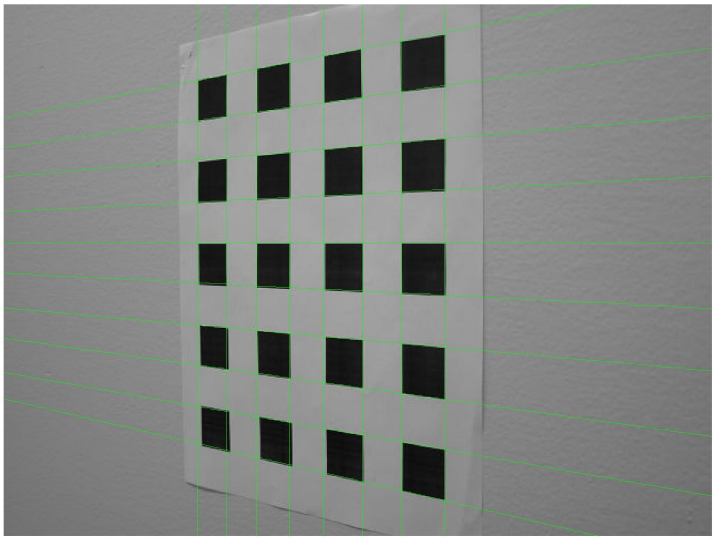

Figure 5: Lines detected using Hough Line Transform for **Image 20** of **Dataset 1**.

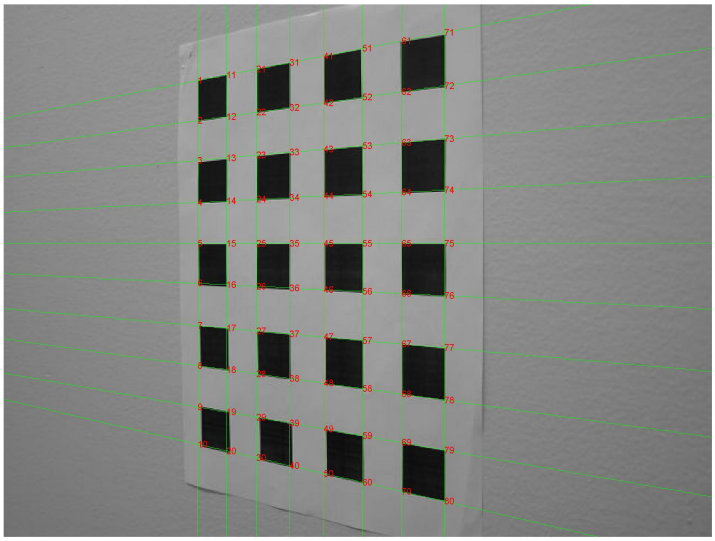

Figure 6: Detected Corners for **Image 20** of **Dataset 1**.

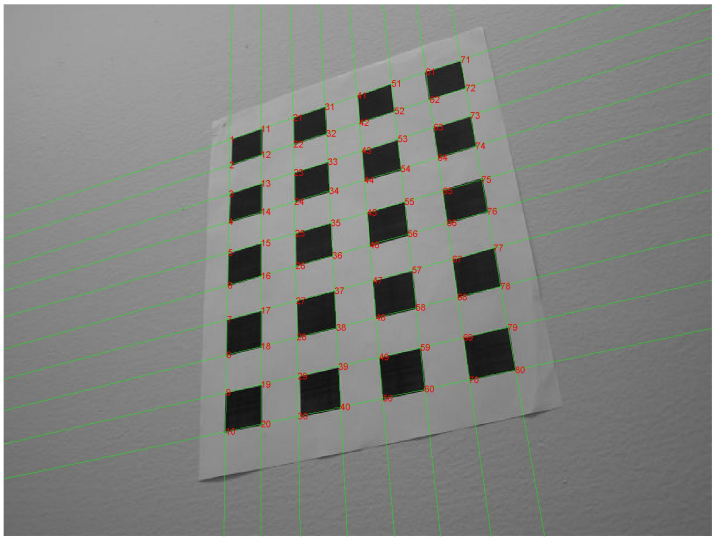

Figure 7: Extrinsic Parameters for **Image 8** of **Dataset 1**.

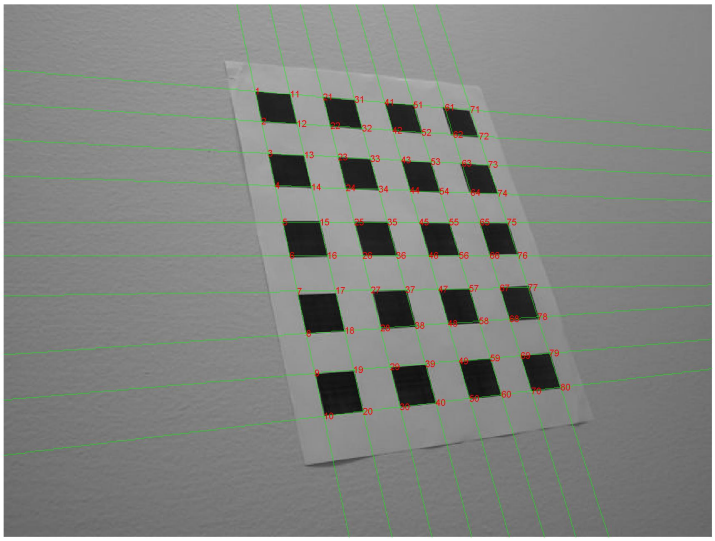

Figure 8: Extrinsic Parameters for **Image 12** of **Dataset 1**.

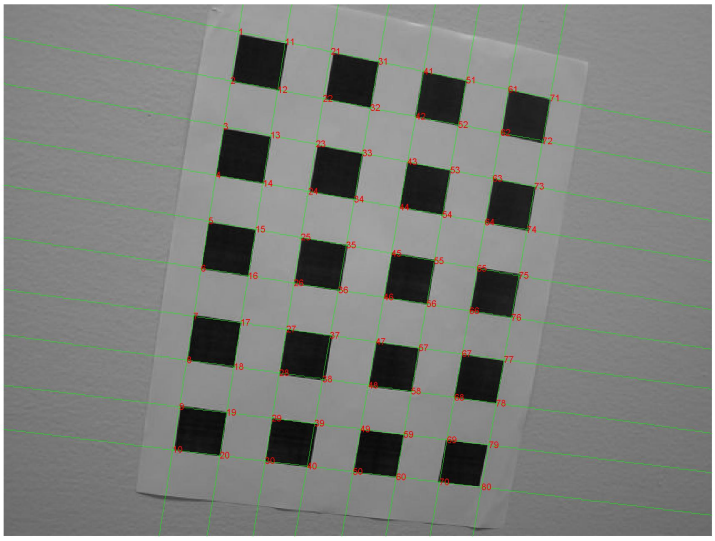

Figure 9: Extrinsic Parameters for **Image 23** of **Dataset 1**.

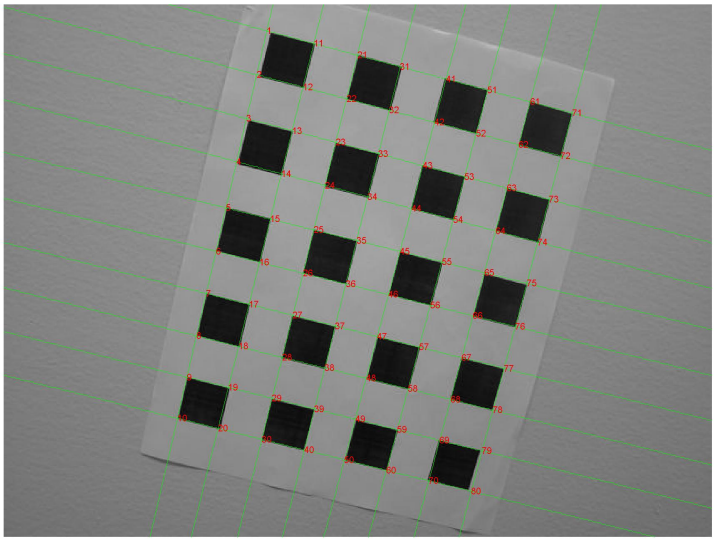

Figure 10: Extrinsic Parameters for **Image 32** of **Dataset 1**.

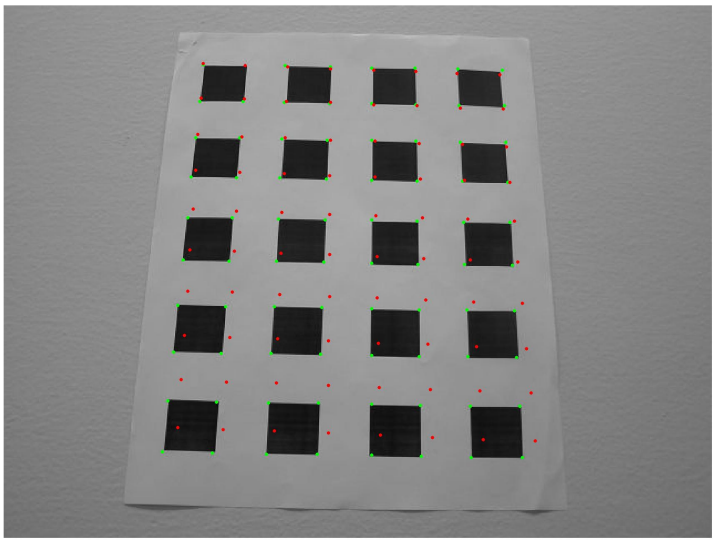

Figure 11: Reprojecting corners of the fixed Image, **Image 26**, using **Image 13** for **Dataset 1 before refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 6.2577, Variance of Error = 26.1524.

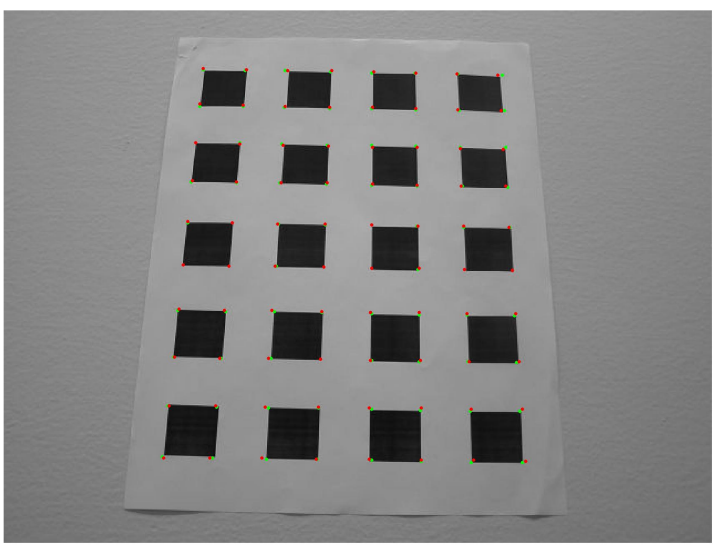

Figure 12: Reprojecting corners of the fixed Image, **Image 26**, using **Image 13** for **Dataset 1 after refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 0.8150, Variance of Error = 0.4404.

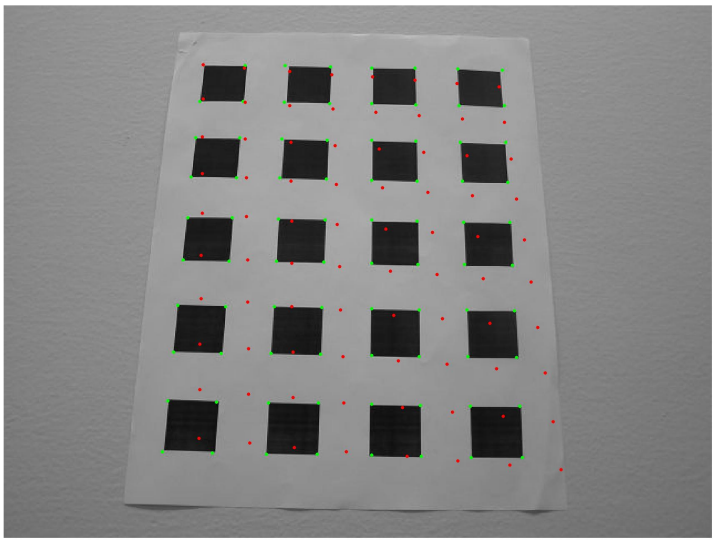

Figure 13: Reprojecting corners of the fixed Image, **Image 26**, using **Image 21** for **Dataset 1 before refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 11.0584, Variance of Error = 78.1149.

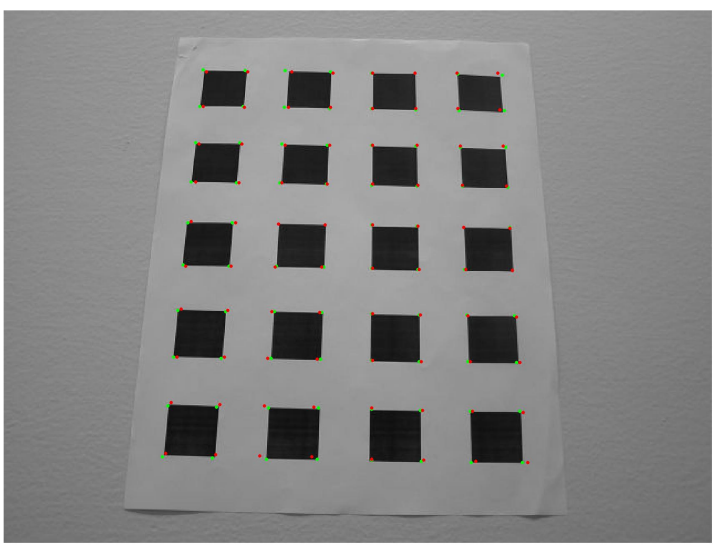

Figure 14: Reprojecting corners of the fixed Image, **Image 26**, using **Image 21** for **Dataset 1 after refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 1.4077, Variance of Error = 1.5652.

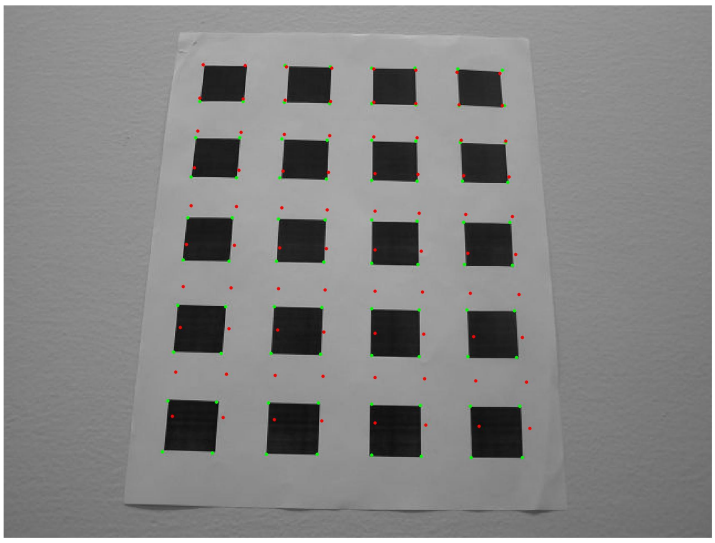

Figure 15: Reprojecting corners of the fixed Image, **Image 26**, using **Image 23** for **Dataset 1 before refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 7.6912, Variance of Error = 67.8816.

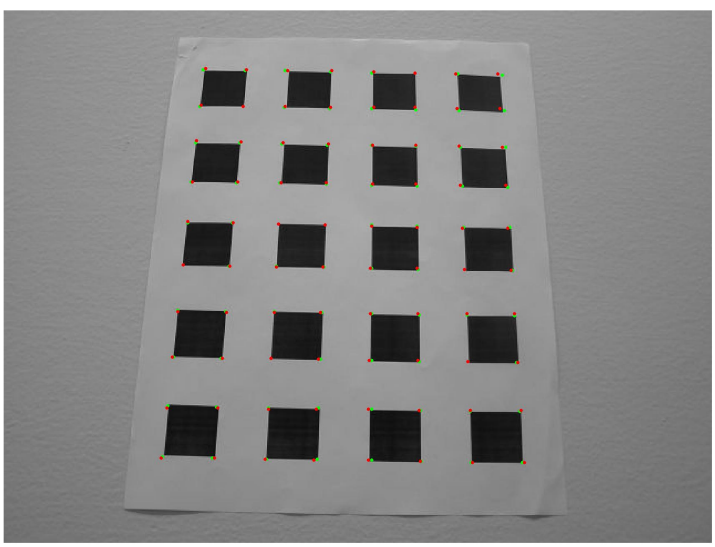

Figure 16: Reprojecting corners of the fixed Image, **Image 26**, using **Image 23** for **Dataset 1 after refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 0.8724, Variance of Error = 0.5363.

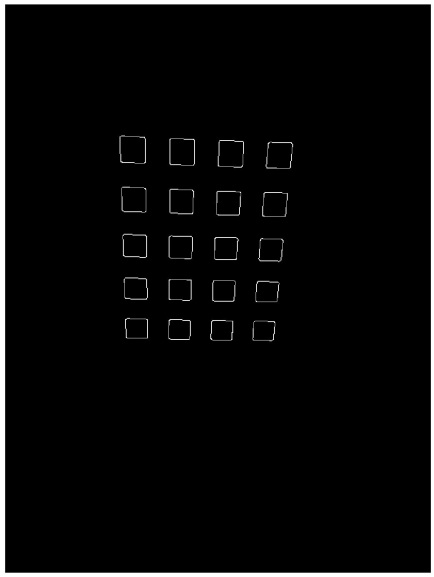

Figure 17: Edges detected using Canny Edge Detector for **Image 3** of **Dataset 2**.

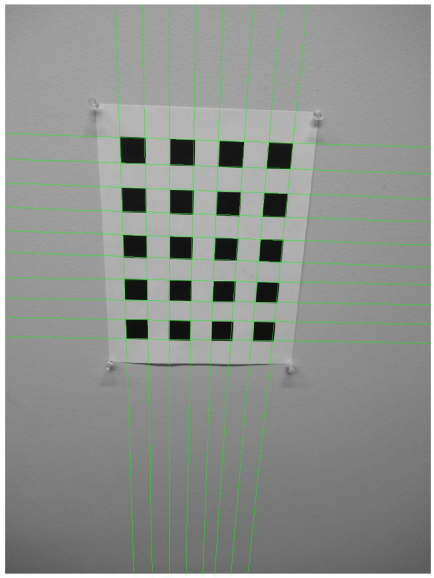

Figure 18: Lines detected using Hough Line Transform for **Image 3** of **Dataset 2**.

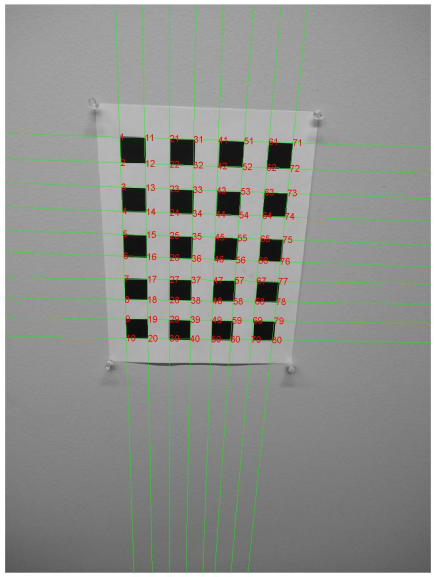

Figure 19: Detected Corners for **Image 3** of **Dataset 2**.

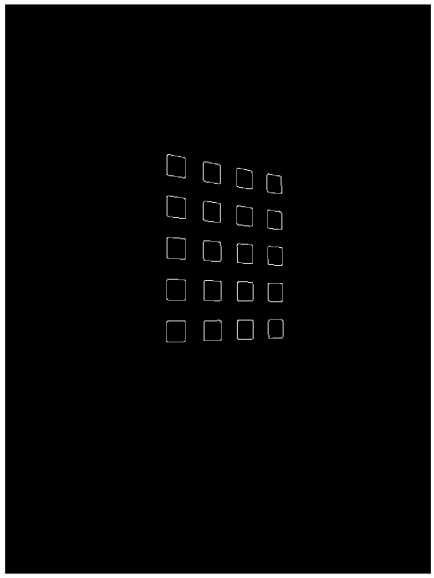

Figure 20: Edges detected using Canny Edge Detector for **Image 10** of **Dataset 2**.

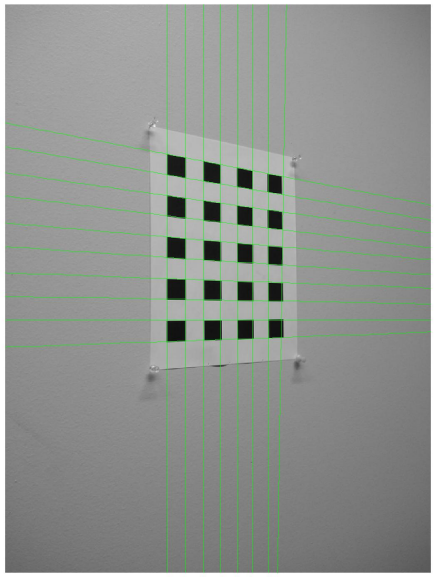

Figure 21: Lines detected using Hough Line Transform for **Image 10** of **Dataset 2**.

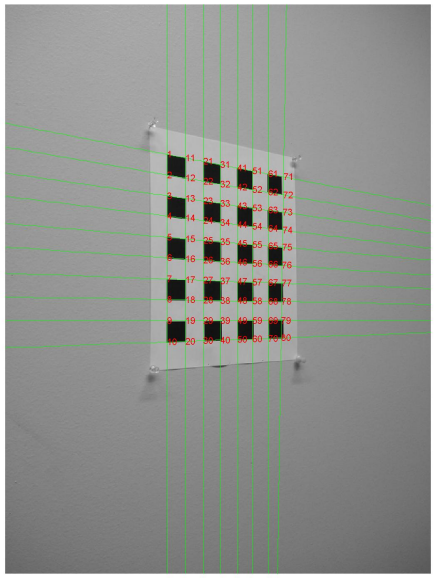

Figure 22: Detected Corners for **Image 10** of **Dataset 2**.

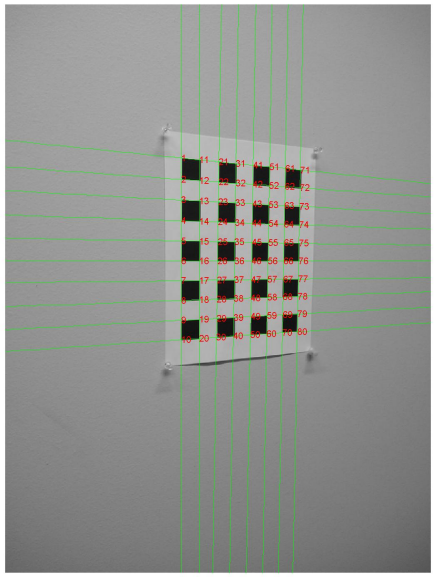

Figure 23: Extrinsic Parameters for **Image 2** of **Dataset 2**.

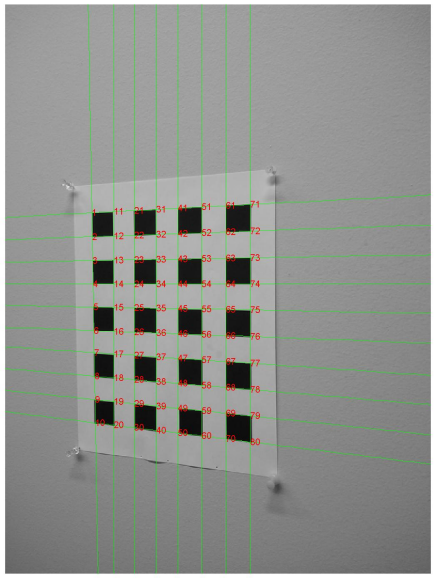

Figure 24: Extrinsic Parameters for **Image 5** of **Dataset 2**.

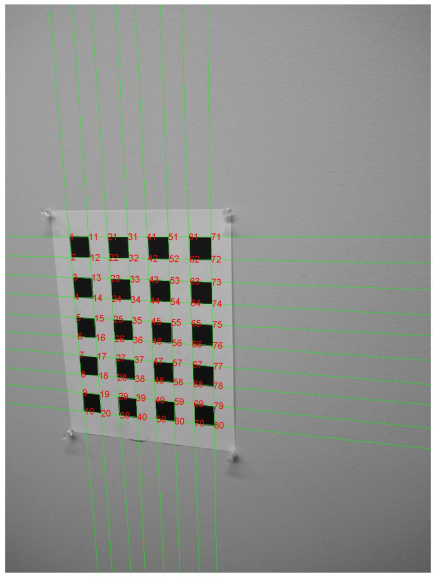

Figure 25: Extrinsic Parameters for **Image 12** of **Dataset 2**.

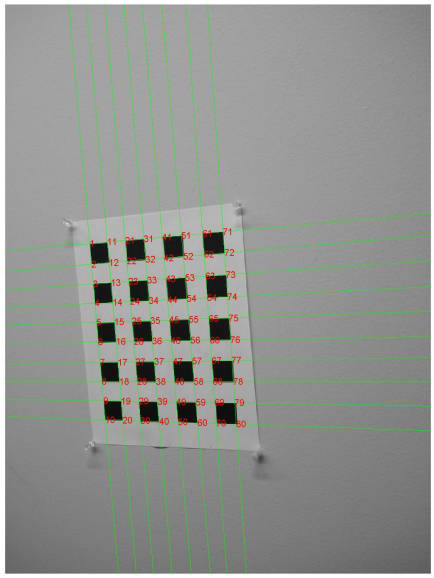

Figure 26: Extrinsic Parameters for **Image 16** of **Dataset 2**.

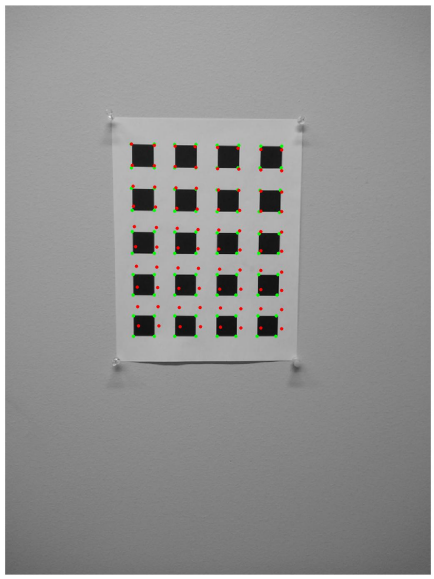

Figure 27: Reprojecting corners of the fixed Image, **Image 1**, using **Image 4** for **Dataset 2 before refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 4.9369, Variance of Error = 13.4332.

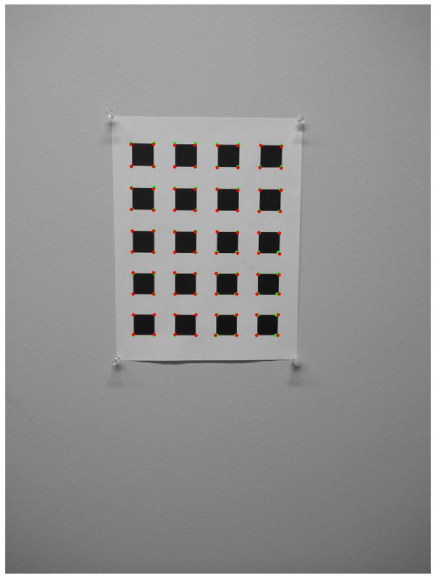

Figure 28: Reprojecting corners of the fixed Image, **Image 1**, using **Image 4** for **Dataset 2 after refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 0.7430, Variance of Error = 0.4161.

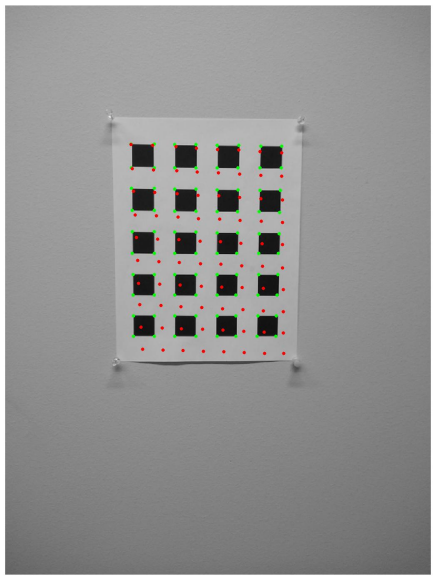

Figure 29: Reprojecting corners of the fixed Image, **Image 1**, using **Image 8** for **Dataset 2 before refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 9.9944, Variance of Error = 43.5005.

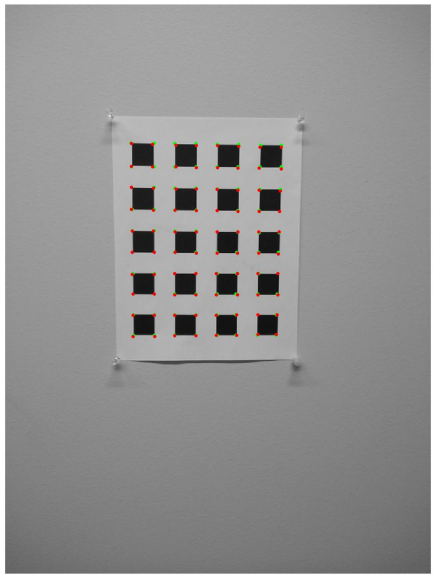

Figure 30: Reprojecting corners of the fixed Image, **Image 1**, using **Image 8** for **Dataset 2 after refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 0.8222, Variance of Error = 0.4554.

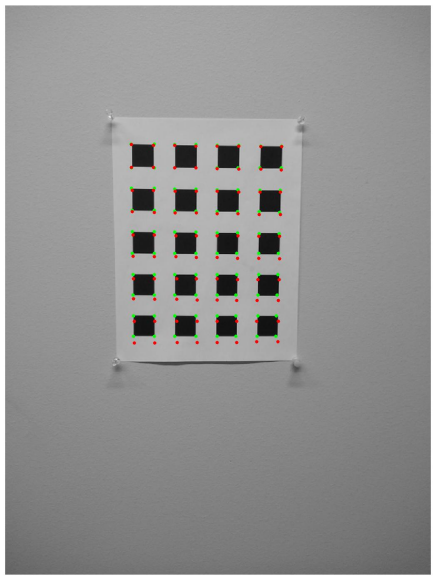

Figure 31: Reprojecting corners of the fixed Image, **Image 1**, using **Image 13** for **Dataset 2 before refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 2.9377, Variance of Error = 7.9479.

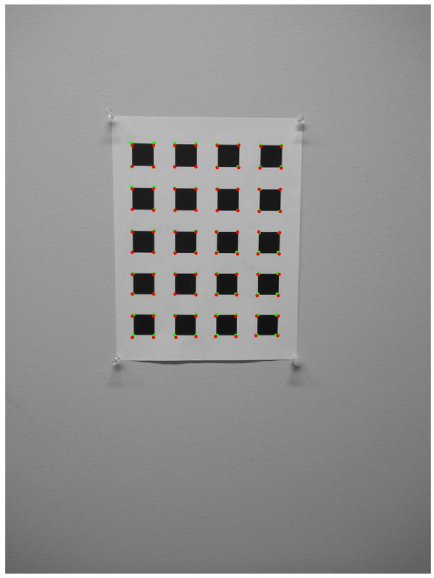

Figure 32: Reprojecting corners of the fixed Image, **Image 1**, using **Image 13** for **Dataset 2 after refining with LM method**. The red dots indicate the reprojected corners and the green dots are the original corners. Mean Error = 1.0403, Variance of Error = 0.7404.