Namrata Vivek Raghavan
nrgahav@purdue.edu

Zhang's Algorithm For Camera Calibration

In the following assignment we are required to implement the Zhang's algorithm for camera calibration. We make the assumption that the camera is a pinhole camera. Thus, we are required to find 5 intrinsic parameters and the 6 intrinsic parameters that determine the position and the orientation of the camera with respect to a reference world coordinate system. Lastly, we are required to optimize the parameters using the Levenberg Marquardt algorithm.

Procedure for extracting intrinsic and extrinsic parameters of the camera:

1. Detection and extraction of corners:

The following are the steps to extract the corners from every image in each dataset:

a) First use the OpenCv implementation of the Canny edge detector on the gray scale image of the calibration patterns in order to find the edges.

b) Next apply the OpenCv implementation of the Hough transform to determine the Hough lines in the image. The transform returns a large number of Hough lines, this can be reduced using maximal suppression.

c) The intersection of the lines is calculated to determine the corner points in the image. The corner points are label from left to right and from top to bottom.

The world coordinates of the corner points in the calibration pattern are determine by assuming that each square in the pattern is of unit dimension.

2. Calculation of intrinsic and extrinsic parameters

Once the corners are detected in every image in the data set following are the steps to extract the intrinsic and extrinsic parameters:

a) Find the homography from the world coordinates to the corners in each image in the dataset. Thus, the image pixels in the images in the data set is given by x' such that

$$x' = Hx_{world\ coordinates}$$

$$\begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \\ \vdots \\ x'_n \\ y'_n \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix}$$

Thus, $H = B^+A$

$$B = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 \end{bmatrix}, A = \begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \\ \vdots \\ x'_n \\ y'_n \end{bmatrix}$$

Namrata Vivek Raghavan
nrgahav@purdue.edu

b) Once the homographes are calculated implement the Zhang's algorithm to calculate the intrinsic parameters K and the extrinsic parameters R,t of the camera. The Zhang's algorithm makes the assumption that the calibration pattern is on the Z = 0 plane and the images in the data set as pictures of the calibration pattern taken at different viewpoints of the camera. The image of the absolute conic is given by $\omega = K^{-T}K^{-1}$. The image of the absolute conic can be calculated using the following procedure:

i. $h_1^T \omega h_1 = h_2^T \omega h_2$ and $h_1^T \omega h_2 = 0$, where ω is the image of the absolute conic and h_i is the i[th] row of the homography matrix H. Let V be a matrix defined as

$$V = \begin{bmatrix} \overrightarrow{v_{12}}^T \\ \overrightarrow{v_{11} - v_{22}}^T \end{bmatrix}$$

Where $\overrightarrow{v_{ij}} = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j1} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix}$. Let b be vector given by $\vec{b} \begin{bmatrix} \omega_{11} \\ \omega_{12} \\ \omega_{22} \\ \omega_{13} \\ \omega_{23} \\ \omega_{33} \end{bmatrix}$, where ω_{ij} are the elements in the image of the absolute conic ω. It is found that $Vb = 0$. Thus ω can be calculated by solving for b which is found using SVD as the eigen vector corresponding to the smallest eigen value.

c)Once the image of the absolute conic is calculated following is the procedure to calculate the intrinsic parameters of the camera is given by the following matrix $K = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$,

where $x_0 = \frac{\omega_{12} - \omega_{13} - \omega_{11}\omega_{23}}{\omega_{11}\omega_{22} - \omega_{12}^2}$, $\lambda = \omega_{33} - \frac{\omega_{13}^2 + x_0(\omega_{12} - \omega_{13} - \omega_{11}\omega_{23})}{\omega_{11}}$, $\alpha_x = \sqrt{\frac{\lambda}{\omega_{11}}}$,

$\alpha_y = \sqrt{\frac{\lambda\omega_{11}}{\omega_{11}\omega_{22} - \omega_{12}^2}}$, $s = -\frac{\omega_{12}\,\alpha_x^2\,\alpha_y}{\lambda}$, $y_0 = \frac{sx_0}{\alpha_y} - \frac{\omega_{13}\,\alpha_x^2}{\lambda}$.

d) Once the intrinsic parameters of the camera are calculated the extrinsic parameter for each camera viewpoint can be calculated using the following formulas:

i. $R = [r_1\ r_2\ r_3]$ where $r_1 = \epsilon K^{-1}h_1, r_2 = \epsilon K^{-1}h_2,\ r_3 = r_1 X\ r_2$
ii. $t = \epsilon K^{-1}h_3$
iii. $\epsilon = \frac{1}{\|r_1\|}$

3. Refining the Intrinsic and Extrinsic parameters are refined using the LM algorithm

Once the intrinsic and extrinsic parameters are calculated for each image in the dataset the LM algorithm is used to refining the projection matrix P for each image in the dataset. Since the Zhang's algorithm makes the assumption that the calibration pattern is on the Z = 0 plane the projection matrix is given by $P = K[r_1\ r_2\ t]$. Thus, the image pixels of the projected image x' is given by $x' = Px$, where x = [x y 1].

Namrata Vivek Raghavan
nrgahav@purdue.edu

The LM algorithm combines both the gradient descent and gauss newton methods. In the LM algorithm the step size in each iteration is given by the formula

$$\vec{\delta}_p = \left(J_{\vec{f}}^T J_{\vec{f}} + \mu I\right)^{-1} J_{\vec{f}}^T \vec{\epsilon}(\overrightarrow{p_k})$$

, $\vec{\epsilon}(\overrightarrow{p_k}) = \|X'_{actual} - F(P_k)\|$ is the error function, where P(k) is the projection matrix estimation $[p_{11}p_{12}p_{13}p_{21}p_{22}p_{23}p_{31}p_{32}p_{33}]$ and $F(P_k)$ is a vector of i points $\begin{bmatrix} x'_i \\ y'_i \end{bmatrix}$

$x' = \frac{p_{11}x + p_{12}y + p_{13}}{p_{31}x + p_{32}y + 1}$ and $y' = \frac{p_{21}x + p_{22}y + p_{23}}{p_{31}x + p_{32}y + 1}$. $J_{\vec{f}}$ is the Jacobian matrix of $\vec{\epsilon}(\overrightarrow{p_k})$ given by $\frac{\delta F(P_k)}{\delta P_k}$. μ, is the damping coefficient.

The orthonormality of the r1 and r2 are determined by dividing them by the norm of r1. In the following experiment I have implemented the LM algorithm to optimize the projection matrix using the scipy.optimize.root library. In order to implement the library, I had to create a function that would calculate the error as well as the Jacobian matrix which was then used as an input to the library.

4. Method for reprojecting corners onto a Fixed image

1. Choose one image from the data set as the fixed image. In this experiment I have chosen image 11 as the fixed image in dataset 1 and image 6 as the fixed image in dataset 2.
2. Calculate the homographes between the fixed image and the other images in the dataset.
3. Calculate the intrinsic and extrinsic parameters using the procedures described above.
4. Multiply the corners in an image with the inverse of the projection matrix so as it finds it mapped version in the fixed image.
5. Plot the mapped version of the corners onto the fixed image

5. Results

Dataset 1:

Intrinsic camera matrix:

$$K = \begin{bmatrix} 782.7596 & 0.7489488 & 229.0341 \\ 0 & 768.947 & 306.8278 \\ 0 & 0 & 1 \end{bmatrix}$$

Output images for edge detection

Namrata Vivek Raghavan
nrgahav@purdue.edu

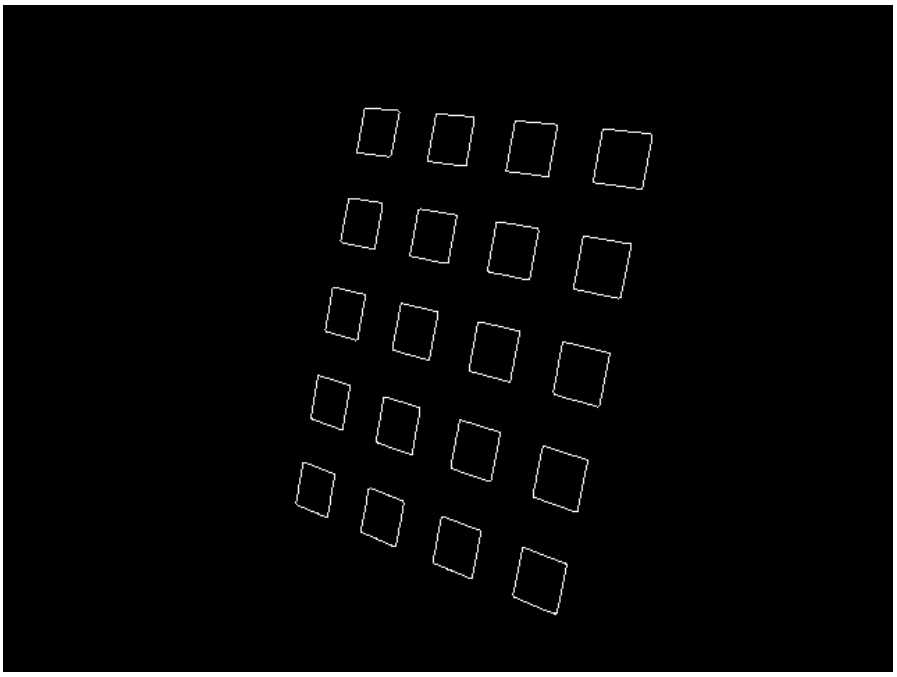

Figure 1: Canny Edge Detection Output for image 1

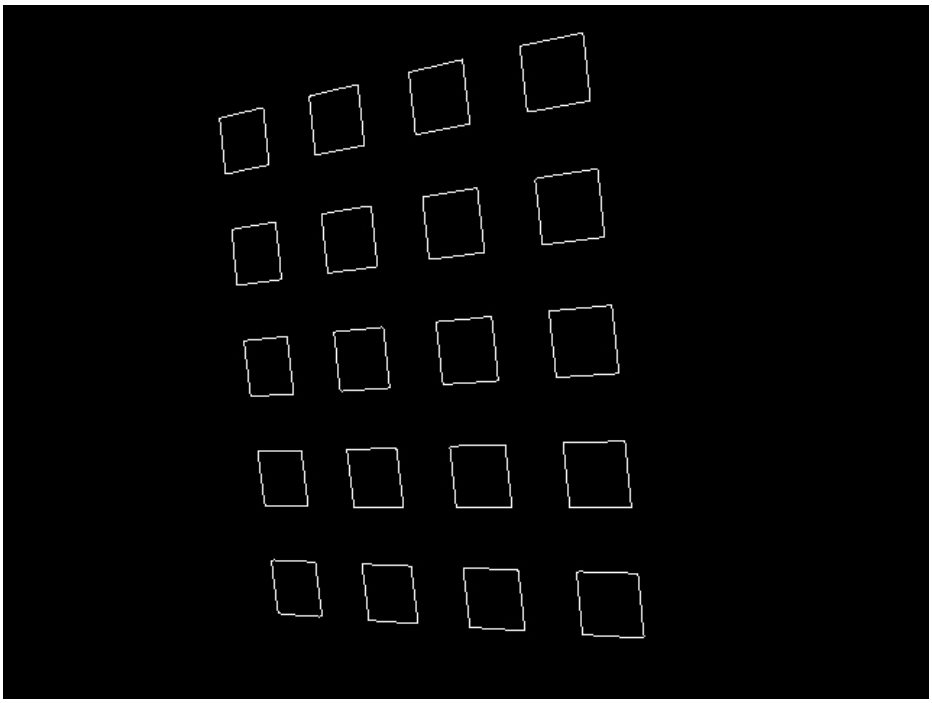

Figure 2: Canny Edge Detection Output for image 19

Output images for hough-lines detection

Namrata Vivek Raghavan
nrgahav@purdue.edu

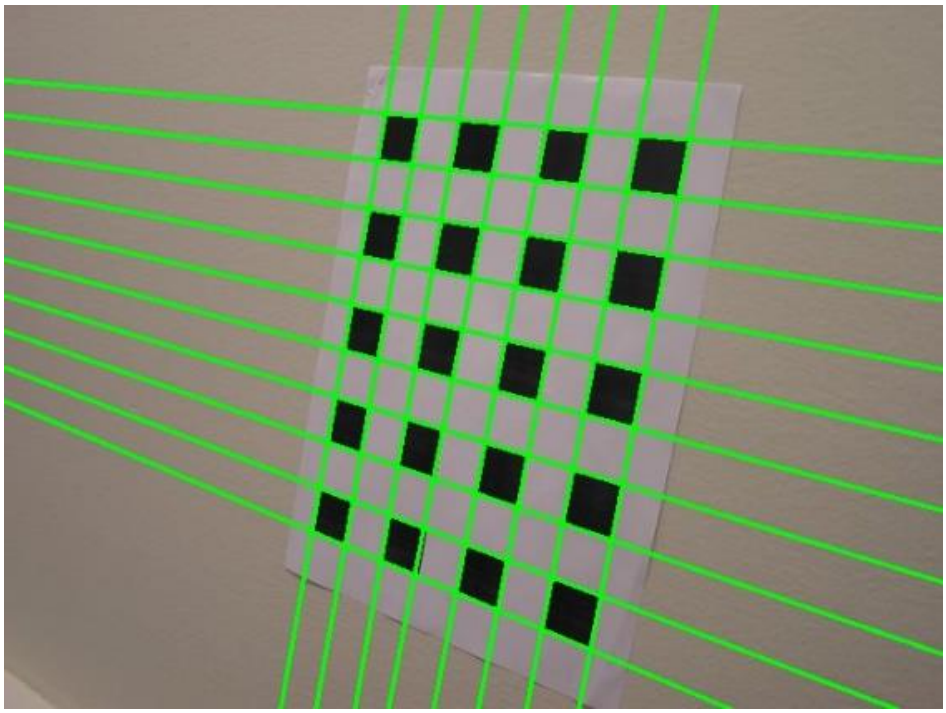

Figure 3: Hough-line detected for image 1

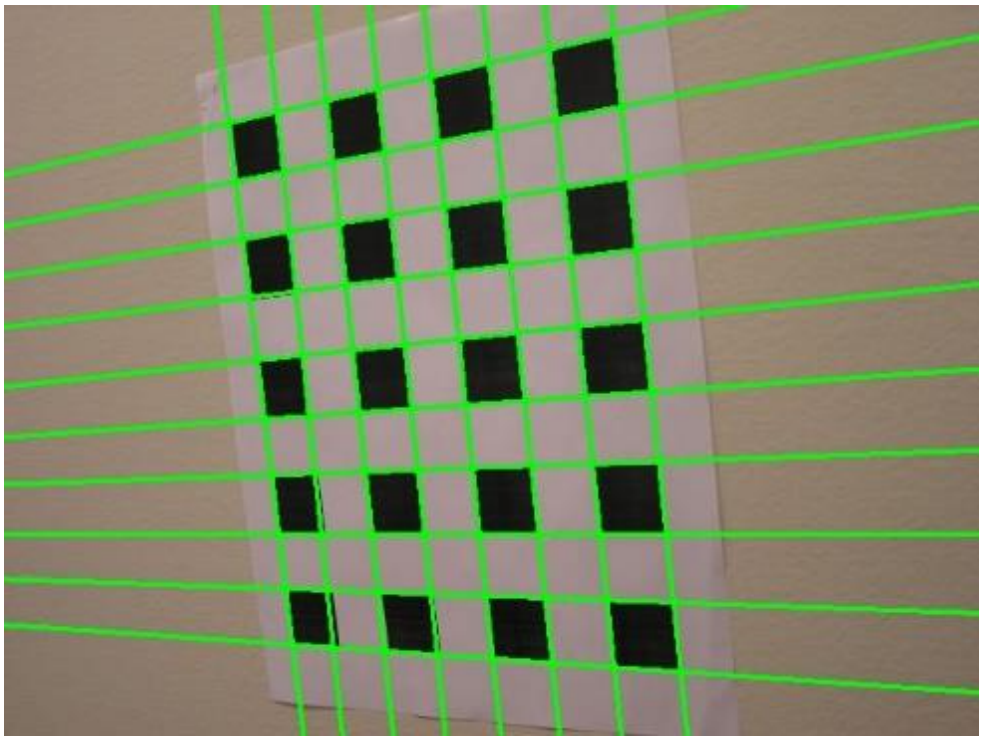

Figure 4: Hough-line detected for image 19

Output images for corner detection

Namrata Vivek Raghavan
nrgahav@purdue.edu

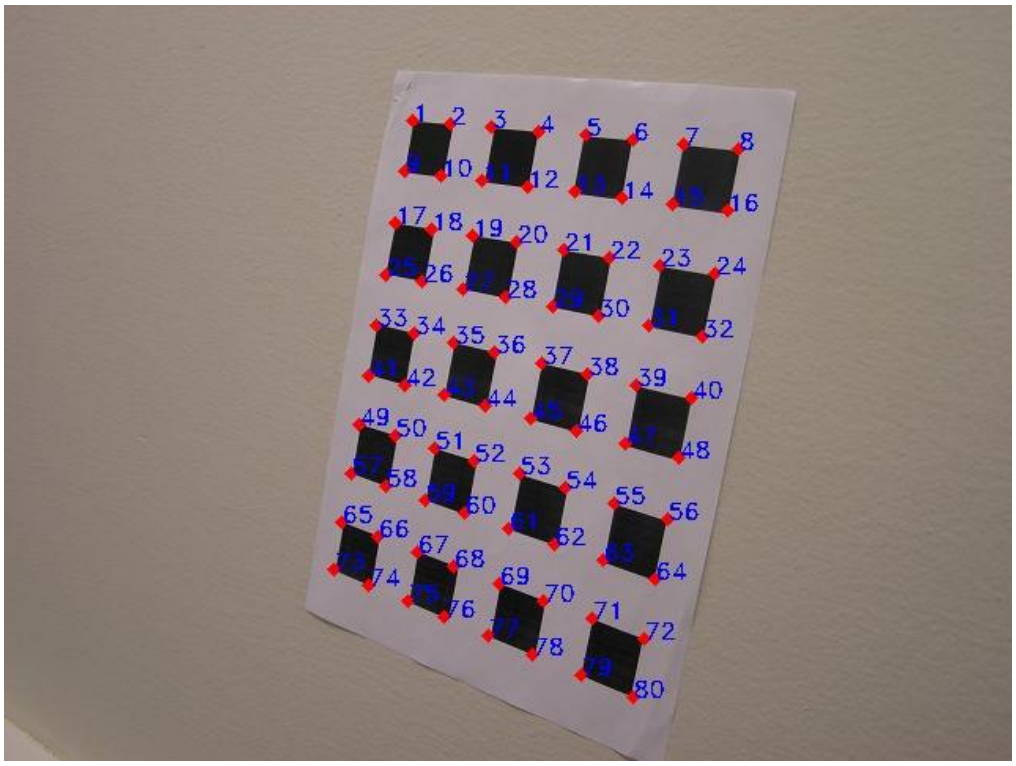

Figure 5: Corners detected for image 1

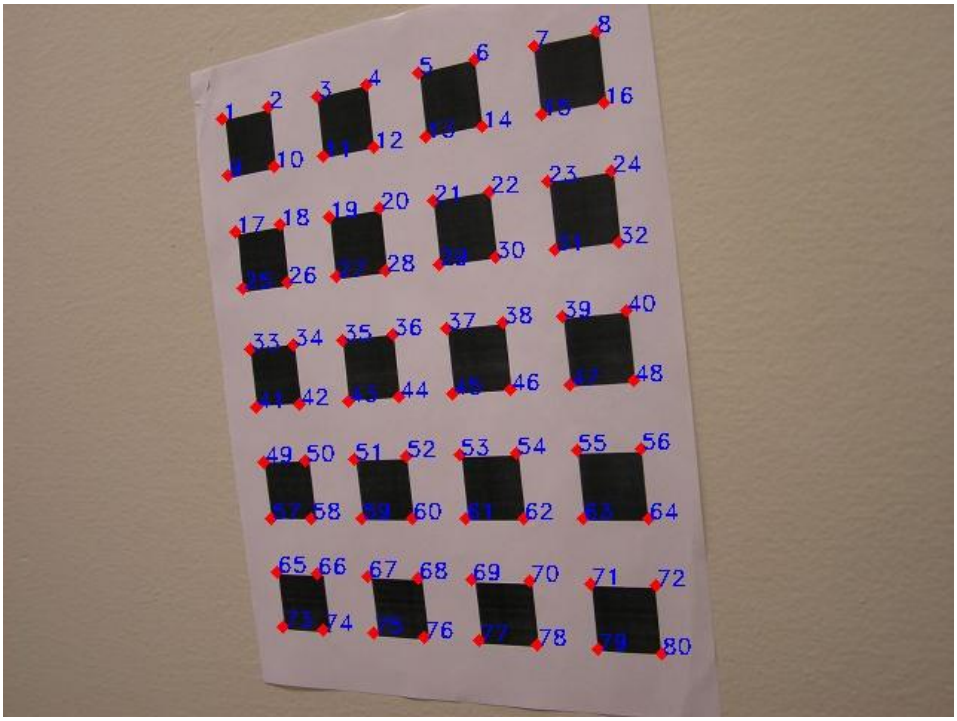

Figure 6: Corners detected for image 19

Output images for Reprojection:

Namrata Vivek Raghavan
nrgahav@purdue.edu

Red spots denoted the projected corner while the green denote the actual corners

Fixed Image: Image 11

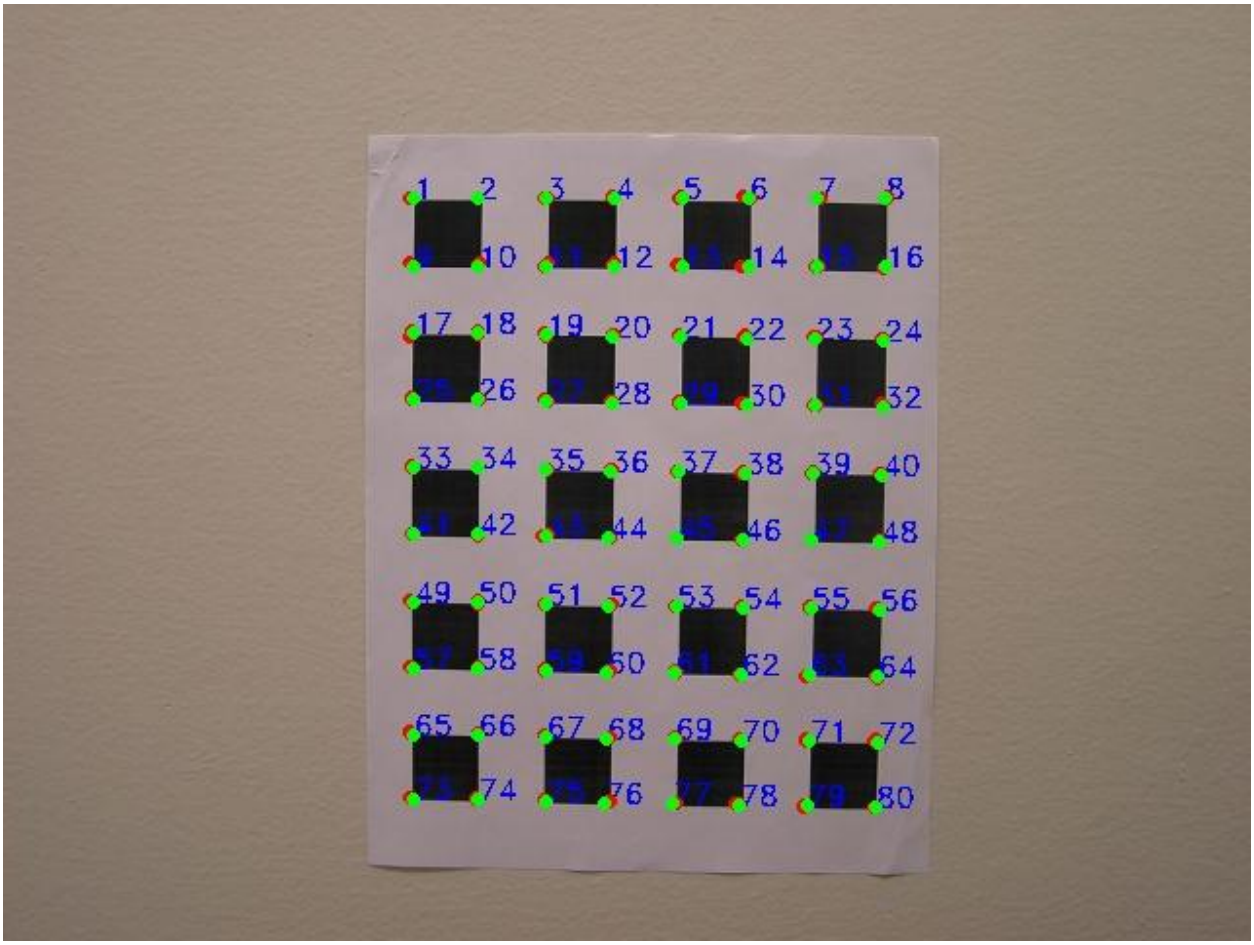

Figure 7: Reprojection of corners from image 2 to fixed image

Extrinsic parameters:

$$[R \mid t] = \begin{bmatrix} 0.74013 & 0.1002793 & -0.565255 & -151.29382 \\ 0.02611 & 0.875484 & -0.402312 & -332.5395 \\ 0.632699 & -0.437812 & 0.6749749 & 588.510591 \end{bmatrix}$$

Mean error = 1.189342

Variance of error = 0.495901

Namrata Vivek Raghavan
nrgahav@purdue.edu

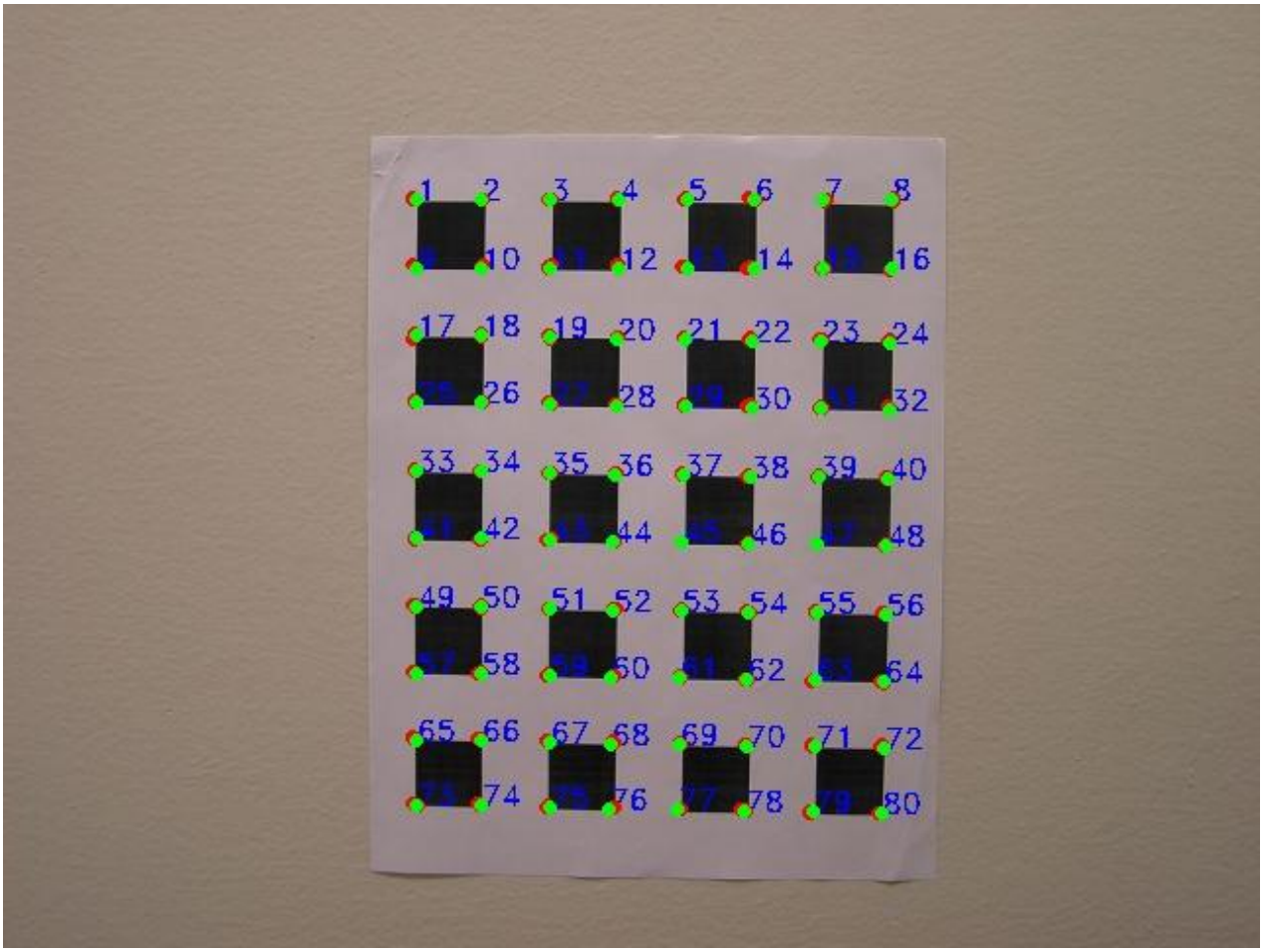

Figure 8: Reprojection of corners from image 5 to fixed image

Extrinsic parameters:

$$[R \mid t] = \begin{bmatrix} 0.83268 & -0.023956 & -0.491264 & -0.0118440 \\ -0.08696 & 0.89544737 & -0.028070 & -0.0278310 \\ 0.54678 & 0.01797688 & 0.7435400 & 503.45882 \end{bmatrix}$$

Mean error = 1.068924

Variance of error = 0.430459

Namrata Vivek Raghavan
nrgahav@purdue.edu

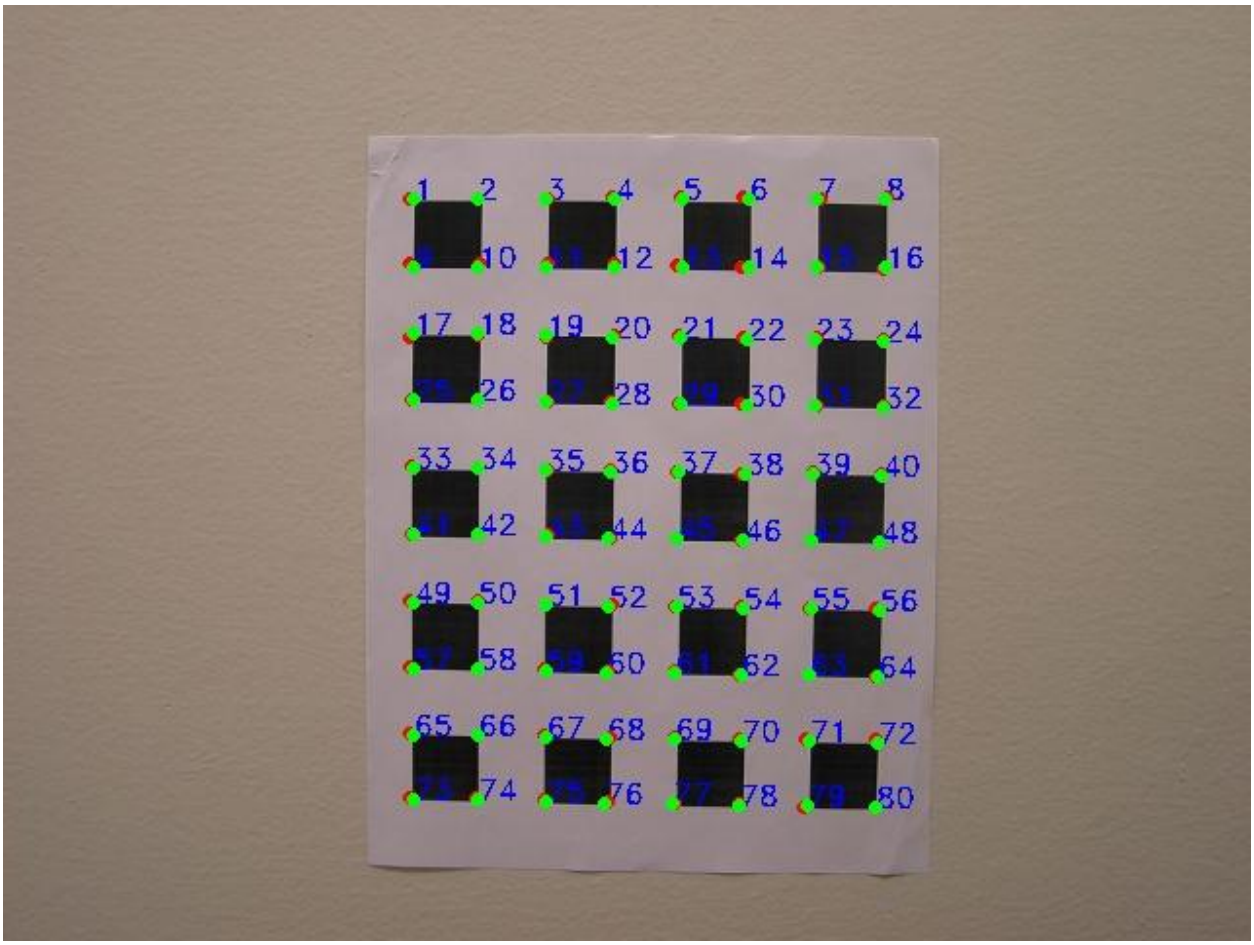

Figure 9: Reprojection of corners from image 10 to fixed image

Extrinsic parameters:

$$[R \mid t] = \begin{bmatrix} 0.989367 & -0.1421699 & -0.0387539 & -146.810341 \\ 0.13966 & 0.9877371 & -0.0150048 & -392.72669 \\ 0.04055 & 0.009338 & 0.97091598 & 720.387144 \end{bmatrix}$$

Mean error = 1.092491

Variance of error = 0.282684

Sample Intrinsic and extrinsic parameters

Namrata Vivek Raghavan
nrgahav@purdue.edu

Intrinsic parameters of camera:

$$K = \begin{bmatrix} 782.7596 & 0.7489488 & 229.0341 \\ 0 & 768.947 & 306.8278 \\ 0 & 0 & 1 \end{bmatrix}$$

Extrinsic parameters of image 1 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.719874 & -0.17500032 & 0.66800283 & 0.92186333 \\ 0.2593276 & 1.00008715 & 0.04575532 & -7.3179793 \\ -0.64384053 & 0.09295651 & 0.76531915 & 24.05689807 \end{bmatrix}$$

Extrinsic parameters of image 2 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.82977576 & -0.02560761 & -0.51817783 & 0.85593482 \\ -0.050784 & 0.92497077 & -0.0811599 & -5.46248655 \\ 0.55578158 & 0.08065753 & 0.76621787 & 19.66424239 \end{bmatrix}$$

Extrinsic parameters of image 3 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.862520084 & 0.166317 & -0.433346 & -317.04055 \\ -.225268423 & 0.90386779 & -0.01572990 & -5.007592 \\ 0.453115 & 0.105610 & 0.8170702 & 19.95867 \end{bmatrix}$$

Extrinsic parameters of image 4 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.81969866 & -0.1421699 & -0.3570473 & -0.52703885 \\ -0.42637383 & 0.85148739 & 0.09775844 & -4.04253813 \\ 0.38249113 & 0.07355273 & 0.09775844 & 22.2588574 \end{bmatrix}$$

Dataset 2:

Intrinsic camera matrix:

$$K = \begin{bmatrix} 838.4596 & -3.17548092 & 278.85921937 \\ 0 & 838.13855446 & 245.55873727 \\ 0 & 0 & 1 \end{bmatrix}$$

Output images for edge detection

Figure 1: Canny Edge Detection Output for image 1

Namrata Vivek Raghavan
nrgahav@purdue.edu

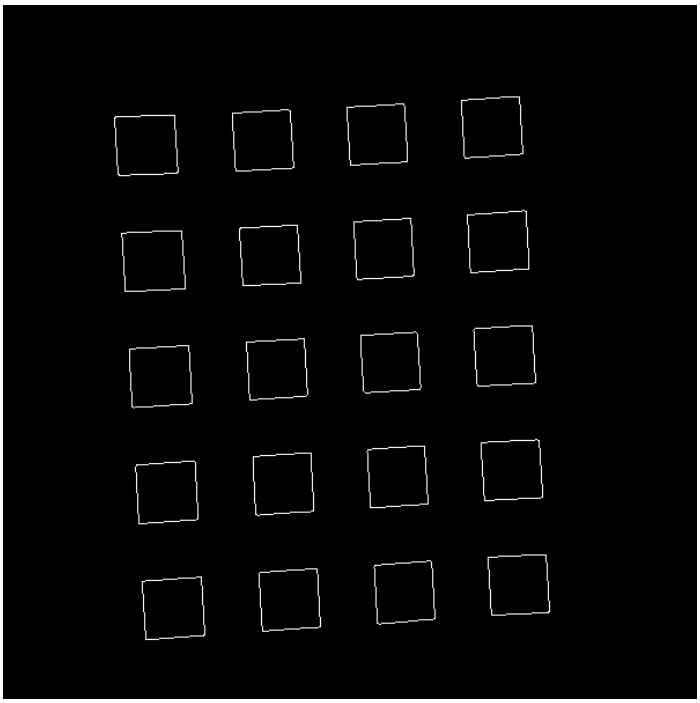

Figure 10: Canny Edge Detection Output for image 2

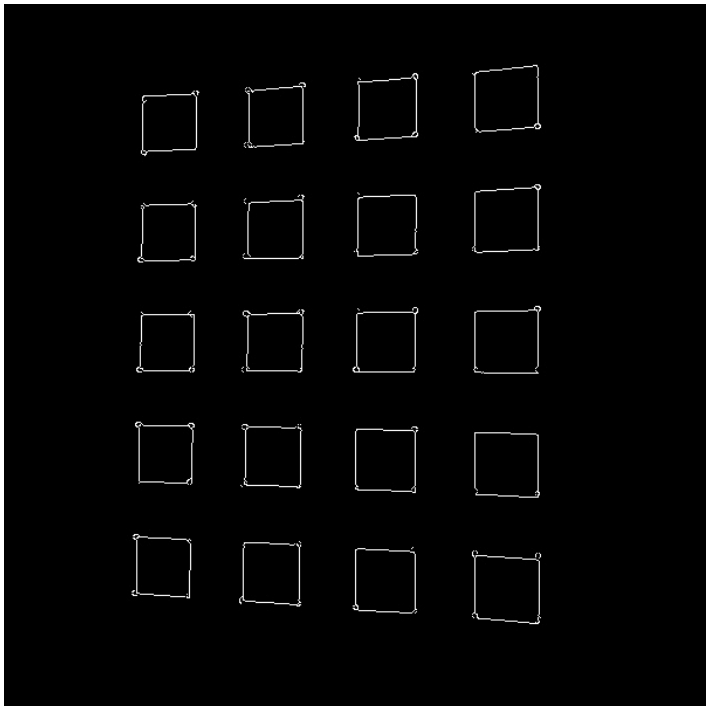

Figure 11: Canny Edge Detection Output for image 8

Namrata Vivek Raghavan
nrgahav@purdue.edu

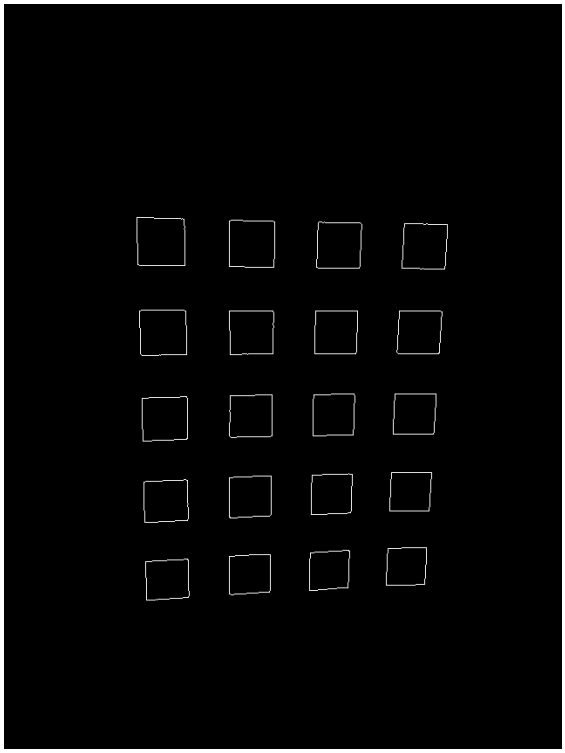

Figure 12: Canny Edge Detection Output for image 22

Output images for hough-lines detection

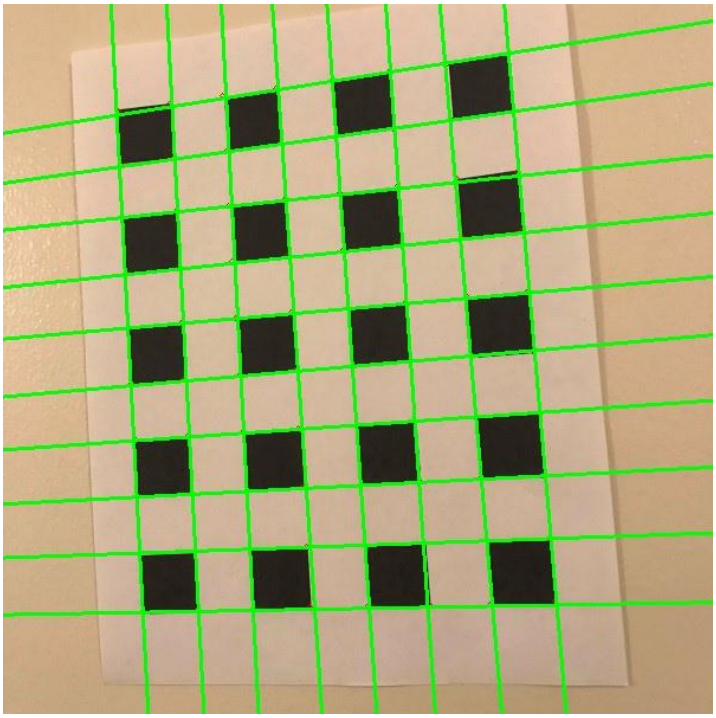

Figure 13: Hough-line detected for image 4

Namrata Vivek Raghavan
nrgahav@purdue.edu

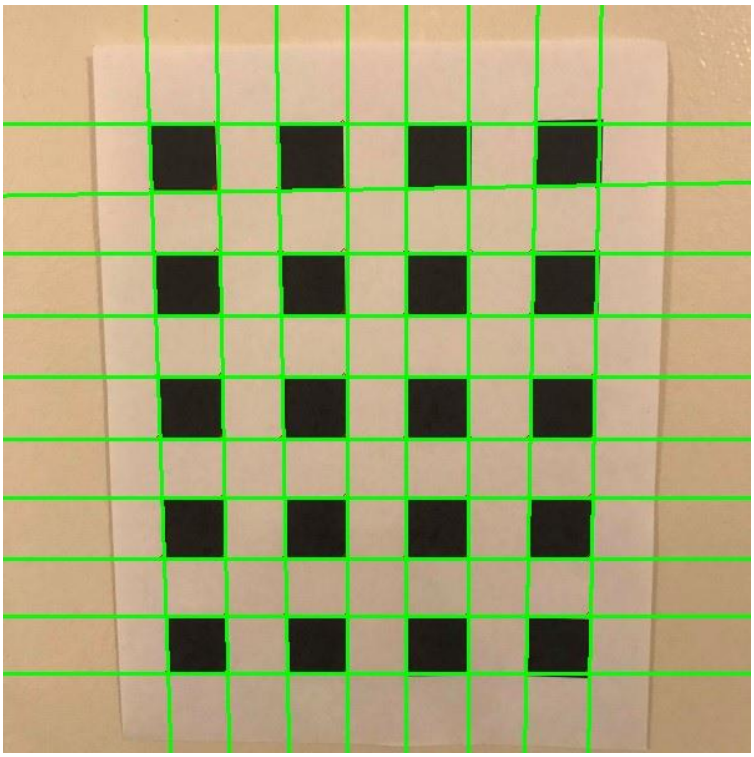

Figure 14: Hough-line detected for image 9

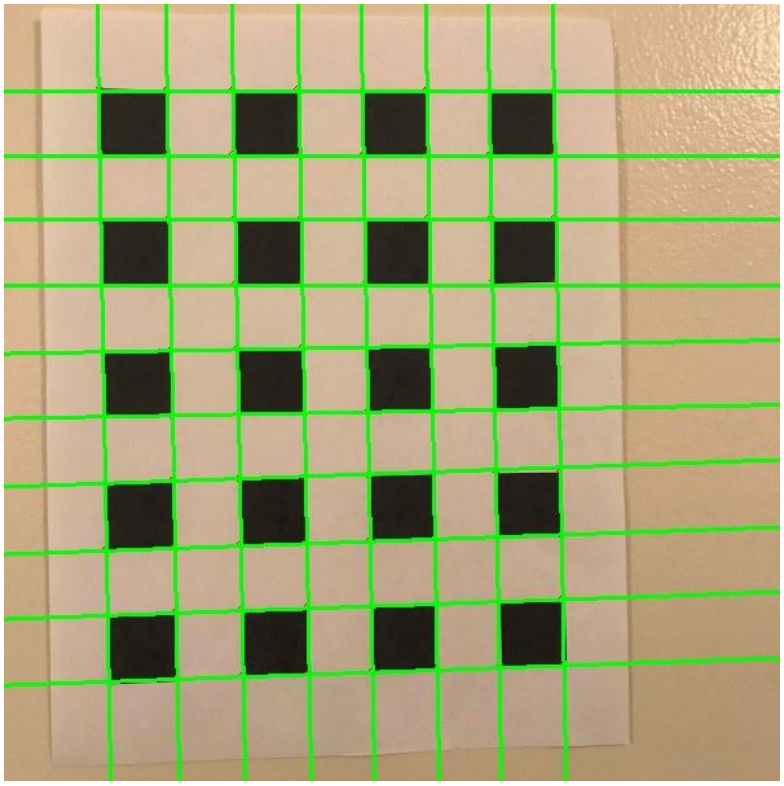

Figure 15: Hough-line detected for image 20

Output images for corner detection

Namrata Vivek Raghavan
nrgahav@purdue.edu

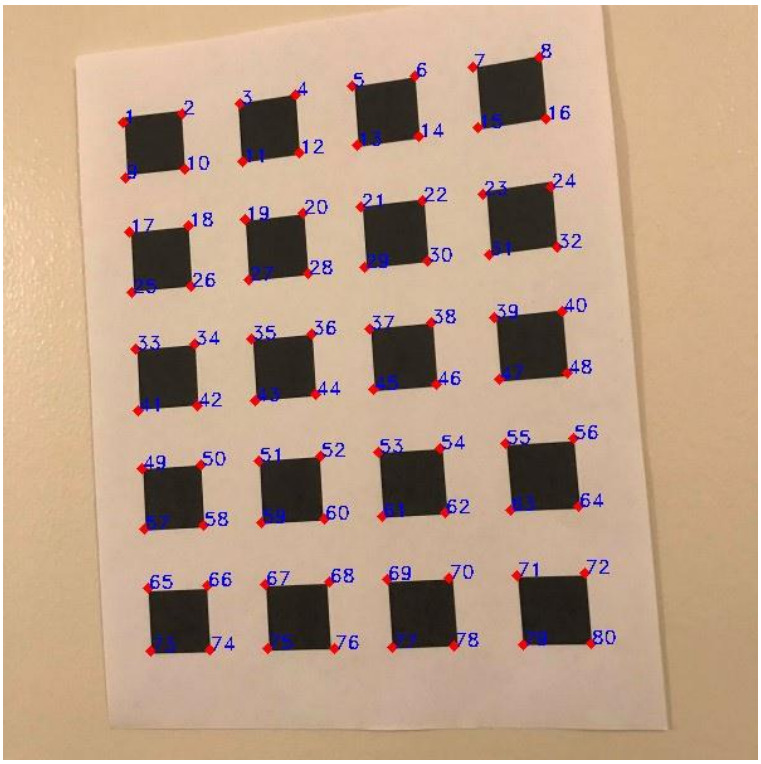

Figure 16: Corners detected for image 4

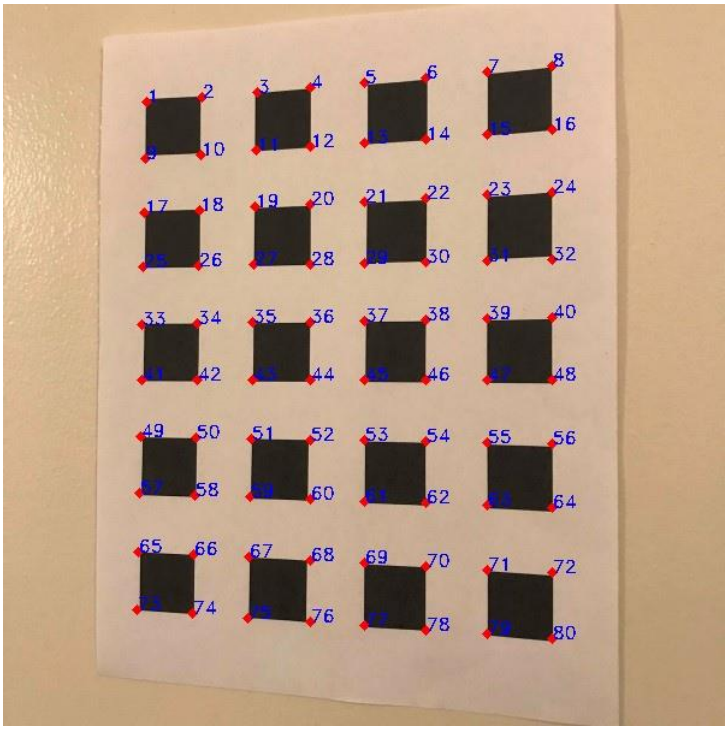

Figure 16: Corners detected for image 8

Namrata Vivek Raghavan
nrgahav@purdue.edu

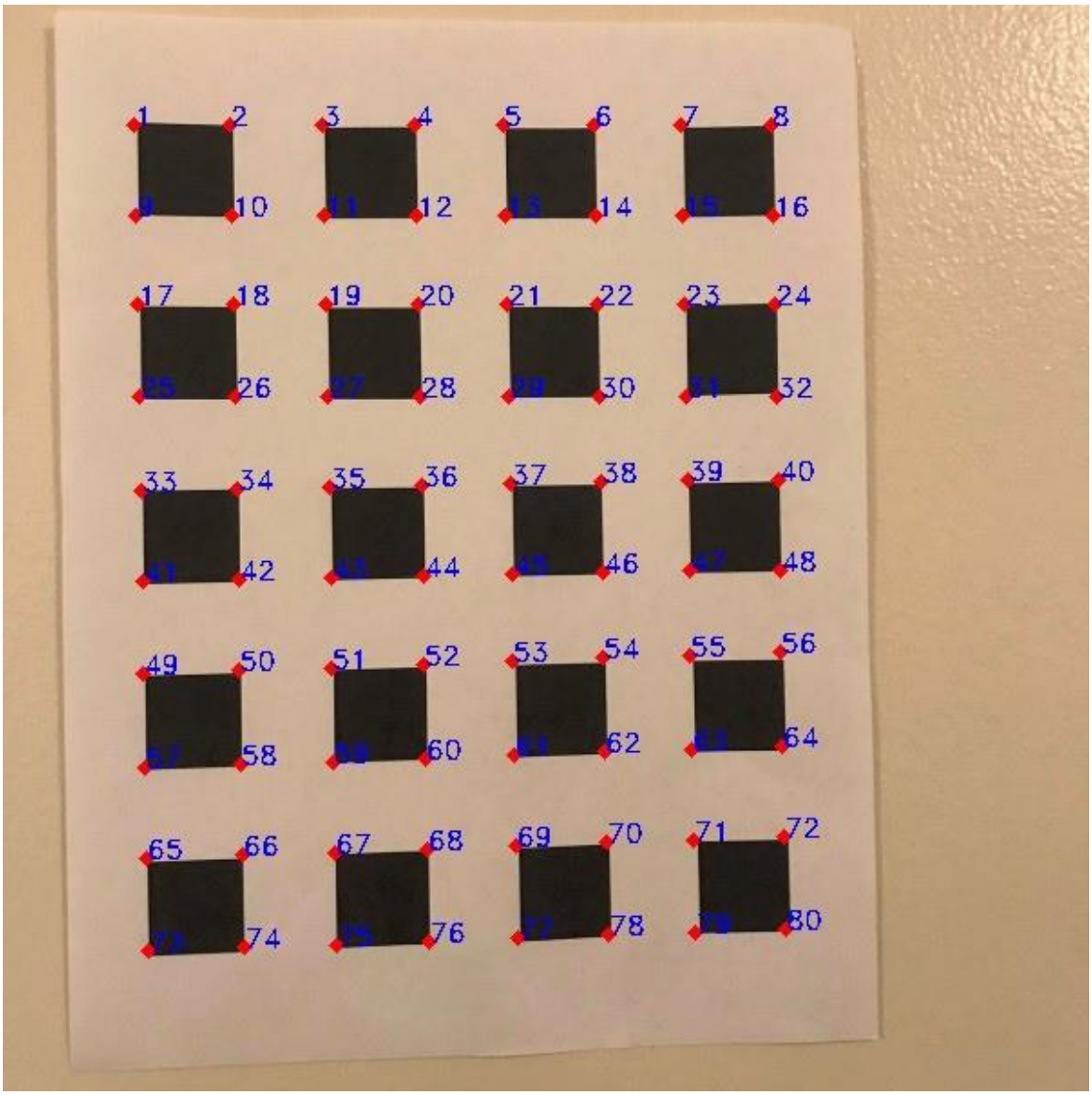

Figure 16: Corners detected for image 20

Output images for Reprojection:

Red spots denoted the projected corner while the green denote the actual corners

Namrata Vivek Raghavan
nrgahav@purdue.edu

Fixed Image: Image 6

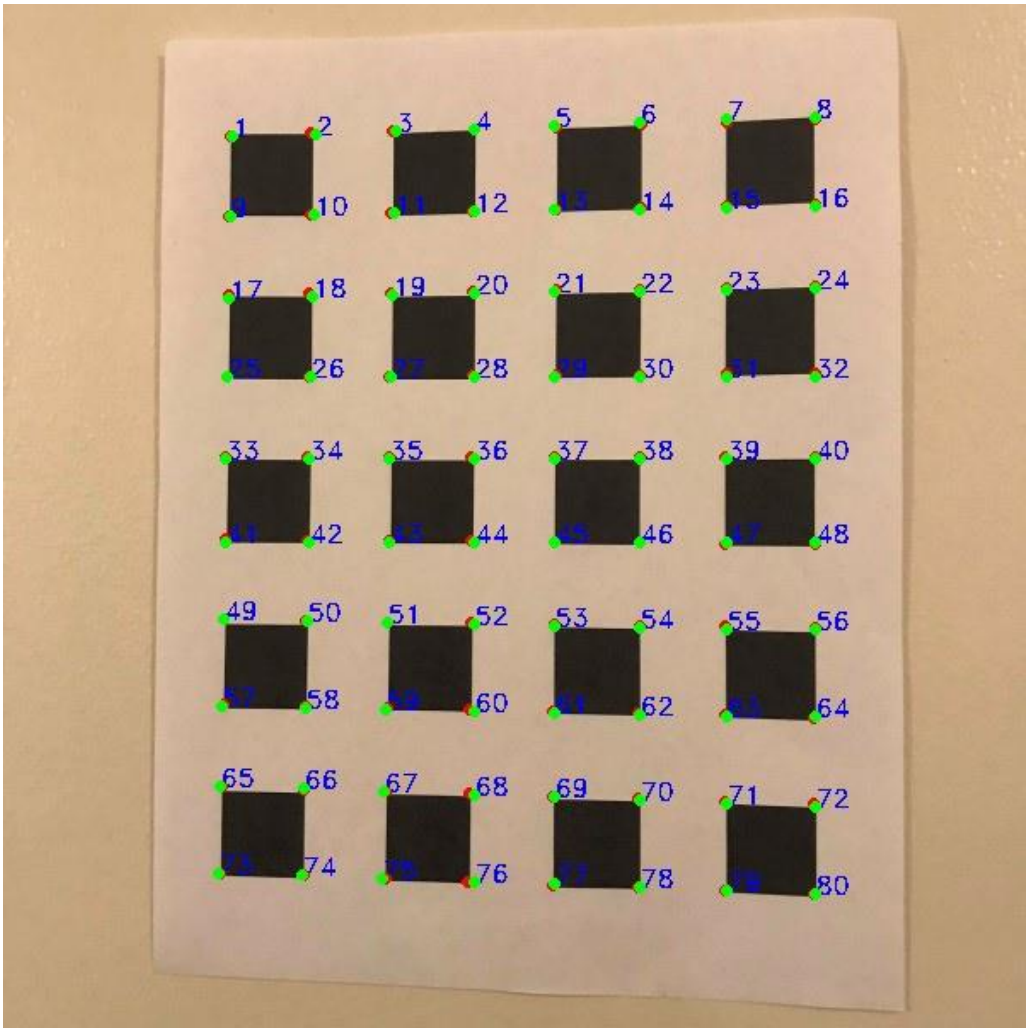

Figure 17: Reprojection of corners from image 1 to fixed image

Extrinsic parameters:

$$[R \mid t] = \begin{bmatrix} 0.994457907 & 0.00642376 & -.103925305 & -324.51884 \\ -0.0147003229 & 0.994585167 & -.0254570224 & -362.45647 \\ 0.104102701 & 0.0262785292 & .989168523 & 698.5495 \end{bmatrix}$$

Mean error = 1.116045

Variance of error = 0.387945

Namrata Vivek Raghavan
nrgahav@purdue.edu

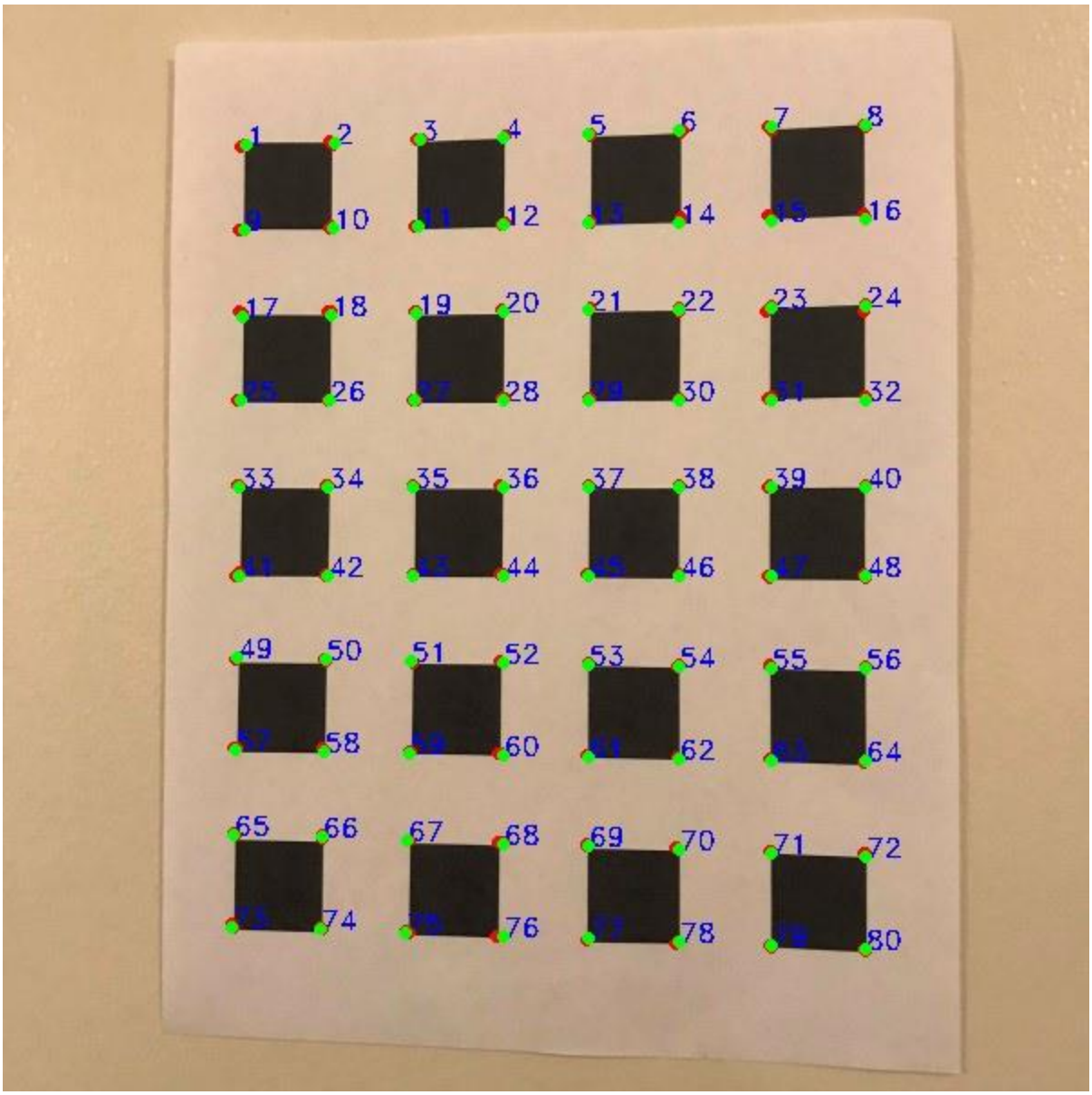

Figure 18: Reprojection of corners from image 4 to fixed image

Extrinsic parameters:

$$[R \mid t] = \begin{bmatrix} .992433 & 0.07823474 & 0.0990948022 & -372.898692 \\ -0.077403960 & 1.00514788 & 0.034756157 & -356.608 \\ -0.09531171 & -0.042534689 & 1.0035981 & 784.1117 \end{bmatrix}$$

Mean error = 1.229830

Variance of error = 0.541647

Namrata Vivek Raghavan
nrgahav@purdue.edu

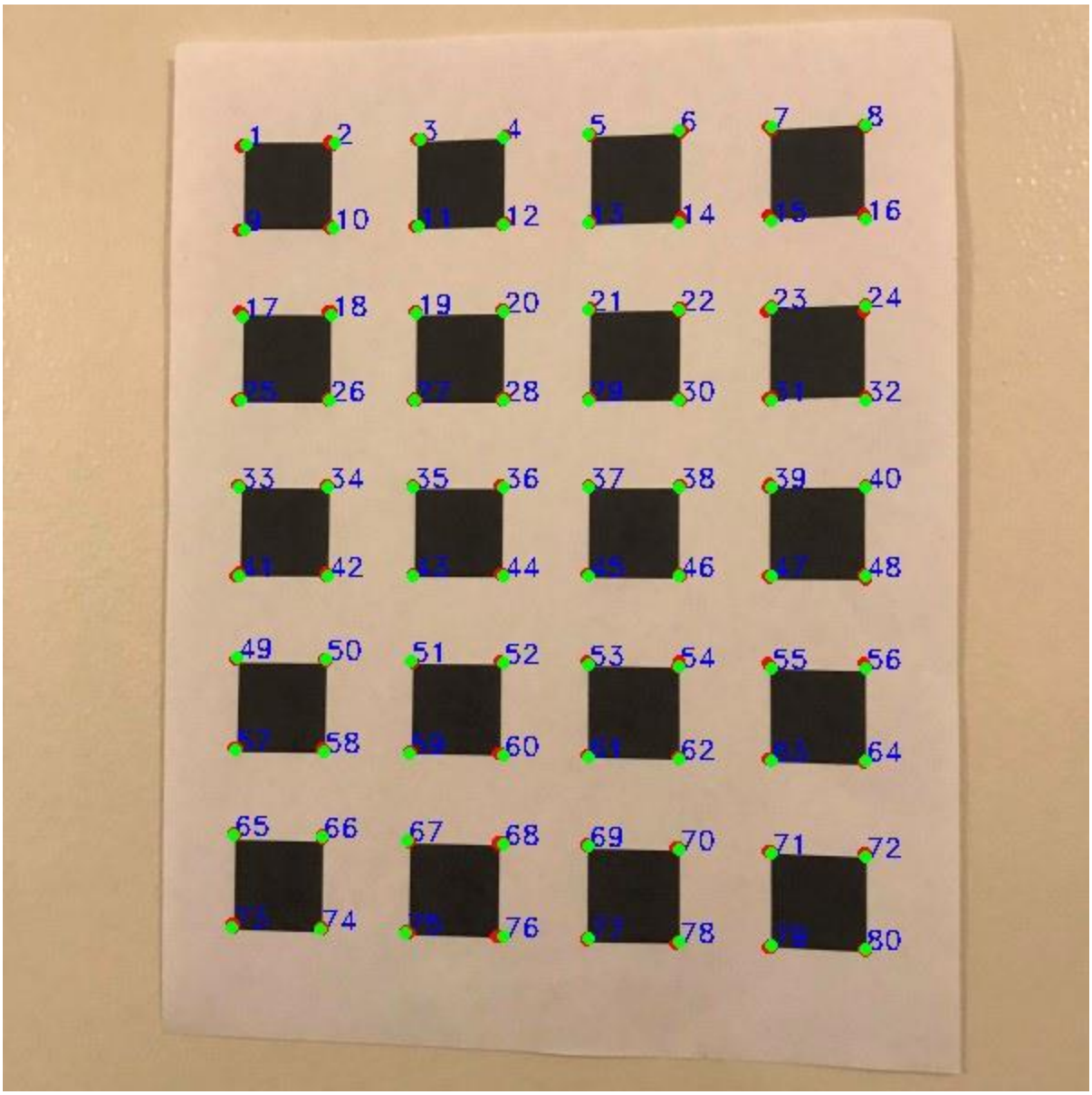

Figure 19: Reprojection of corners from image 20 to fixed image

Extrinsic parameters:

$$[R \mid t] = \begin{bmatrix} 0.9828527 & 0.021807304 & -0.17925819 & -365.3333 \\ -0.0434821 & .989556 & -0.0260455 & -354.3885 \\ .1799041 & 0.0304916311 & .973470005 & 661.9648 \end{bmatrix}$$

Mean error = 1.160506

Variance of error = 0.378103

Sample Intrinsic and extrinsic parameters

Namrata Vivek Raghavan
nrgahav@purdue.edu

Intrinsic parameters of camera:

$$K = \begin{bmatrix} 838.4596 & -3.17548092 & 278.85921937 \\ 0 & 838.13855446 & 245.55873727 \\ 0 & 0 & 1 \end{bmatrix}$$

Extrinsic parameters of image 1 of dataset:

$$[R \mid t] = \begin{bmatrix} .99999972 & 0.00337916 & 0.0023793084 & -3.15287241 \\ -1.33536118 & 1.00105 & 0.0197236133 & -3.555069 \\ -2.37417842 & -0.019724 & 1.00105 & 17.14309 \end{bmatrix}$$

Extrinsic parameters of image 2 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.96991877 & 0.07121145 & 0.22911788 & -3.91554047 \\ -0.09392145 & 0.98057117 & 0.07592585 & -3.33097982 \\ -0.22458035 & -0.09476932 & 0.95776267 & 18.05040176 \end{bmatrix}$$

Extrinsic parameters of image 3 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.991245233 & 0.014300480 & .130751462 & -3.447835 \\ -0.02295286 & .996737571 & 0.0479069806 & -3.44783832 \\ -.130023284 & -0.0502059172 & .988339602 & 17.8120159 \end{bmatrix}$$

Extrinsic parameters of image 4 of dataset:

$$[R \mid t] = \begin{bmatrix} 0.99997 & 0.0108988 & .2606011 & -3.398625 \\ -0.002189 & .995935640 & 0.0518735 & -3.52288239 \\ -.0065747 & .12172144 & .9503788 & 17.879602 \end{bmatrix}$$

Optimization of reprojection in data set 1 using LM algorithm

Namrata Vivek Raghavan
nrgahav@purdue.edu

Red spots denoted the projected corner while the green denote the actual corners

Projection of corners in image 11 to image 1

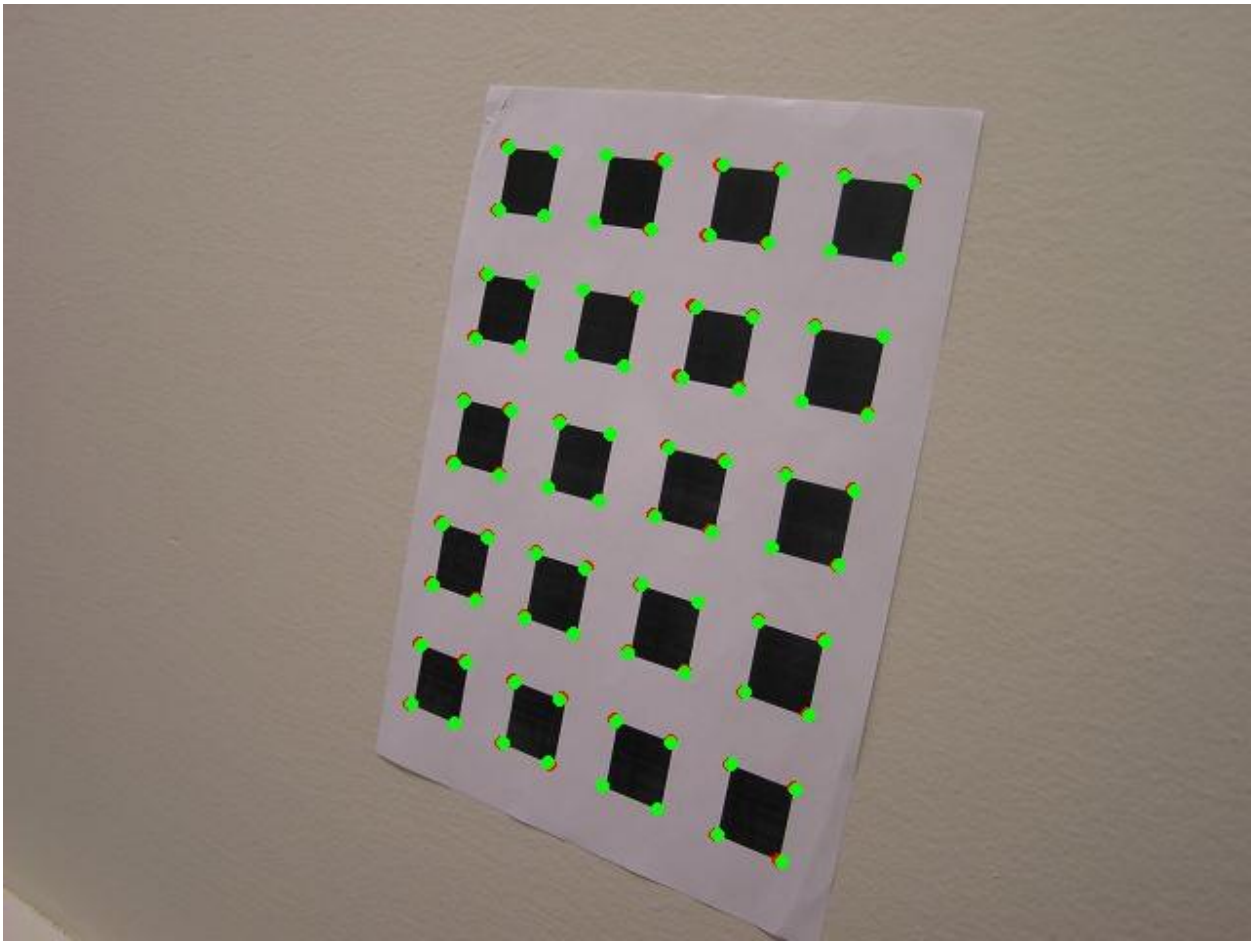

Figure 20: Reprojection of corners from image 11 to image 1 without using LM

Projection Matrix

$$K[r_1\ r_2\ t] = \begin{bmatrix} 0.041622109 & -0.0114943962 & 6.22596667e+03 \\ 1.8609847 & 0.0797535675 & 1.75418779e+03 \\ -.643840526 & .0929565118 & 2.40568981e+01 \end{bmatrix}$$

Mean of error: 0.694886

Variance of error: 0.175593

Namrata Vivek Raghavan
nrgahav@purdue.edu

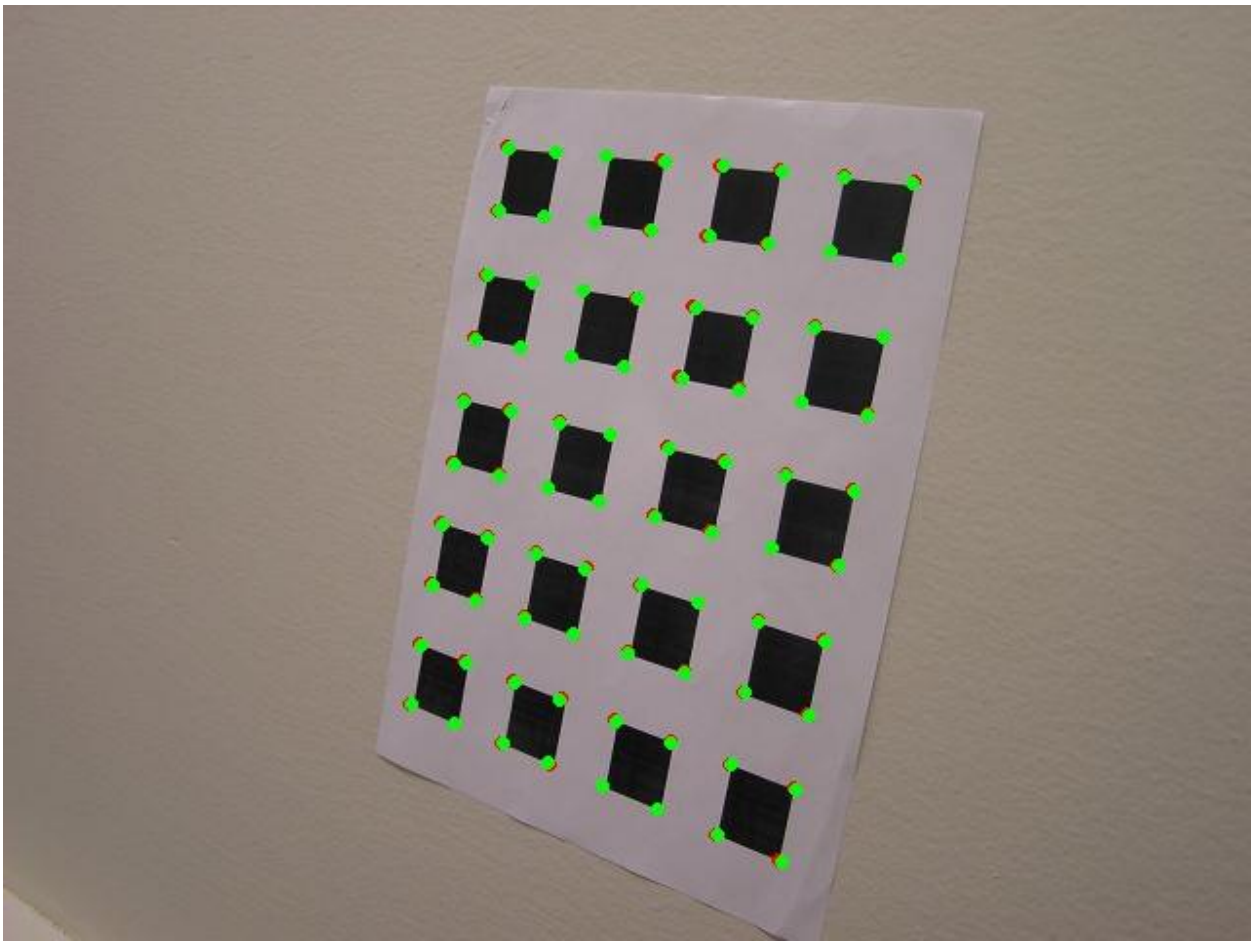

Figure 21: Reprojection of corners from image 11 to image 1 using LM

Projection Matrix

$$K[r_1\ r_2\ t] = \begin{bmatrix} 4.16220150e+02 & -1.14943957e+02 & 6.22596590e+03 \\ 1.86355070 & 7.97537671e+02 & 1.75420575e+03 \\ -6.43838742e-01 & 9.29569273e-02 & 2.40569028e+01 \end{bmatrix}$$

Mean of error : 0.694816

Variance of error: 0.175688

Projection of corners in image 11 to image 2

Namrata Vivek Raghavan
nrgahav@purdue.edu

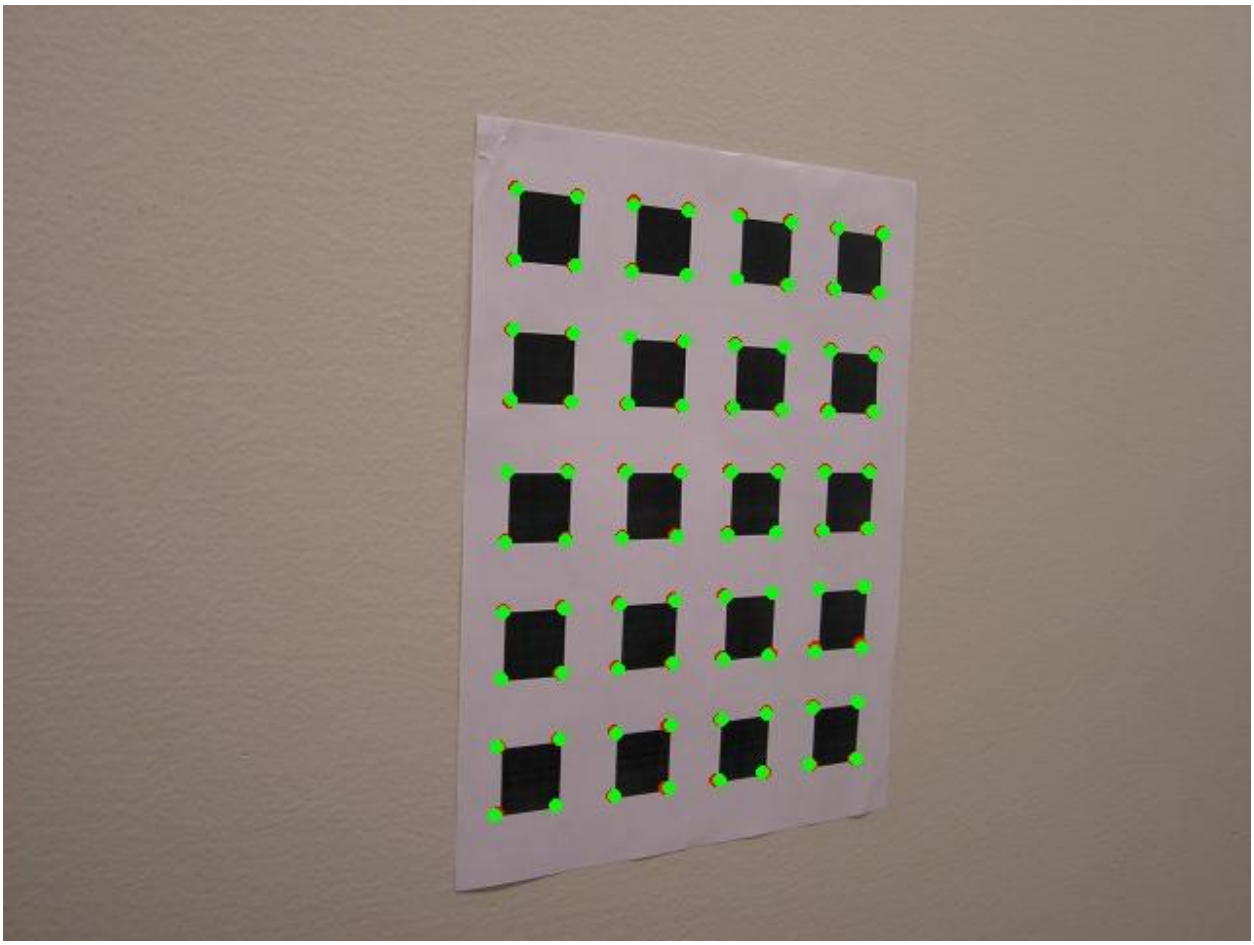

Figure 21:Reprojection of corners from image 11 to image 2 without using LM

Projection Matrix

$$K[r_1\ r_2\ t] = \begin{bmatrix} 7.76769871e+02 & -8.78525884e-01 & 5.16968221e+03 \\ 1.31479057e+02 & 7.36001484e+02 & 1.83317433e+03 \\ 5.55781583e-01 & 8.06575262e-02 & 1.96642424e+01 \end{bmatrix}$$

Mean of error: 0.665445

Variance of error: 0.127275

Namrata Vivek Raghavan
nrgahav@purdue.edu

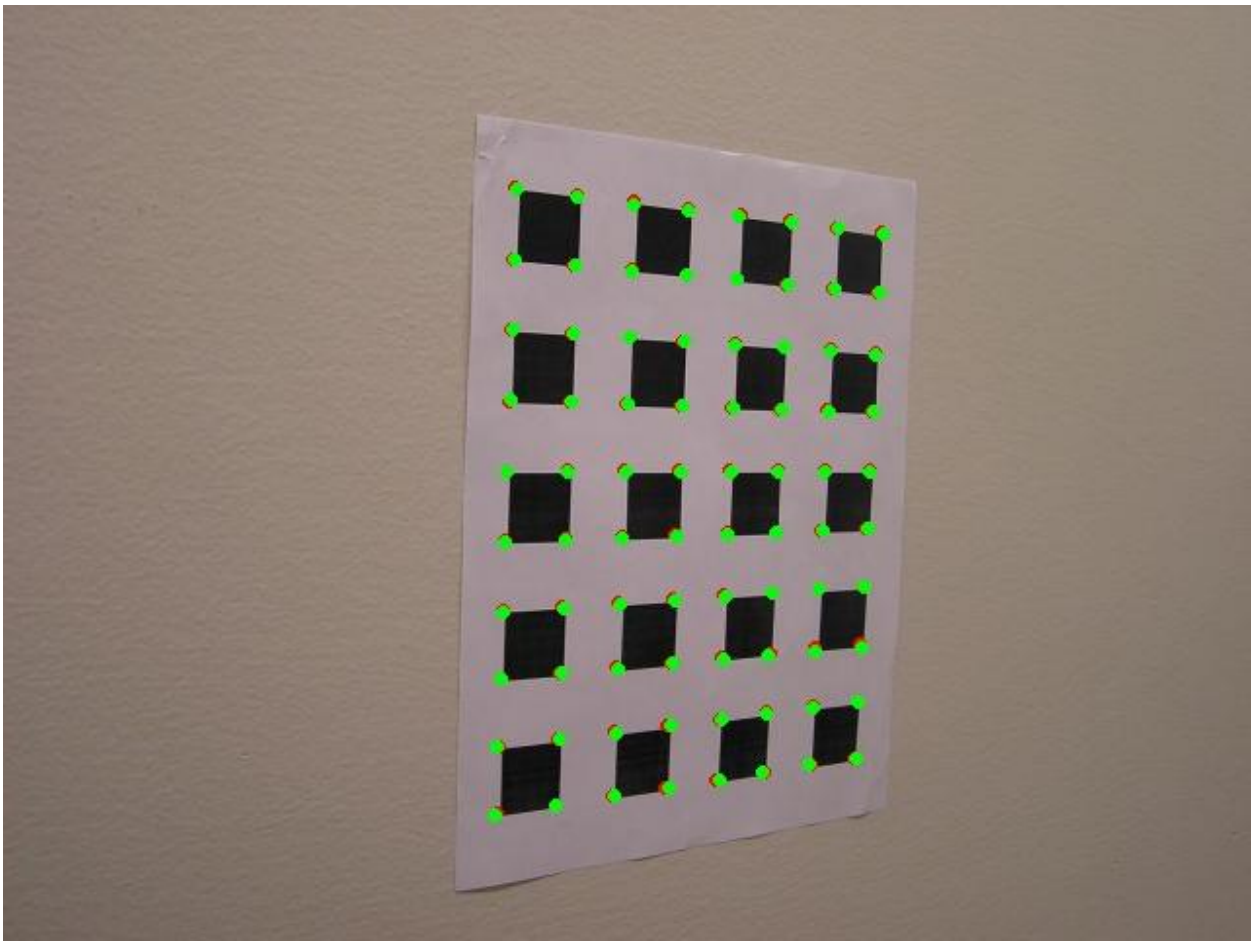

Figure 23: Reprojection of corners from image 11 to image 2 using LM

Projection Matrix

$$K[r_1\ r_2\ t] = \begin{bmatrix} 7.76769932e+02 & -8.78607744e-01 & 5.16968187e+03 \\ 1.31478164e+02 & 7.36000790e+02 & 1.83316808e+03 \\ 5.55781475e-01 & 8.06577449e-02 & 1.96642431e+01 \end{bmatrix}$$

Mean of error: 0.665420

Variance of error: 0.127309