ECE 661 Fall 2018
Homework #8

Ye Shi
shi349@purdue.edu

November 8, 2018

Contents

1 Methodology

1.1 Corner Detection

1. Apply Canny edge detector;
2. Use Hough Transform to find candidate lines;
3. Transform the candidate lines to Homogeneous Coordinate(HC) representation as $l = [a\ b\ 1]$, Pick the 10 lines with top a values as the vertical lines,
   the 8 lines with top b values as the horizon lines;
   (a is the inverse of the y-intercept and b is x-intercept. In the image plane, the vertical lines are listed form left to right according to a value; the horizon are listed from top to bottom.)
4. Find all corners from the vertical and horizon lines and label them by the order we mention above.

1.2 Camera Calibration

This is based on Zhang's Alogorithm. It assumes the calibration pattern is on the plane $z = 0$. We can write

$$x' = K[R|\vec{t}]x,$$

where $\vec{x'} = [x'\ y'\ w']^T$ of the point is the pixel location in image coordinates;
$\vec{x} = [x\ y\ 0\ w]^T$ is the point in world coordinates;
$K = \begin{bmatrix} a_x & s & x_0 \\ 0 & a_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$ is the camera intrinsic matrix;
R is the 3×3 rotation matrix; $\vec{t}$ is the translation vector from world coordinates to camera coordinates.

Before we apply the algorithm, we need to find all the homographies $\{H = K[R|\vec{t}]\}$ of the image set by least squre estimation (LSE).

1.2.1 Camera Intrinsic Matrix

We introduce a varible

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

where hab is the element at a row and b column of H.
We have a vector b, where

$$\begin{bmatrix} v_{12} & (v_{11} - v_{12}) \end{bmatrix}^T b = 0.$$

We then apply Least Squre Estimation (LSE) to all images in the dataset, and we apply SVD to solve b.

Then we get K by

$$
\begin{aligned}
y_0 &= \frac{b_2 b_4 - b_1 b_5}{b_1 b_3 b_2^2} \\
\lambda &= b_6 - \frac{b_4^2 + y_0 * (b_2 b_4 - b_1 b_5)}{b_1} \\
a_x &= \sqrt{\frac{\lambda}{b_1}} \\
a_y &= \sqrt{\frac{\lambda b_1}{b_1 b_3 - b_2^2}} \\
s &= -\frac{b_2 a_x^2 a_y}{\lambda} \\
x_0 &= \frac{s y_0}{a_y} - \frac{b_4 a_x^2}{\lambda}.
\end{aligned}
$$

1.2.2 Camera Extrinsic Parameters

Since we have K in former section and $H = K[R|\vec{t}]$ for each image, we have

$$[r_1 \ r_2 \ \vec{t}] = K^{-1} H$$

and

$$r_3 = r_1 \times r_2,$$

where $R' = [r_1 \ r_2 \ r_3]$ Because the columns of rotation matrix should be orthogonal, we can use SVD to fix it as

$$
\begin{aligned}
U D V^H &= R' \\
R &= U V^H
\end{aligned}
$$

1.2.3 Radial Distortion

In this homework, I assume the principal point is the central of the image (x'_c, y'_c) since I do not have the physical parameters of the camera. And k_1 and k_2 are introduced as the radial distortion parameters of the camera.

We then have

$$
\begin{aligned}
r^2 &= (x' - x'_c)^2 + (y' - y'_c)^2 \\
\begin{cases} \tilde{x'} = & x' + (x' - x'_c)(k_1 r^2 + k_2 r^4) \\ \tilde{y'} = & y' + (y' - y'_c)(k_1 r^2 + k_2 r^4) \end{cases} &.
\end{aligned}
$$

$(\tilde{x'}, \tilde{x'})$ is the coordinates with the radial distortion.

In the homework, I naively use Levenberg–Marquardt(LM) to find the k_1 and k_2 from zeros.

1.3 Levenberg–Marquardt(LM) Refinement

Since R is a 3×3 matrix and has 3 DoFs, we need to transfer it to a rotation vector form by Rodrigues rotation formula as follows.

$$\Phi = \arccos(\frac{trace(R) - 1}{2})$$

$$\vec{\omega} = \frac{\Phi}{2\sin(\Phi)} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

To reconstruct R,

$$W = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}$$

$$R = I + \frac{2\sin(\Phi)}{\Phi} W + \frac{1-\cos(\Phi)}{\Phi} W^2.$$

K has 5 DoFs; $\vec{\omega}$ has 3 DoFs and $\vec{t}$ has 3 DoFs. Then we construct a vector

$$\vec{p} = [k_{11}\ k_{12}\ k_{13}\ k_{22}\ k_{23}\ \vec{\omega_1}\ \vec{t_1}\ \vec{\omega_2}\ \vec{t_2}\ \ldots\ \vec{\omega_n}\ \vec{t_n}]^T$$

where n is nth image in dataset.

The cost function is

$$d^2_{geom}(\vec{p}) = \sum_n \sum_i \sum_j \|x_{ij\in n} - x'_{ij\in n}\|^2.$$

For radial distortion, we only need to add k_1 and k_2 into $\vec{p}$.

2 Results

This section presents the results with 4 images randomly picked with different views in each dataset. All the calculation results are produced with whole usage of the dataset, whereas I used all images to calibrated the camera matrices.

The LM method always gives the best result among all images and datasets.

It is also noticeable that most quantitative estimation from LM w/ radial distortion are the same to LSE. However, they are not the same but similar. There are several reasons caused this and I listed them as follows.

1. The cameras have very little radial distortion and/or have inner calibration to correct radial distortion. The extra parameters make the calibration matrix overfit.
2. We use the Hough algorithm to detect line and corners, which will not generate the ground truth of the corner location.
3. I do not initialize a k_1 and k_2, but use zeros as the seed for LM optimization.

Here are the instructions and notations for Fig.1, 2, 3, 4, 6,7, 8, and 9.

- all corner labels are in yellow;
- all sub-figures (a) are the Canny edge detection;
- all sub-figures (b) are the Hough line detection, where horizon lines are blue, vertical lines are green and conners are red '×';
- all sub-figures (c) are the LSE re-projection vs LM re-projection from the ground truth, where LSE points are marked by blue '+' and LM points are marked by red '×';
  (Sometimes you may regard red '×' as purple '×' due to the combination with blue '+'. You may zoom out the figure to see it better.)
- all sub-figures (d) are LM re-projection vs LM re-projection w/ radial distortion from the ground truth, where LM points are marked by red '×' and LM w/ radial distortion are marked by green '+';

2.1 Given Dataset

The dataset is provided by the homework instruction website.

- LSE

$$K = \begin{bmatrix} 723.4804 & 1.4809 & 321.8777 \\ 0 & 721.5681 & 328.2272 \\ 0 & 0 & 1 \end{bmatrix}$$

- LM

$$K = \begin{bmatrix} 722.3751 & 1.8197 & 325.8470 \\ 0 & 720.9432 & 239.8996 \\ 0 & 0 & 1 \end{bmatrix}$$

- LM w/ radial Distortion

$$K = \begin{bmatrix} 723.4804 & 1.4809 & 321.8777 \\ 0 & 721.5681 & 238.2272 \\ 0 & 0 & 1 \end{bmatrix}$$

$$k_1 = -6.8609e - 9$$

$$k_2 = 4.2824e - 14$$

2.1.1 Pic_1.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.7898 & -0.1804 & 0.5862 \\ 0.1977 & 0.9796 & 0.0350 \\ -0.5806 & 0.0883 & 0.8094 \end{bmatrix}$$

$$t = \begin{bmatrix} -46.8456 \\ -125.2711 \\ 551.7947 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.7873 & -0.1824 & 0.5890 \\ 0.1965 & 0.9797 & 0.0407 \\ -0.5844 & 0.0836 & 0.8071 \end{bmatrix}$$

$$t = \begin{bmatrix} -49.5042 \\ -125.5576 \\ 548.2087 \end{bmatrix}$$

- LM w/ radial Distortion

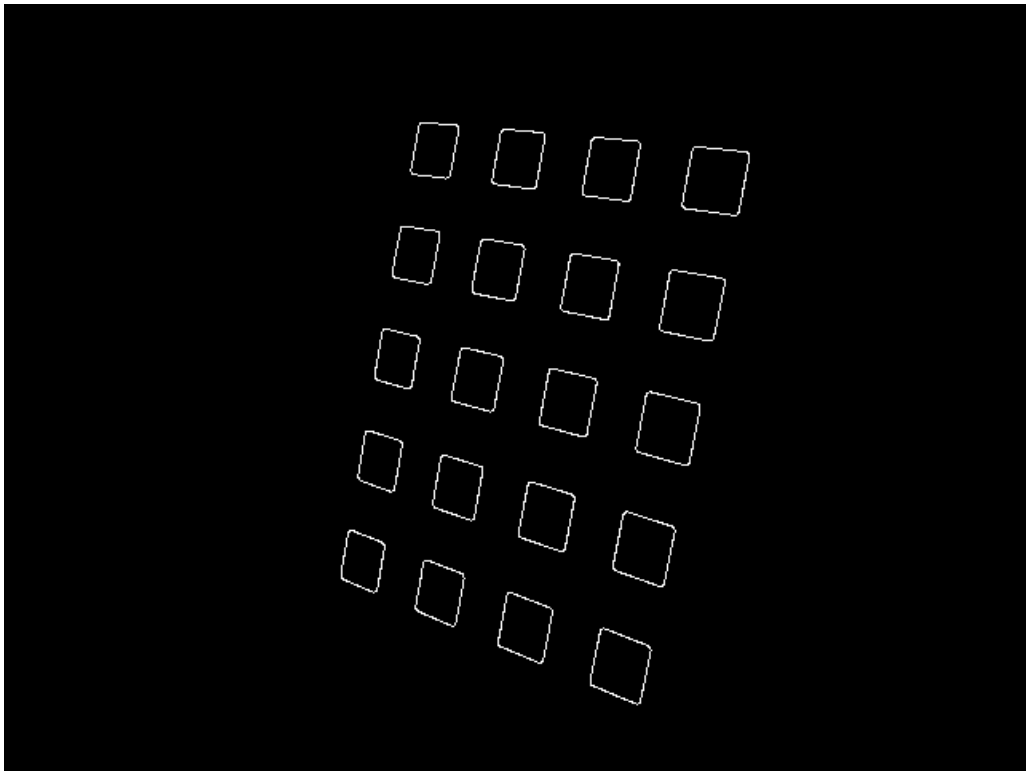
$$R = \begin{bmatrix} 0.7898 & -0.1804 & 0.5862 \\ 0.1977 & 0.9796 & 0.0350 \\ -0.5806 & 0.0883 & 0.8094 \end{bmatrix}$$

$$t = \begin{bmatrix} -46.8456 \\ -125.2711 \\ 551.7947 \end{bmatrix}$$

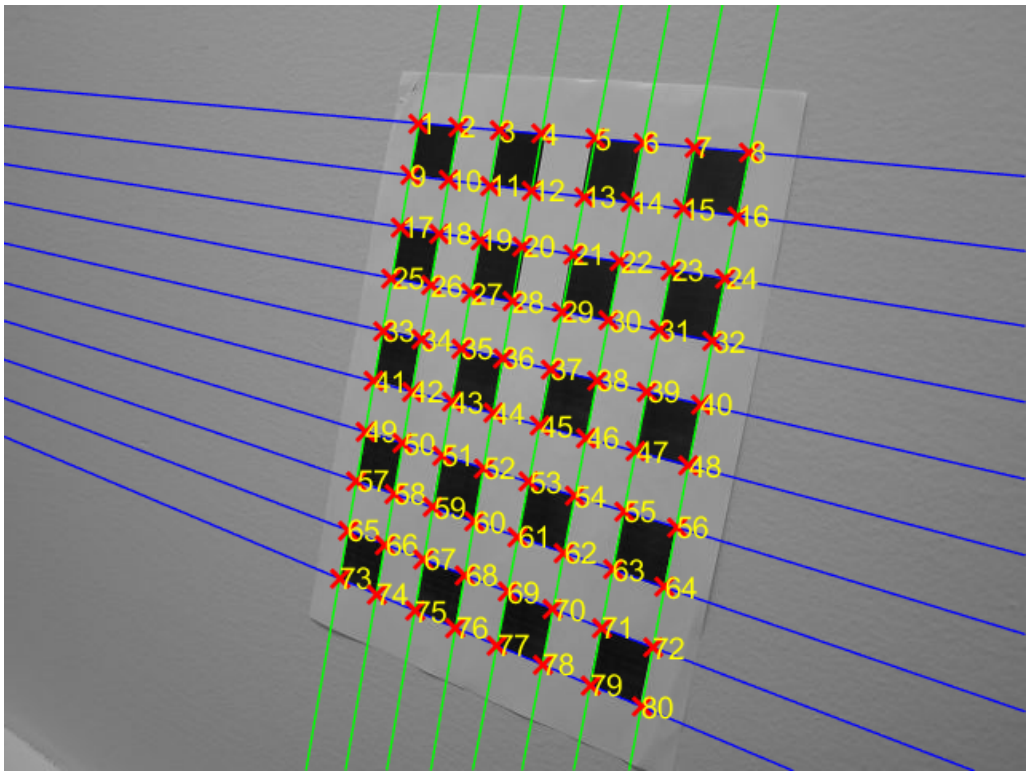
Table.1 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	1.5123	0.5988
LM	0.8464	0.1699
LM w/ radial Distortion	1.5123	0.5987

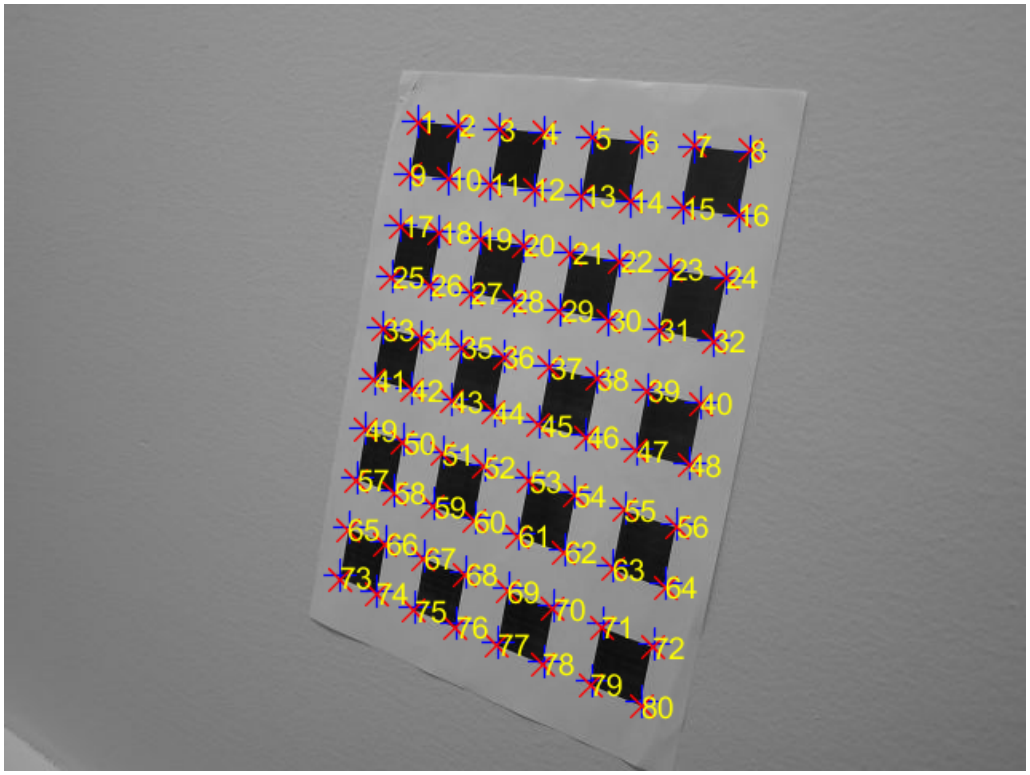
Table 1: Quantitative Comparison Between Different Method



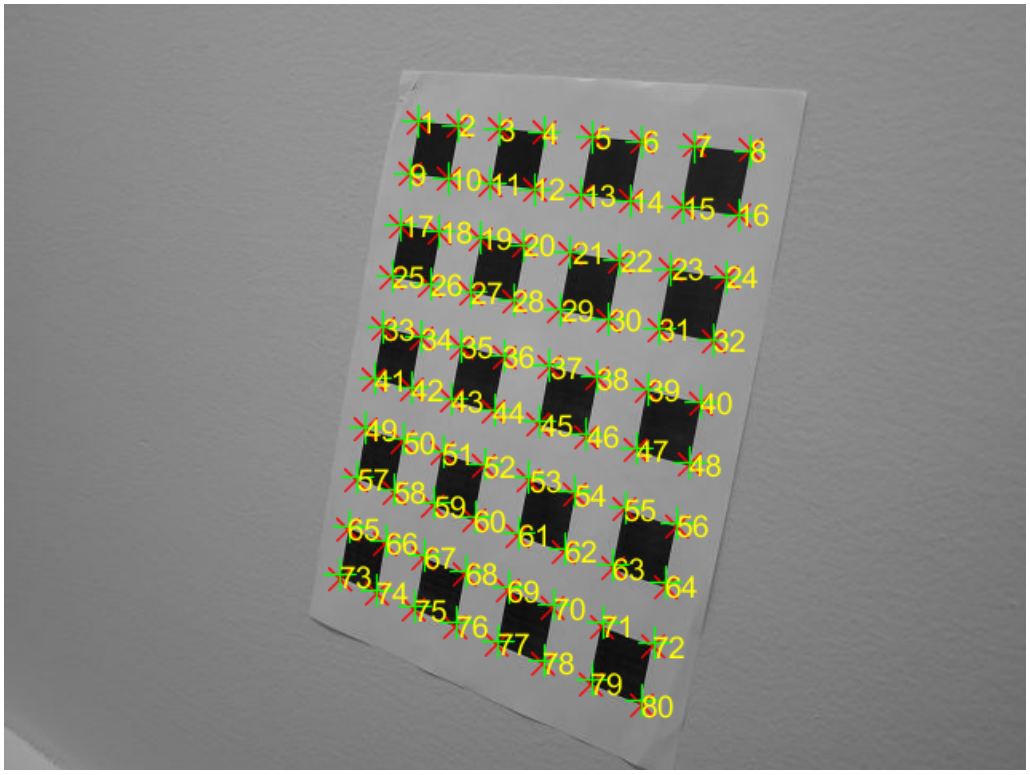
(a) Edges of 'Pic_1.jpg'



(b) Corners of 'Pic_1.jpg'



(c) LM vs LSE Re-projection of 'Pic_1.jpg'



(d) With vs Without radial Correction Re-projection of 'Pic_1.jpg'

Figure 1: Results of 'Pic_1.jpg'

2.1.2 Pic_3.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.7898 & -0.1804 & 0.5862 \\ 0.1977 & 0.9796 & 0.0350 \\ -0.5806 & 0.0883 & 0.8094 \end{bmatrix}$$

$$t = \begin{bmatrix} -46.8456 \\ -125.2711 \\ 551.7947 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.7873 & -0.1824 & 0.5890 \\ 0.1965 & 0.9797 & 0.0407 \\ -0.5844 & 0.0836 & 0.8071 \end{bmatrix}$$

$$t = \begin{bmatrix} -49.5042 \\ -125.5576 \\ 548.2087 \end{bmatrix}$$

- LM w/ radial Distortion

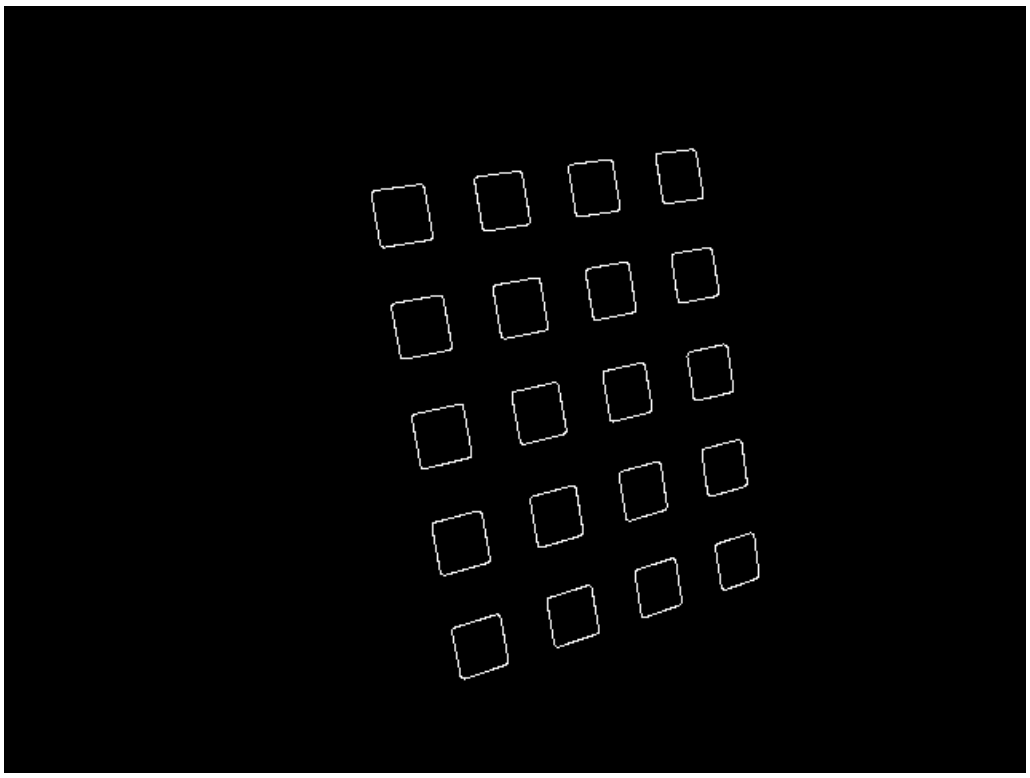
$$R = \begin{bmatrix} 0.7898 & -0.1804 & 0.5862 \\ 0.1977 & 0.9796 & 0.0350 \\ -0.5806 & 0.0883 & 0.8094 \end{bmatrix}$$

$$t = \begin{bmatrix} -46.8456 \\ -125.2711 \\ 551.7947 \end{bmatrix}$$

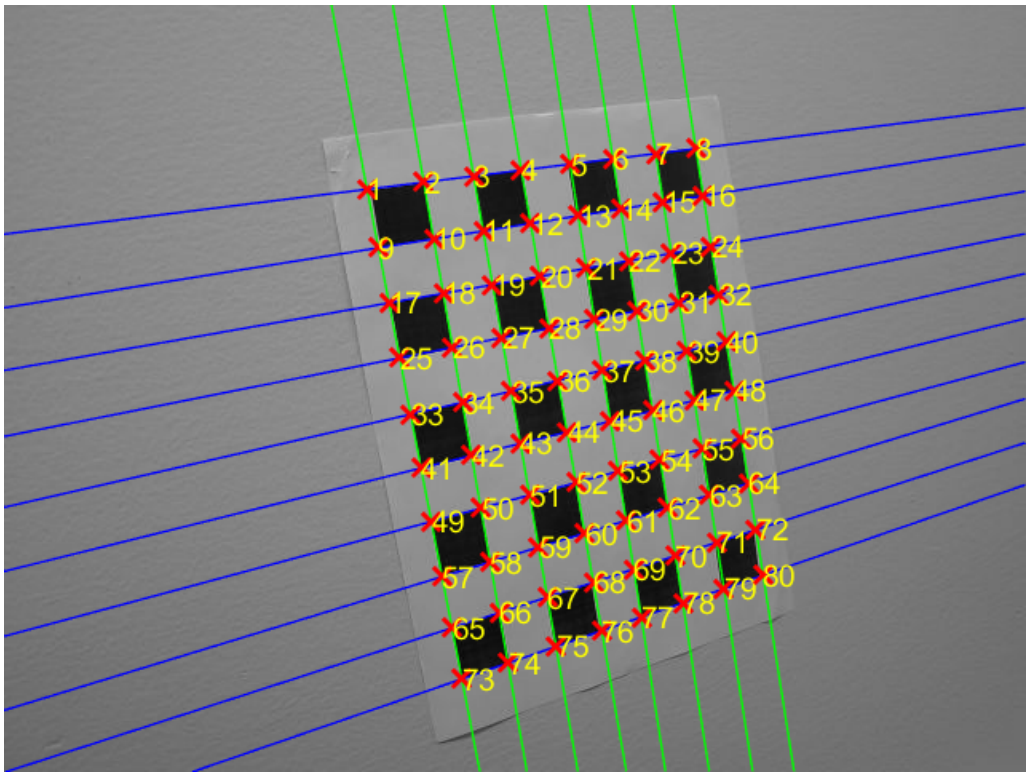
Table.2 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	1.9327	1.4960
LM	0.6141	0.1540
LM w/ radial Distortion	1.9327	1.4960

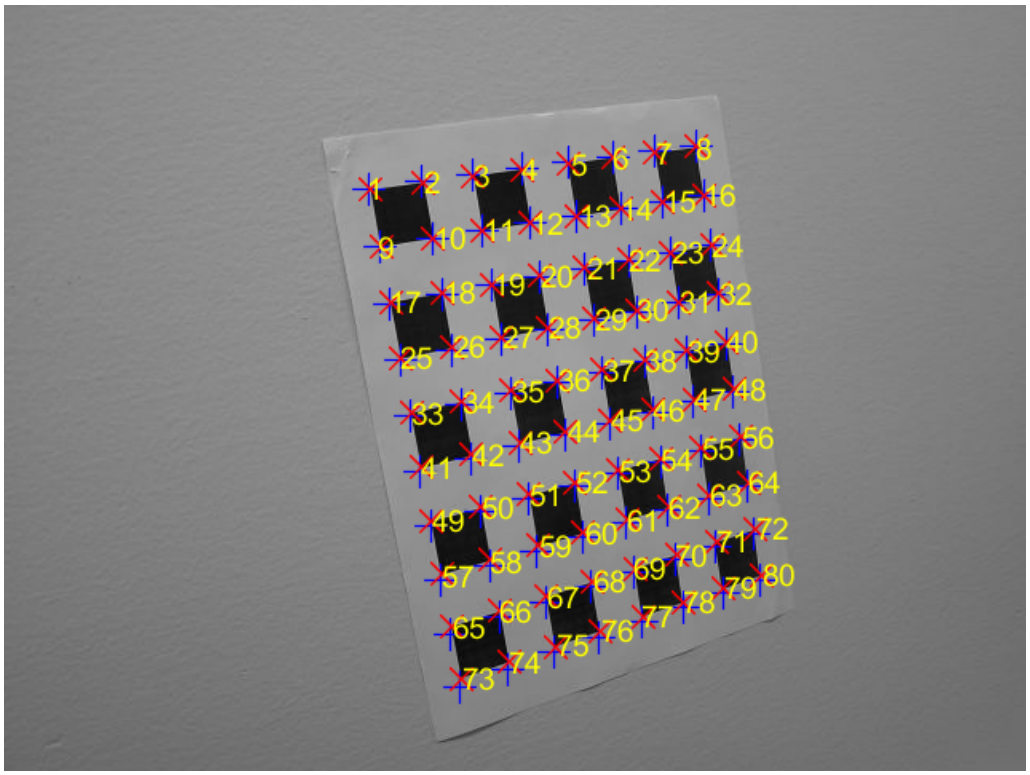
Table 2: Quantitative Comparison Between Different Method



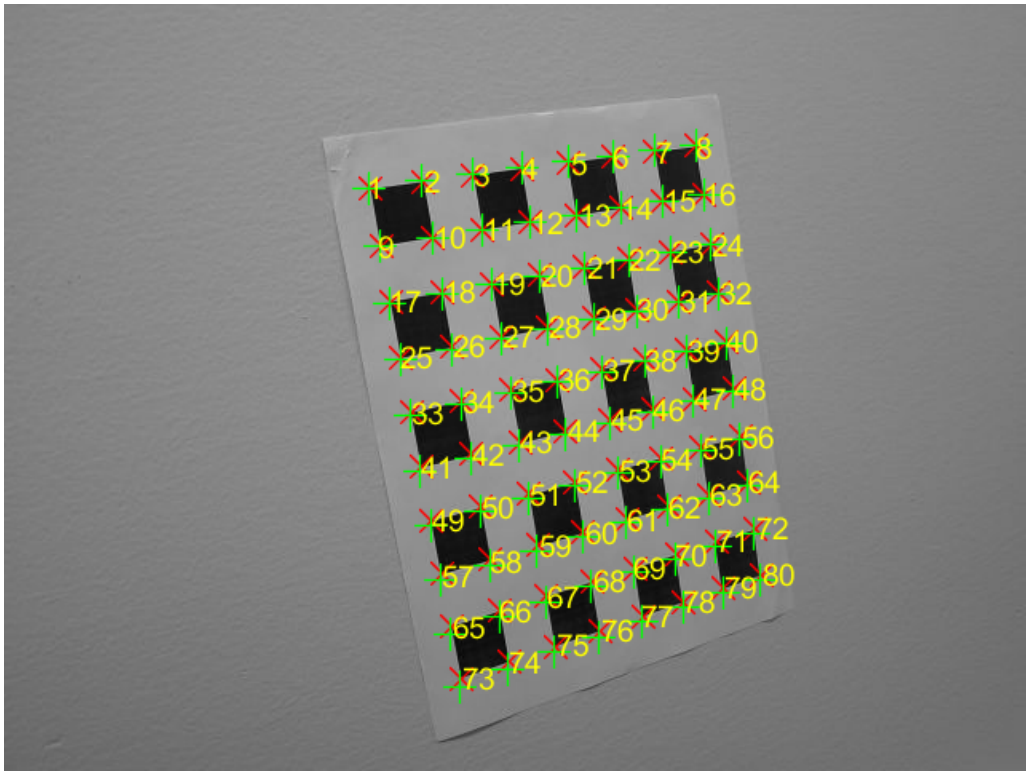
(a) Edges of 'Pic_3.jpg'



(b) Corners of 'Pic_3.jpg'



(c) LM vs LSE Re-projection of 'Pic_3.jpg'



(d) With vs Without radial Correction Re-projection of 'Pic_3.jpg'

Figure 2: Results of 'Pic_3.jpg'

2.1.3 Pic_12.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.8838 & 0.2183 & -0.4139 \\ -0.0186 & 0.9002 & 0.4350 \\ 0.4675 & -0.3767 & 0.7997 \end{bmatrix}$$

$$t = \begin{bmatrix} -68.9014 \\ -117.9828 \\ 533.7641 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.8851 & 0.2193 & -0.4104 \\ -0.0199 & 0.8990 & 0.4376 \\ 0.4649 & -0.3791 & 0.8001 \end{bmatrix}$$

$$t = \begin{bmatrix} -72.1378 \\ -119.8747 \\ 536.6060 \end{bmatrix}$$

- LM w/ radial Distortion

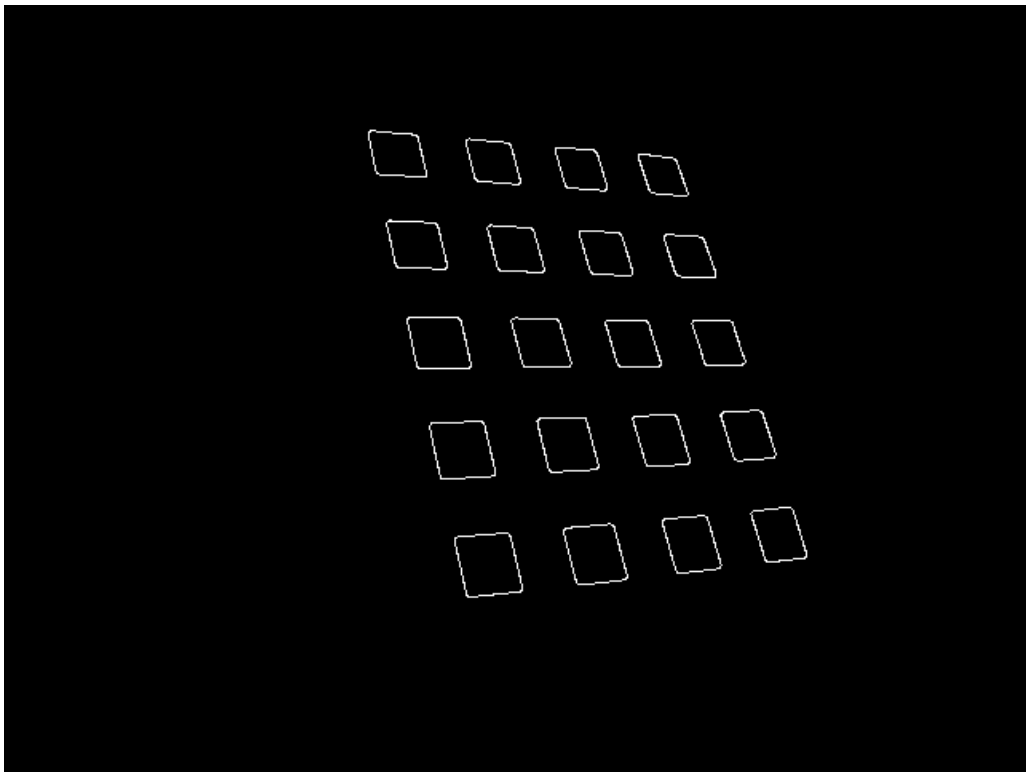
$$R = \begin{bmatrix} 0.8838 & 0.2183 & -0.4139 \\ -0.0186 & 0.9002 & 0.4350 \\ 0.4675 & -0.3767 & 0.7997 \end{bmatrix}$$

$$t = \begin{bmatrix} -68.9014 \\ -117.9828 \\ 533.7641 \end{bmatrix}$$

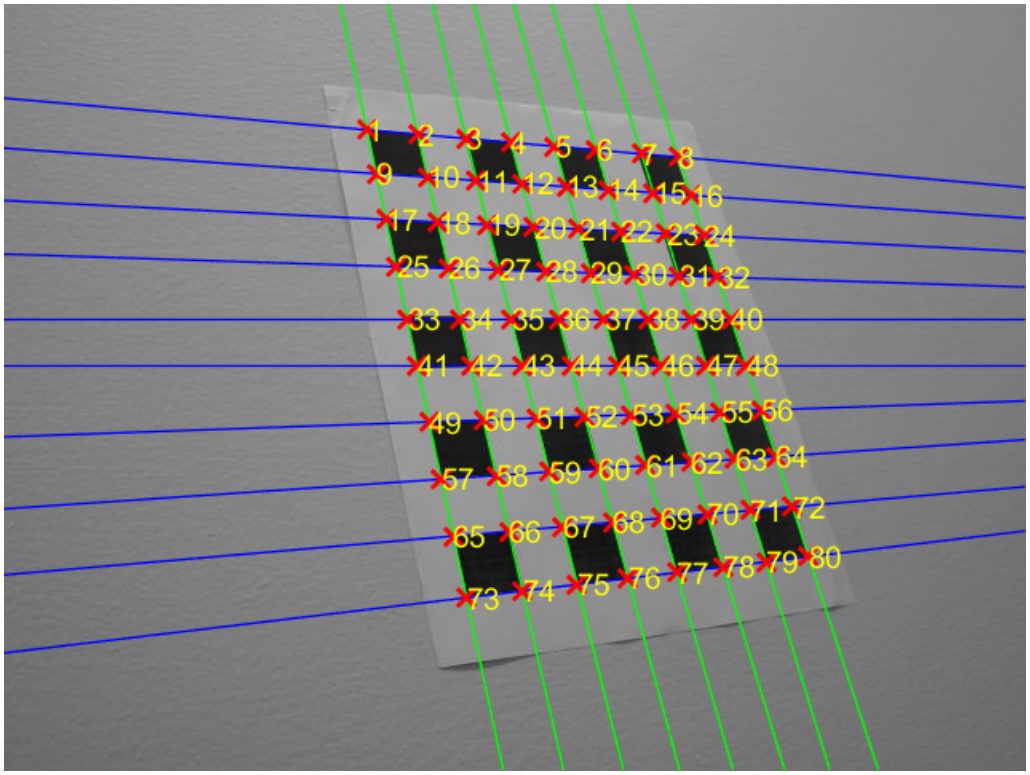
Table.3 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	1.3963	1.4960
LM	0.5785	0.1685
LM w/ radial Distortion	1.3963	1.5291

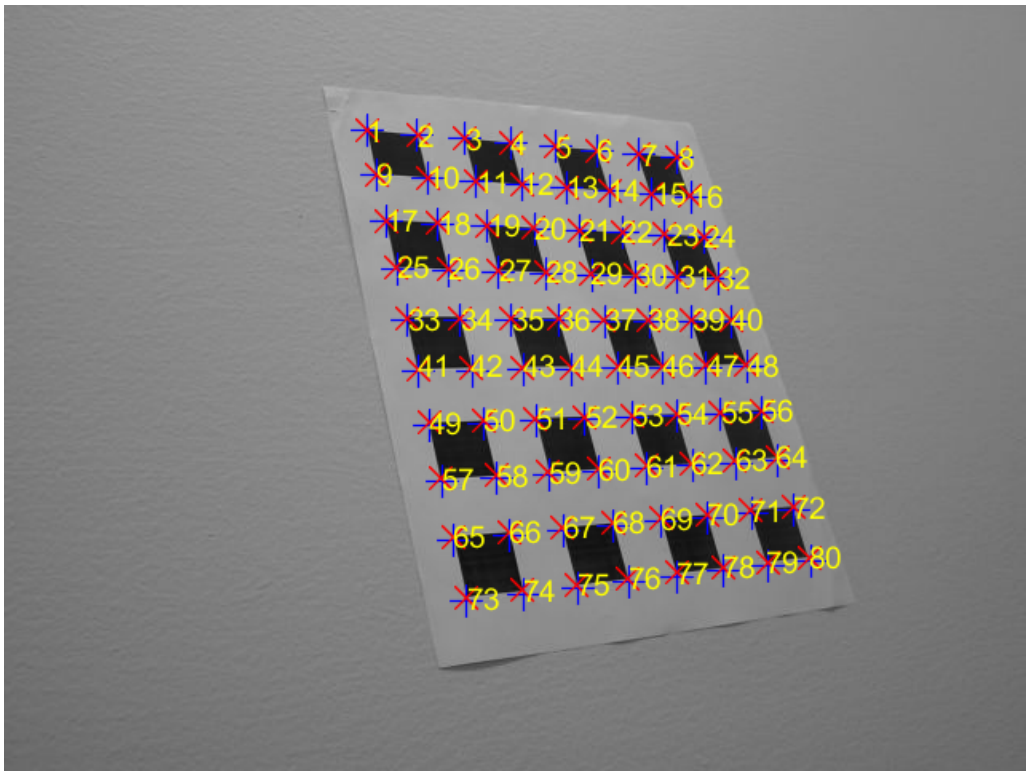
Table 3: Quantitative Comparison Between Different Method



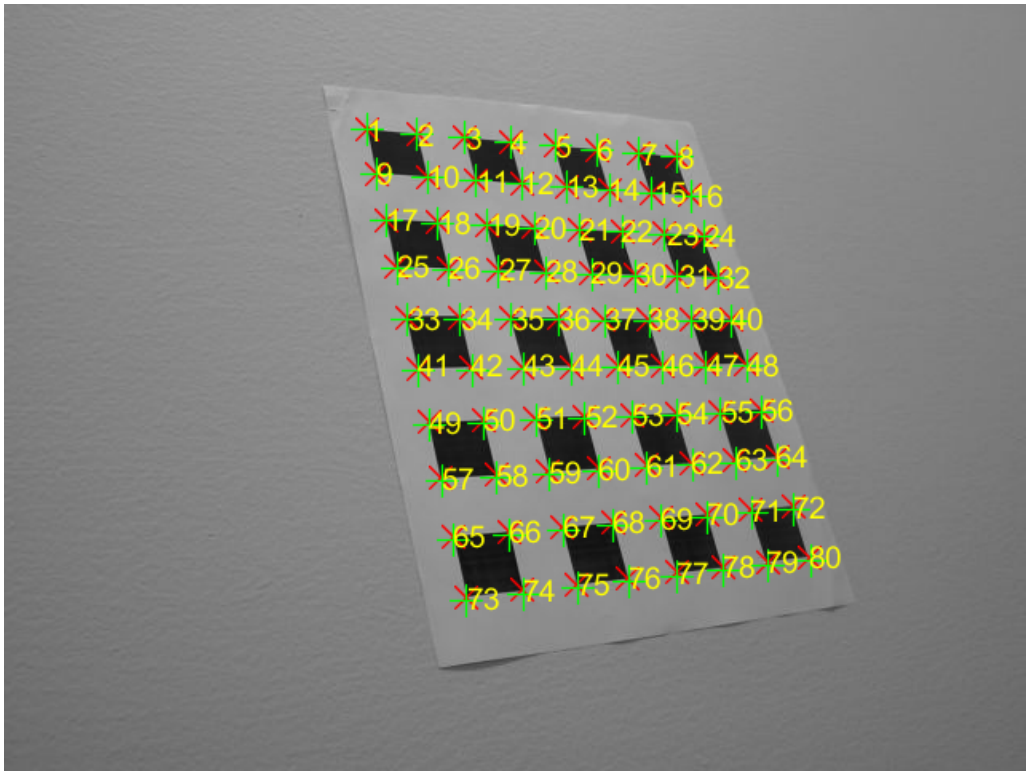
(a) Edges of 'Pic_12.jpg'



(b) Corners of 'Pic_12.jpg'



(c) LM vs LSE Re-projection of 'Pic_12.jpg'



(d) With vs Without radial Correction Re-projection of 'Pic_12.jpg'

Figure 3: Results of 'Pic_12.jpg'

2.1.4 Pic_15.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.9148 & 0.1723 & 0.3653 \\ -0.1573 & 0.9850 & -0.0707 \\ -0.3721 & 0.0072 & 0.9282 \end{bmatrix}$$

$$t = \begin{bmatrix} -112.6418 \\ -95.5289 \\ 480.2895 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.9149 & 0.1711 & 0.3657 \\ -0.1570 & 0.9852 & -0.0683 \\ -0.3720 & 0.0051 & 0.9282 \end{bmatrix}$$

$$t = \begin{bmatrix} -115.0555 \\ -96.4192 \\ 478.9348 \end{bmatrix}$$

- LM w/ radial Distortion

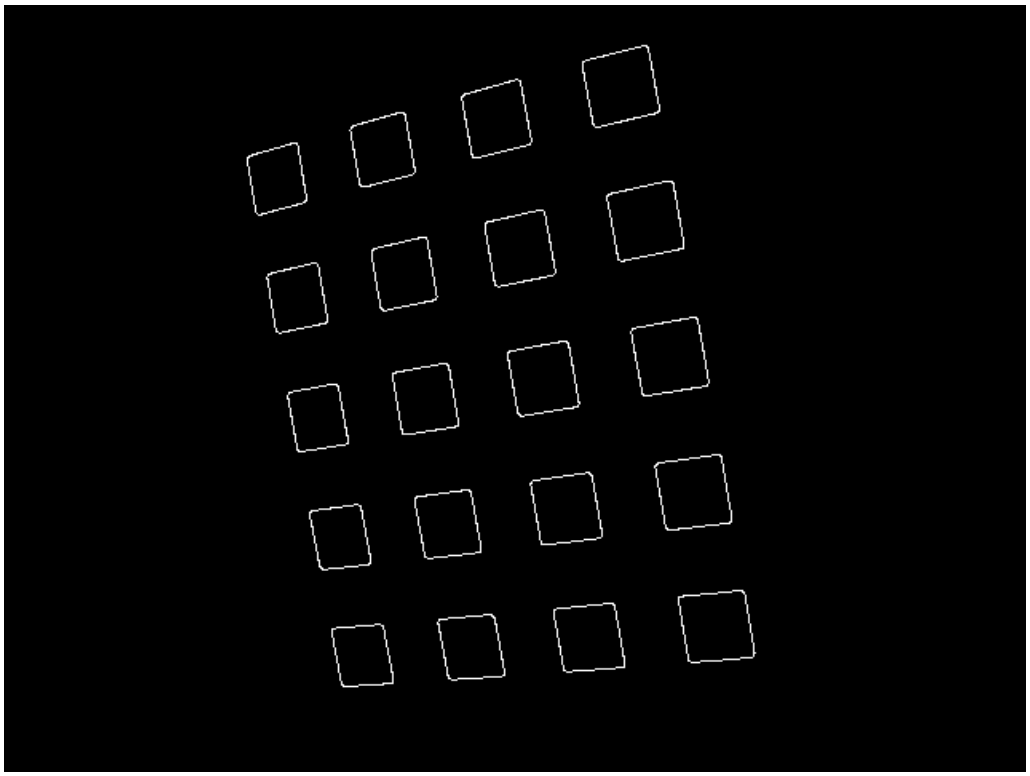
$$R = \begin{bmatrix} 0.9148 & 0.1723 & 0.3653 \\ -0.1573 & 0.9850 & -0.0707 \\ -0.3721 & 0.0072 & 0.9282 \end{bmatrix}$$

$$t = \begin{bmatrix} -112.641767410138 \\ -95.5288702263621 \\ 480.289546393523 \end{bmatrix}$$

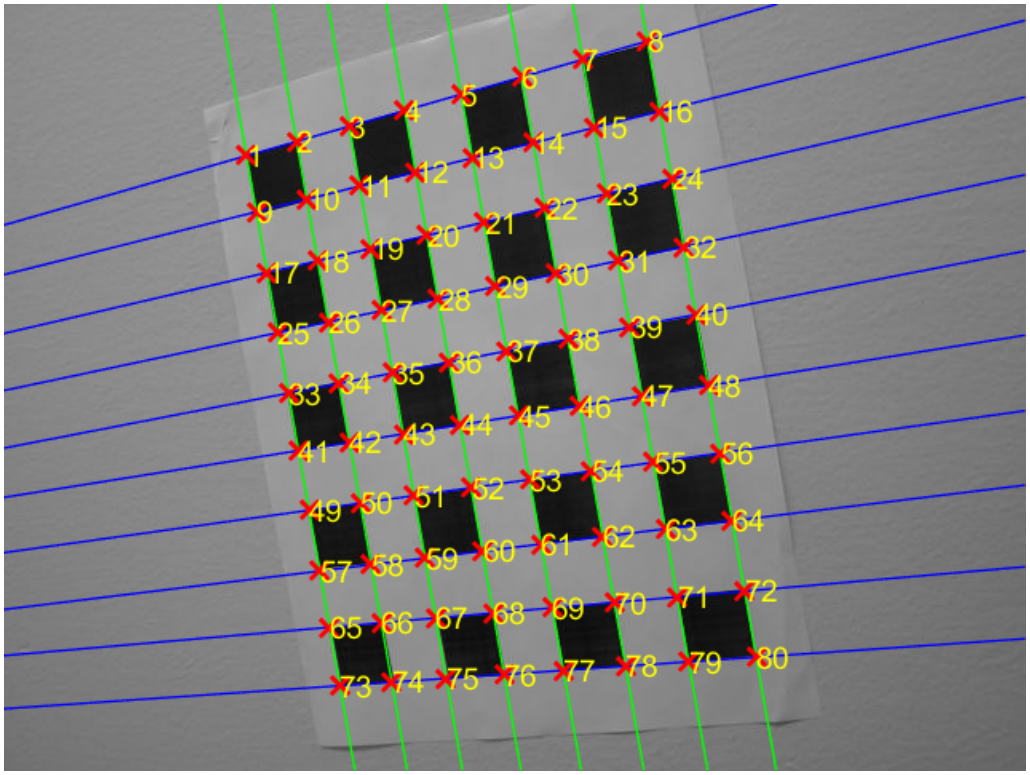
Table.4 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	0.7194	0.1523
LM	0.5451	0.0754
LM w/ radial Distortion	0.7194	0.1523

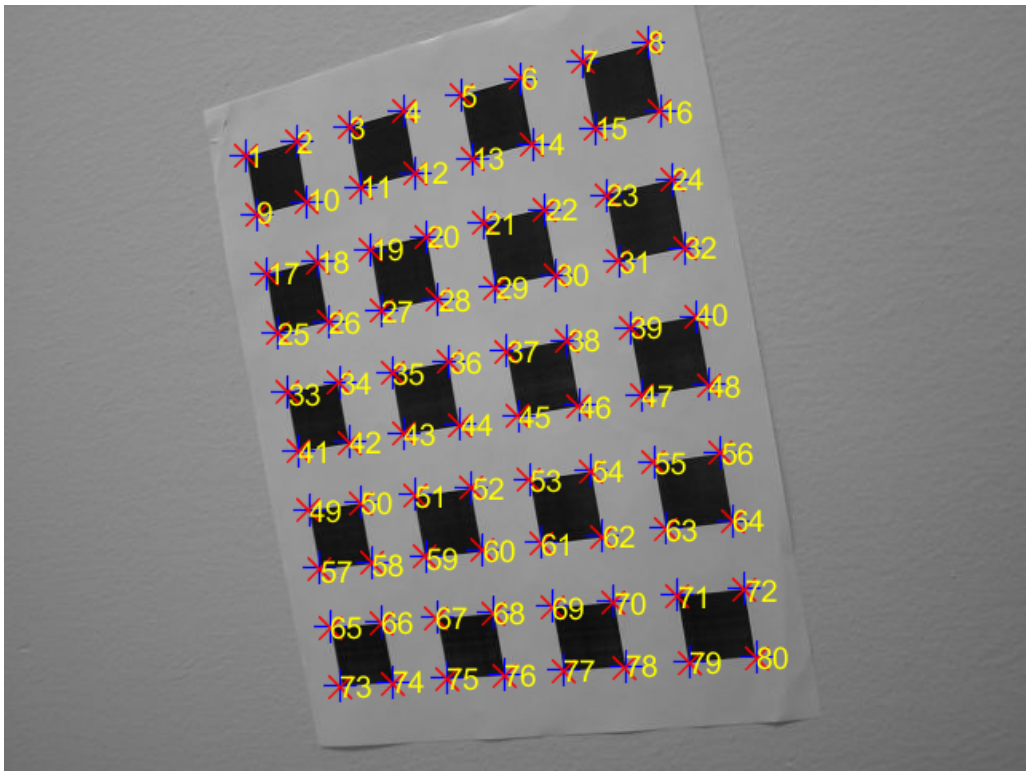
Table 4: Quantitative Comparison Between Different Method



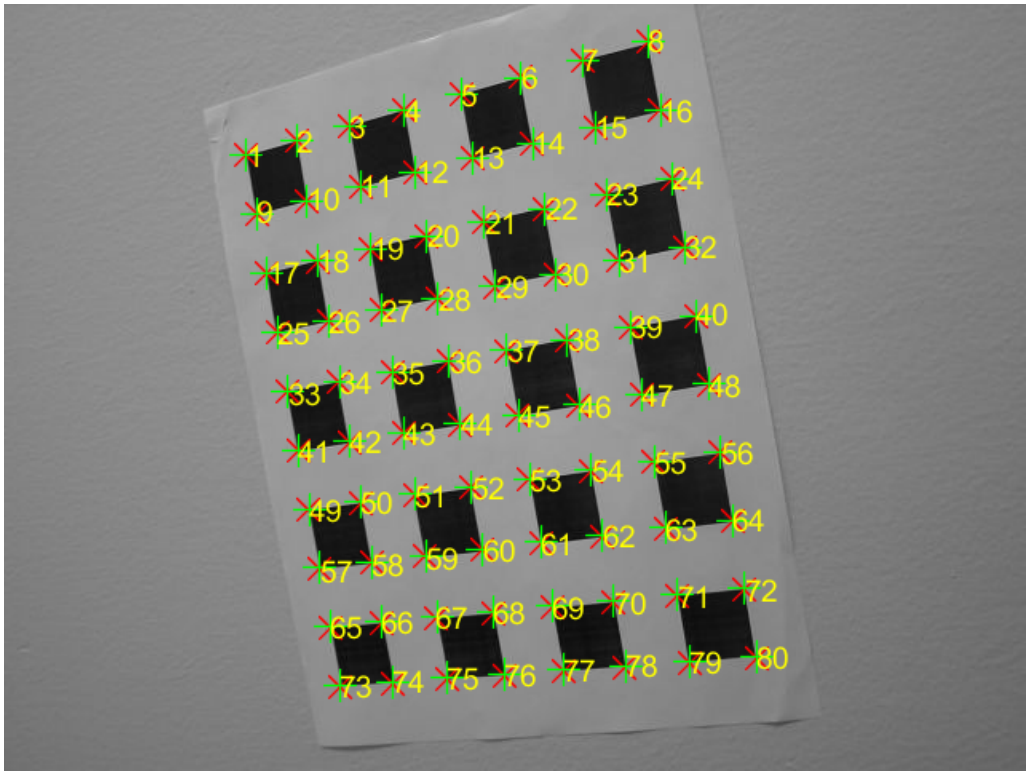
(a) Edges of 'Pic_15.jpg'



(b) Corners of 'Pic_15.jpg'



(c) LM vs LSE Re-projection of 'Pic_15.jpg'



(d) With vs Without radial Correction Re-projection of 'Pic_15.jpg'

Figure 4: Results of 'Pic_15.jpg'

2.1.5 Re-projection to 'Pic_11.jpg'

We select 'Pic_11.jpg' as the Fixed Image and re-project by the parameters from LM method. The extrinsic parameters of 'Pic_11.jpg' are

$$R = \begin{bmatrix} 0.9995 & -0.0084 & 0.0300 \\ 0.0096 & 0.9990 & -0.0429 \\ -0.0296 & 0.0432 & 0.9986 \end{bmatrix}$$

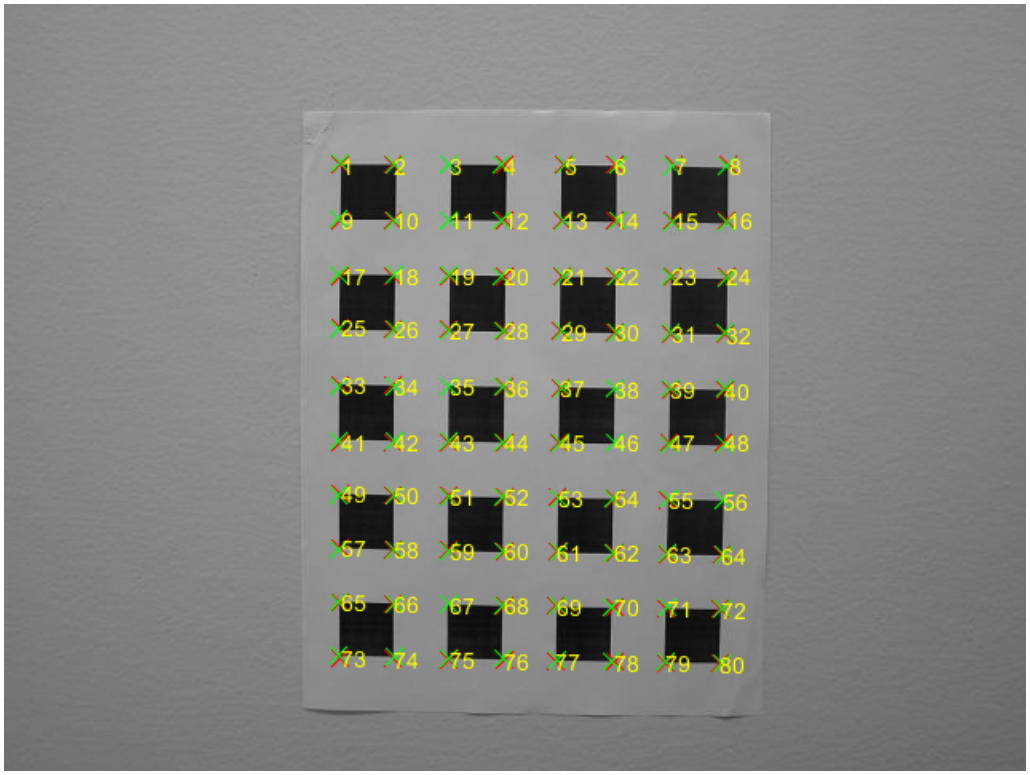
$$t = \begin{bmatrix} -83.0047 \\ -100.4916 \\ 520.3042 \end{bmatrix}$$

Table.5 is the quantitative comparison of different re-projections.

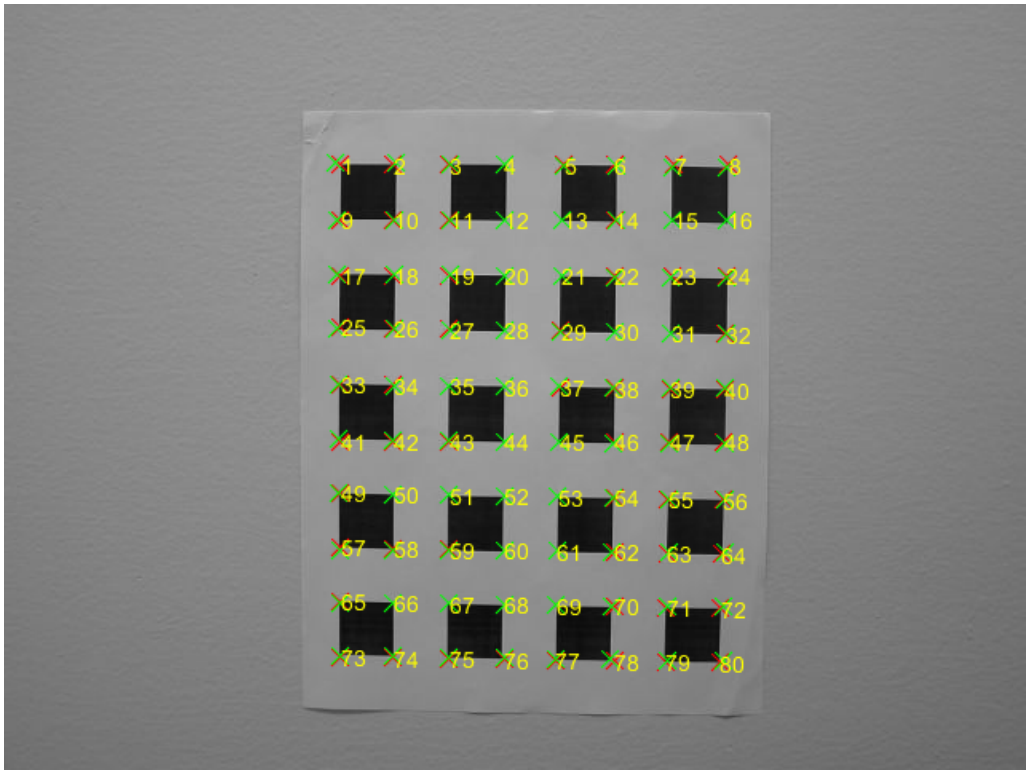
Target	mean	var
'Pic_1.jpg'	0.9644	0.1564
'Pic_3.jpg'	1.0067	0.5279
'Pic_12.jpg'	1.0904	0.4123
'Pic_15.jpg'	1.0054	0.2046

Table 5: Quantitative Comparison

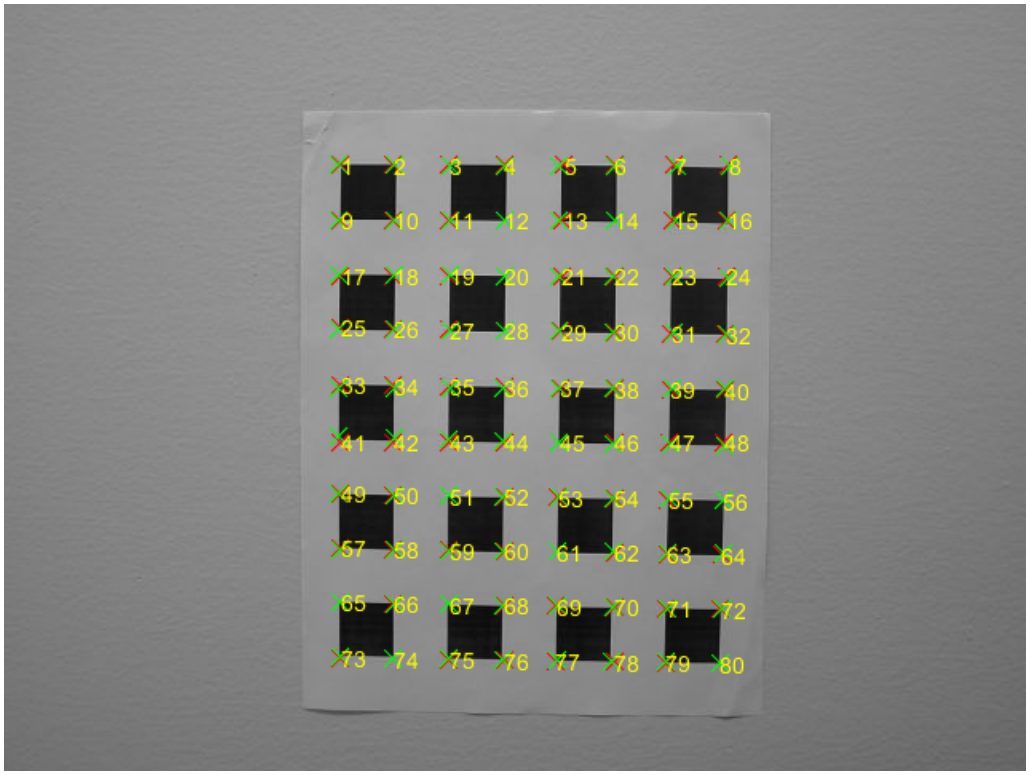
Fig.5 shows the results, where red '×' are the true corners and the green '×' are the re-projected corners.



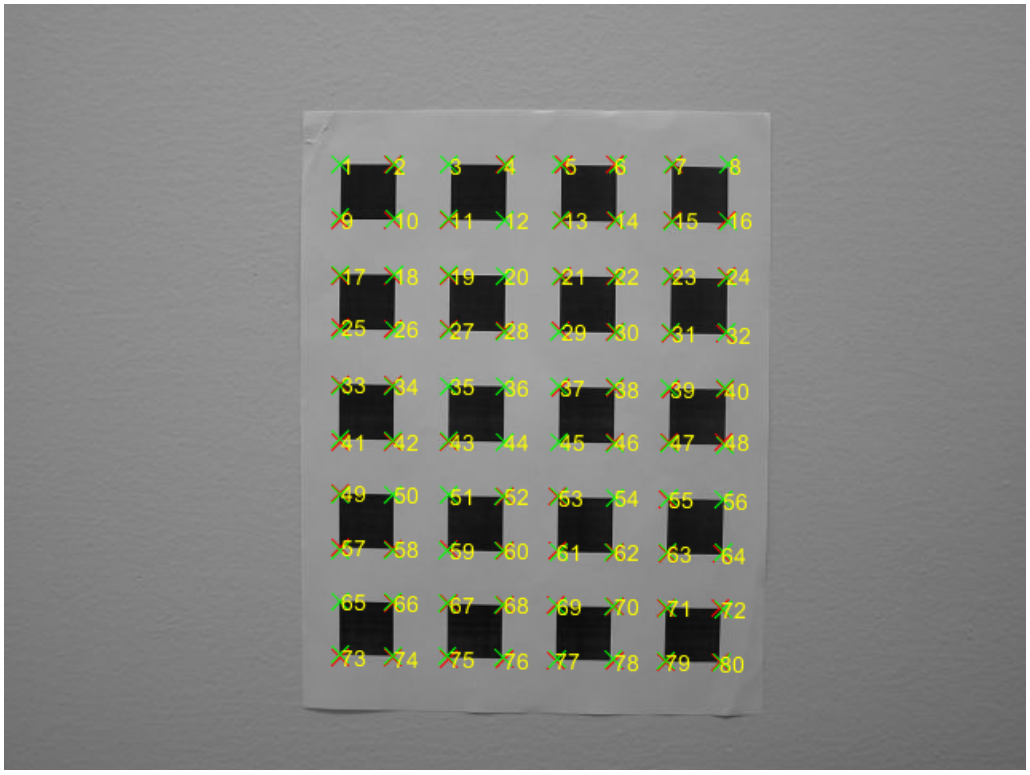
(a) Re-projection from 'Pic_1.jpg' to 'Pic_11.jpg'



(b) Re-projection from 'Pic_3.jpg' to 'Pic_11.jpg'



(c) Re-projection from 'Pic_12.jpg' to 'Pic_11.jpg'



(d) Re-projection from 'Pic_15.jpg' to 'Pic_11.jpg'

Figure 5: Re-projections to 'Pic_11.jpg'

2.2 My Dataset

The dataset includes 22 images taken by an iPhone 7 and downsampled to 640 × 480 for computation convenience.

- LSE

$$K = \begin{bmatrix} 549.76 & -0.96271 & 317.25 \\ 0 & 548.86 & 236.41 \\ 0 & 0 & 1 \end{bmatrix}$$

- LM

$$K = \begin{bmatrix} 546.35 & -1.2314 & 322.07 \\ 0 & 545.15 & 236.02 \\ 0 & 0 & 1 \end{bmatrix}$$

- LM w/ radial Distortion

$$K = \begin{bmatrix} 549.76 & -0.96271 & 317.25 \\ 0 & 548.86 & 236.41 \\ 0 & 0 & 1 \end{bmatrix}$$

$$k_1 = 1.6039e - 09$$

$$k_2 = -1.0003e - 14$$

2.2.1 IMG_1755.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.99952 & 0.016058 & 0.026528 \\ -0.0066737 & 0.94682 & -0.32168 \\ -0.030283 & 0.32135 & 0.94648 \end{bmatrix}$$

$$t = \begin{bmatrix} -81.308 \\ -65.303 \\ 287.45 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.99978 & 0.012678 & 0.016717 \\ -0.0066618 & 0.94737 & -0.32008 \\ -0.019895 & 0.31989 & 0.94724 \end{bmatrix}$$

$$t = \begin{bmatrix} -73.0 \\ -79.62 \\ 295.46 \end{bmatrix}$$

- LM w/ radial Distortion

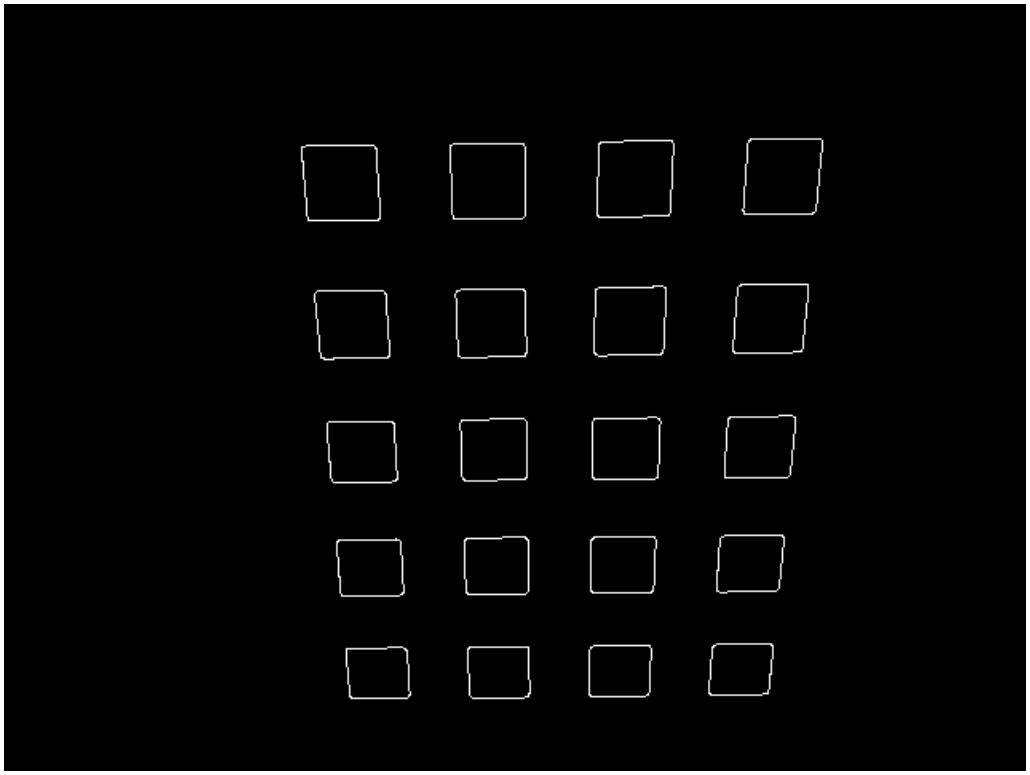
$$R = \begin{bmatrix} 0.99952 & 0.016058 & 0.026528 \\ -0.0066737 & 0.94682 & -0.32168 \\ -0.030283 & 0.32135 & 0.94648 \end{bmatrix}$$

$$t = \begin{bmatrix} -70.298 \\ -79.779 \\ 298.76 \end{bmatrix}$$

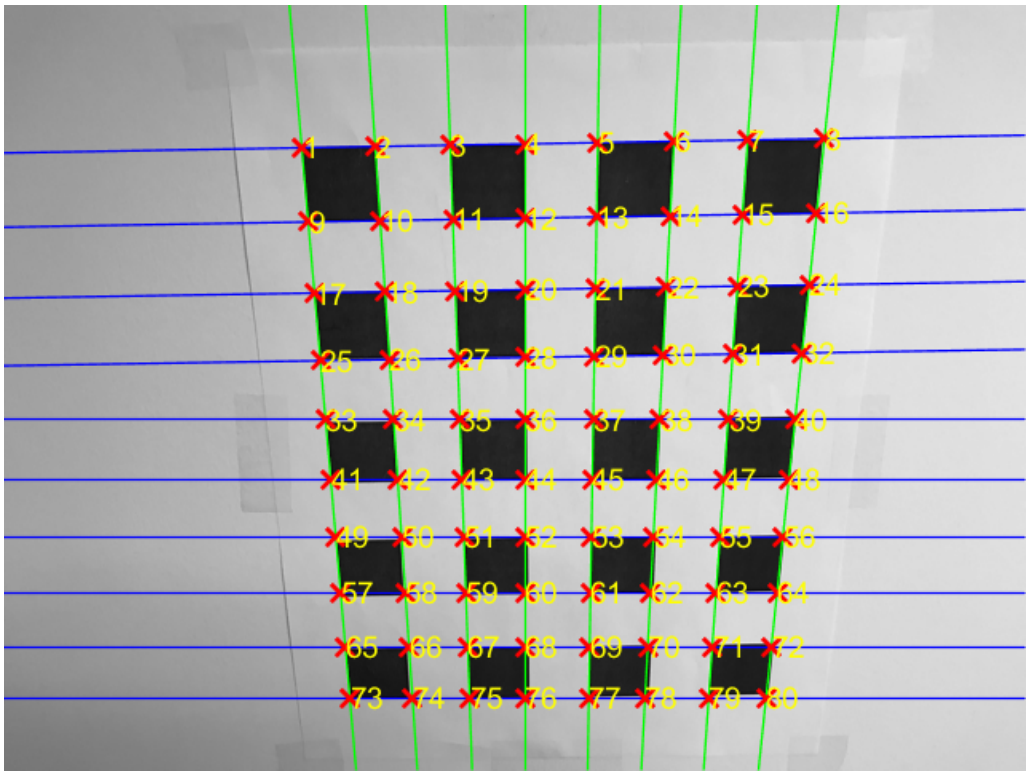
Table.6 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	0.9565	0.3176
LM	0.9176	0.2203
LM w/ radial Distortion	0.9565	0.3176

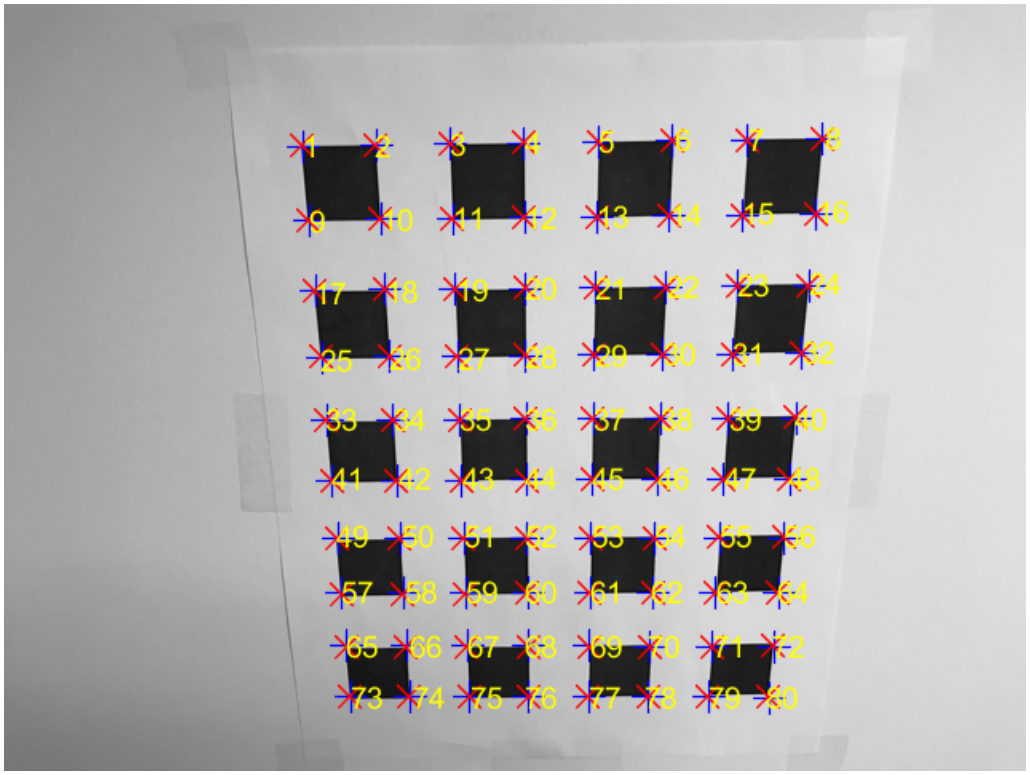
Table 6: Quantitative Comparison Between Different Method



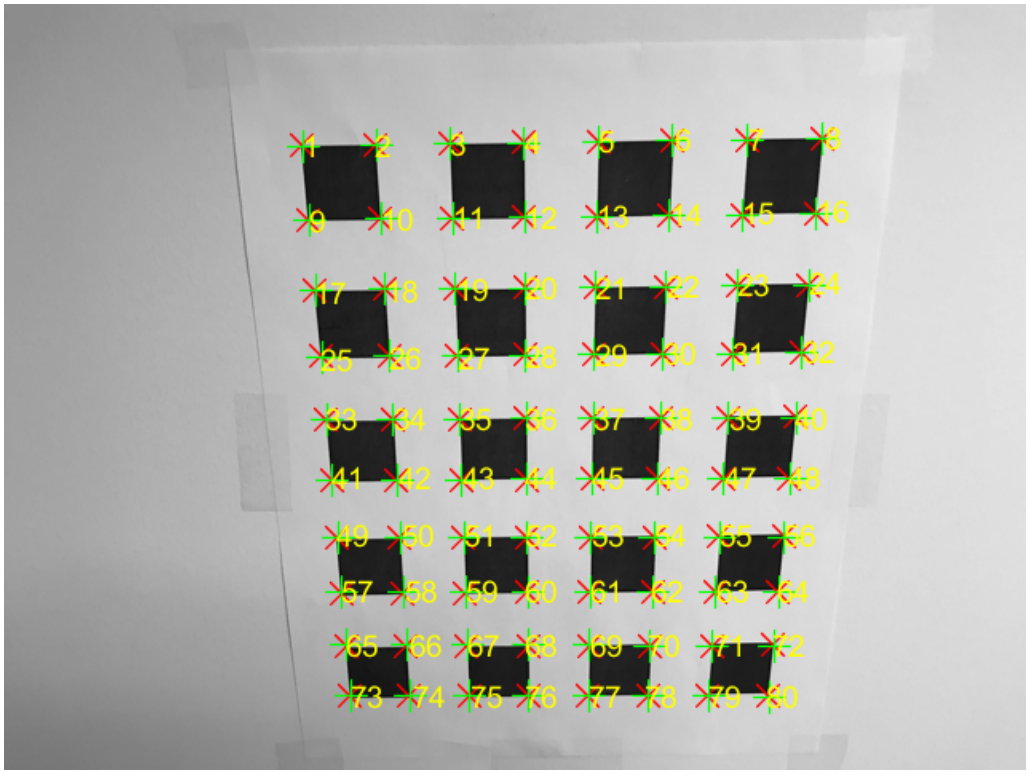
(a) Edges of 'IMG_1755.jpg'



(b) Corners of 'IMG_1755.jpg'



(c) LM vs LSE Re-projection of 'IMG_1755.jpg'



(d) With vs Without radial Correction Re-projection of 'IMG_1755.jpg'

Figure 6: Results of 'IMG_1755.jpg'

2.2.2 IMG_1759.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.99977 & -0.016515 & -0.013945 \\ 0.01882 & 0.9824 & 0.18586 \\ 0.01063 & -0.18608 & 0.98248 \end{bmatrix}$$

$$t = \begin{bmatrix} -70.104 \\ -116.21 \\ 328.61 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.99983 & -0.016594 & -0.0074595 \\ 0.017664 & 0.9836 & 0.17948 \\ 0.004359 & -0.17958 & 0.98373 \end{bmatrix}$$

$$t = \begin{bmatrix} -73.553 \\ -116.23 \\ 326.36 \end{bmatrix}$$

- LM w/ radial Distortion

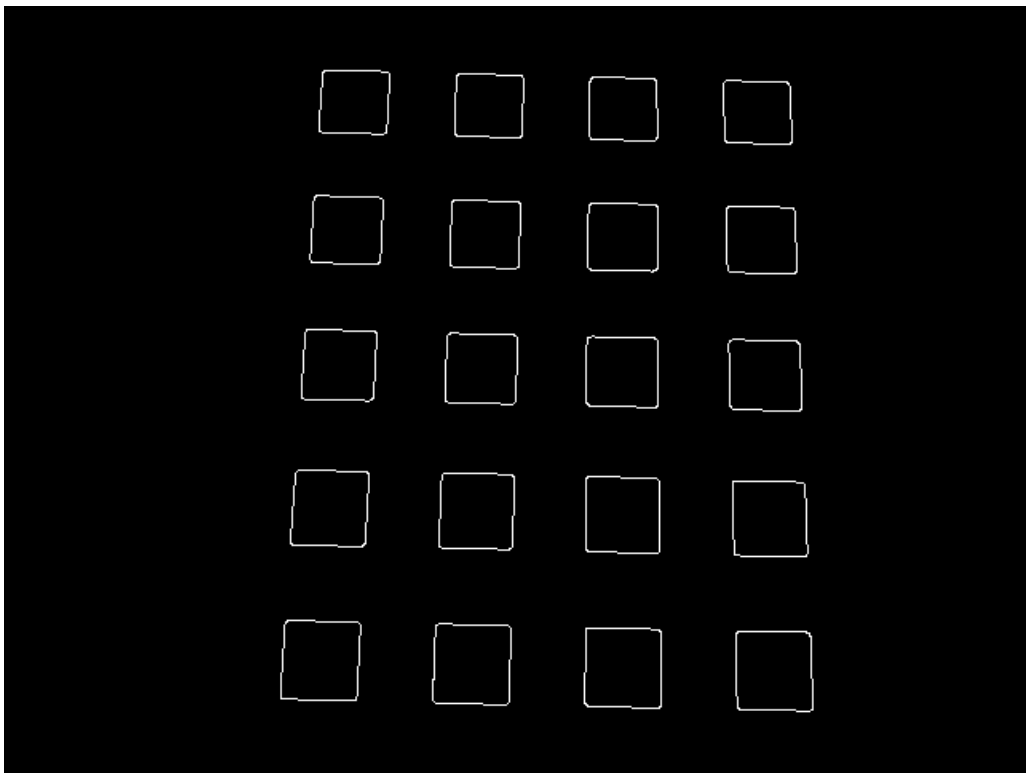
$$R = \begin{bmatrix} 0.99977 & -0.016515 & -0.013945 \\ 0.01882 & 0.9824 & 0.18586 \\ 0.01063 & -0.18608 & 0.98248 \end{bmatrix}$$

$$t = \begin{bmatrix} -70.104 \\ -116.21 \\ 328.61 \end{bmatrix}$$

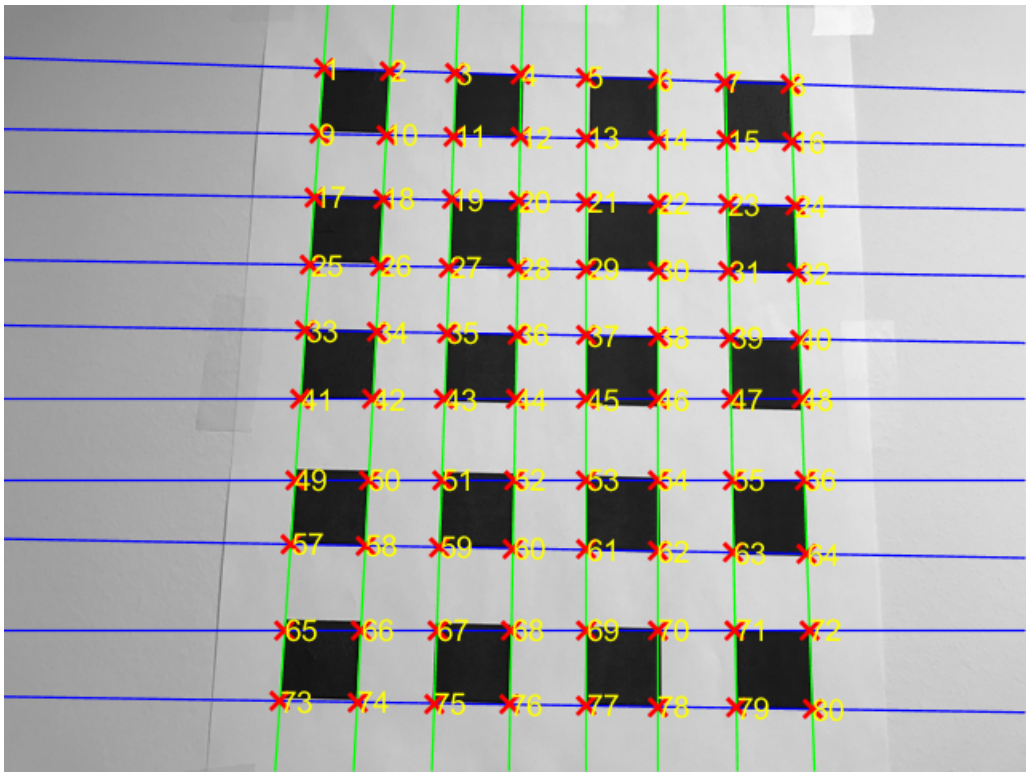
Table.7 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	2.1806	1.9543
LM	1.6155	1.9550
LM w/ radial Distortion	2.1806	1.9543

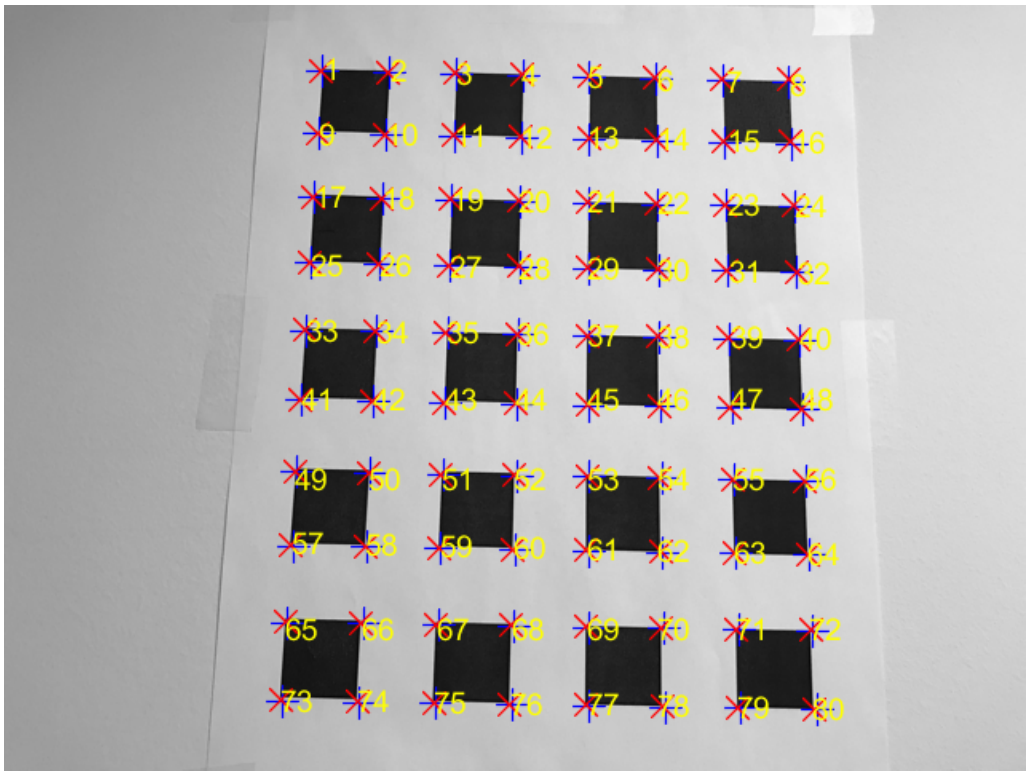
Table 7: Quantitative Comparison Between Different Method



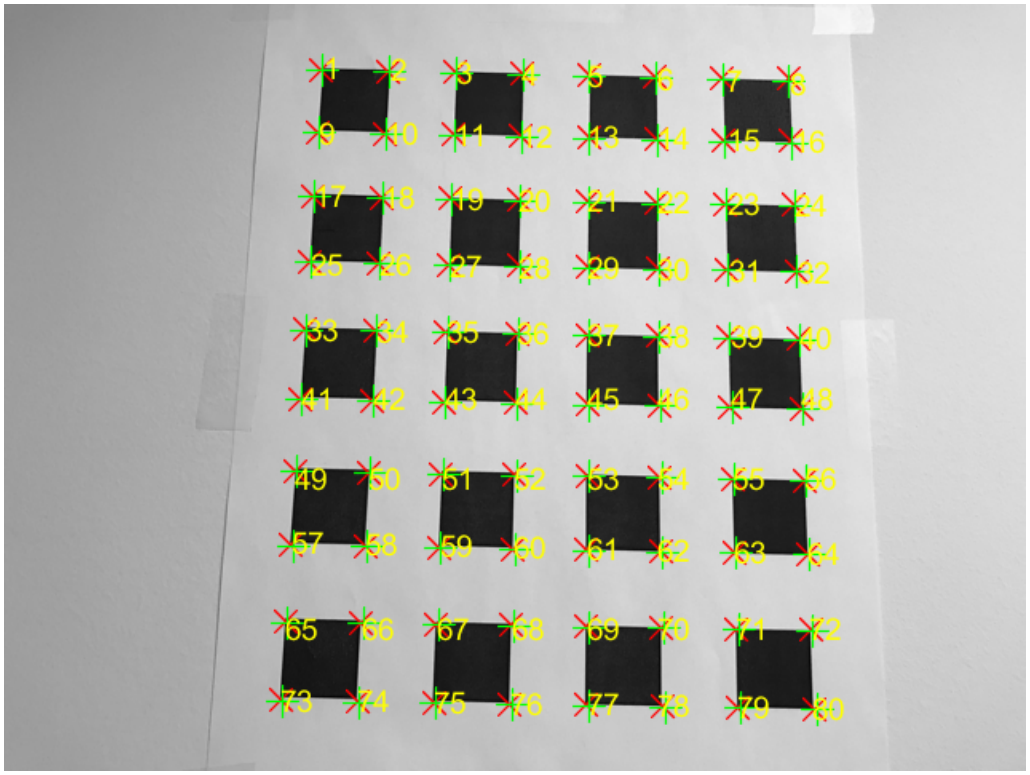
(a) Edges of 'IMG_1759.jpg'



(b) Corners of 'IMG_1759.jpg'



(c) LM vs LSE Re-projection of 'IMG_1759.jpg'



(d) With vs Without radial Correction Re-projection of 'IMG_1759.jpg'

Figure 7: Results of 'IMG_1759.jpg'

2.2.3 IMG_1767.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.94114 & -0.0014603 & -0.33803 \\ -0.0035472 & 0.99989 & -0.014196 \\ 0.33801 & 0.014559 & 0.94103 \end{bmatrix}$$

$$t = \begin{bmatrix} -69.876 \\ -108.28 \\ 279.5 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.93827 & -0.00044957 & -0.34589 \\ -0.0031388 & 0.99995 & -0.009814 \\ 0.34588 & 0.010294 & 0.93822 \end{bmatrix}$$

$$t = \begin{bmatrix} -72.207 \\ -107.76 \\ 276.89 \end{bmatrix}$$

- LM w/ radial Distortion

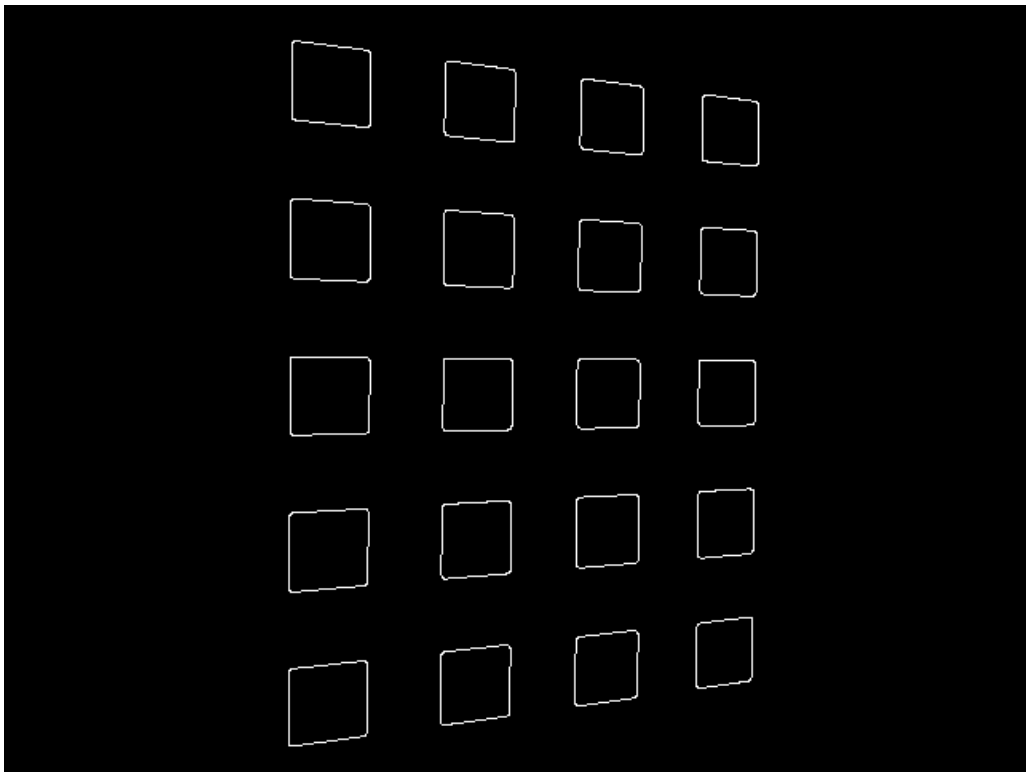
$$R = \begin{bmatrix} 0.94114 & -0.0014603 & -0.33803 \\ -0.0035472 & 0.99989 & -0.014196 \\ 0.33801 & 0.014559 & 0.94103 \end{bmatrix}$$

$$t = \begin{bmatrix} -69.876 \\ -108.28 \\ 279.5 \end{bmatrix}$$

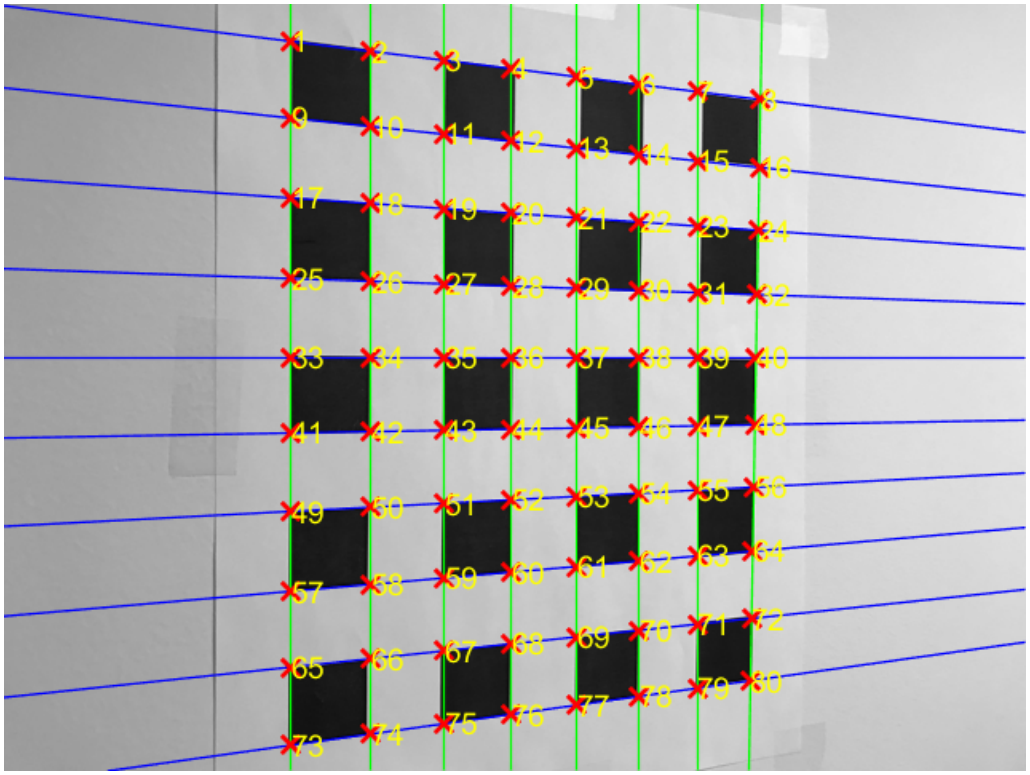
Table.8 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	1.2084	0.3468
LM	0.9539	0.2611
LM w/ radial Distortion	1.2084	0.3469

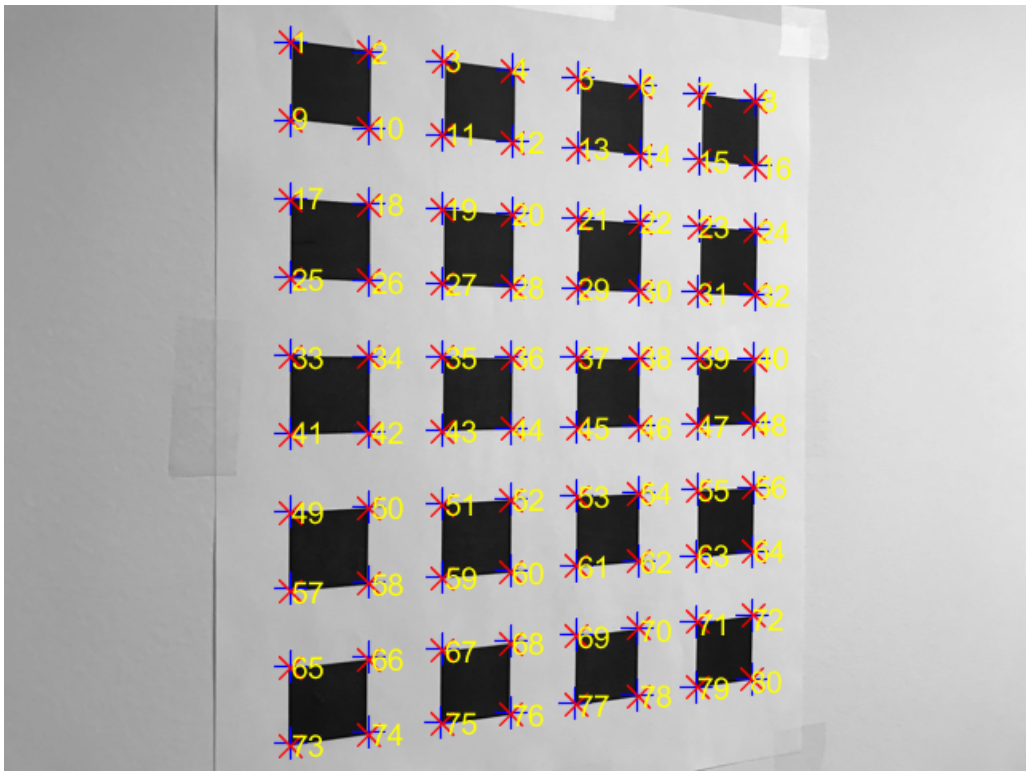
Table 8: Quantitative Comparison Between Different Method



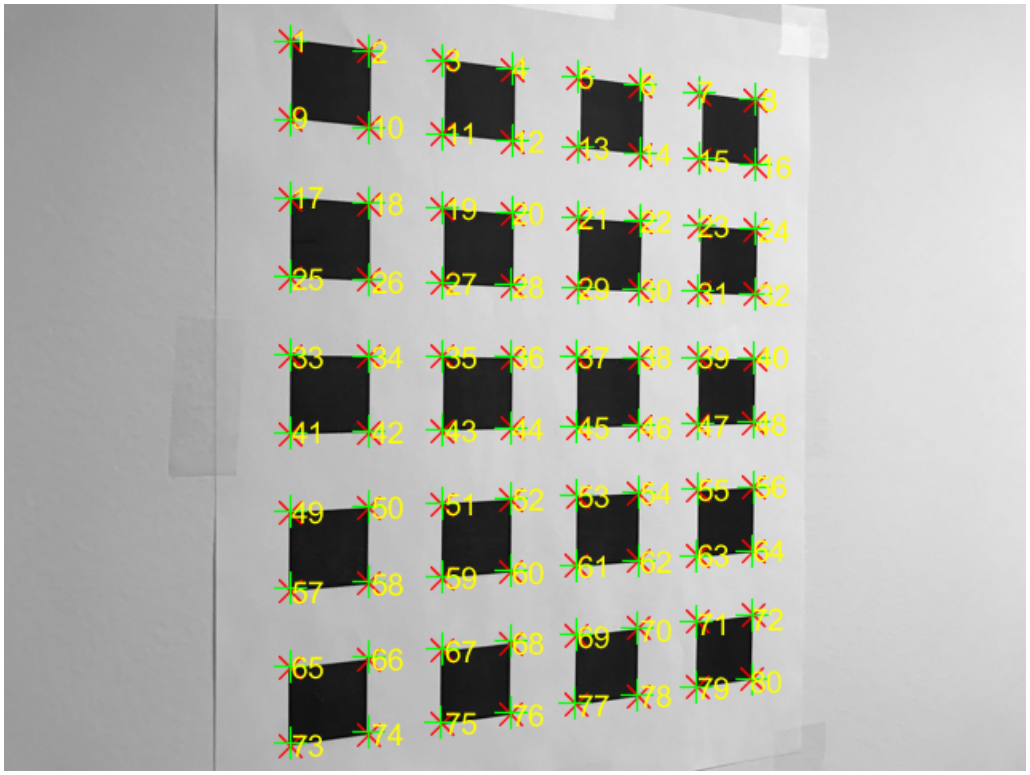
(a) Edges of 'IMG_1767.jpg'



(b) Corners of 'IMG_1767.jpg'



(c) LM vs LSE Re-projection of 'IMG_1767.jpg'



(d) With vs Without radial Correction Re-projection of 'IMG_1767.jpg'

Figure 8: Results of 'IMG_1767.jpg'

2.2.4 IMG_1773.jpg

The extrinsic parameters of different methods are listed as follows.

- LSE

$$R = \begin{bmatrix} 0.98142 & -0.0038482 & 0.19183 \\ 0.0086514 & 0.99967 & -0.024207 \\ -0.19168 & 0.025417 & 0.98113 \end{bmatrix}$$

$$t = \begin{bmatrix} -99.009 \\ -103.88 \\ 326.84 \end{bmatrix}$$

- LM

$$R = \begin{bmatrix} 0.98125 & -0.0026918 & 0.19272 \\ 0.0084907 & 0.99954 & -0.02927 \\ -0.19255 & 0.030358 & 0.98082 \end{bmatrix}$$

$$t = \begin{bmatrix} -101.58 \\ -103.11 \\ 323.61 \end{bmatrix}$$

- LM w/ radial Distortion

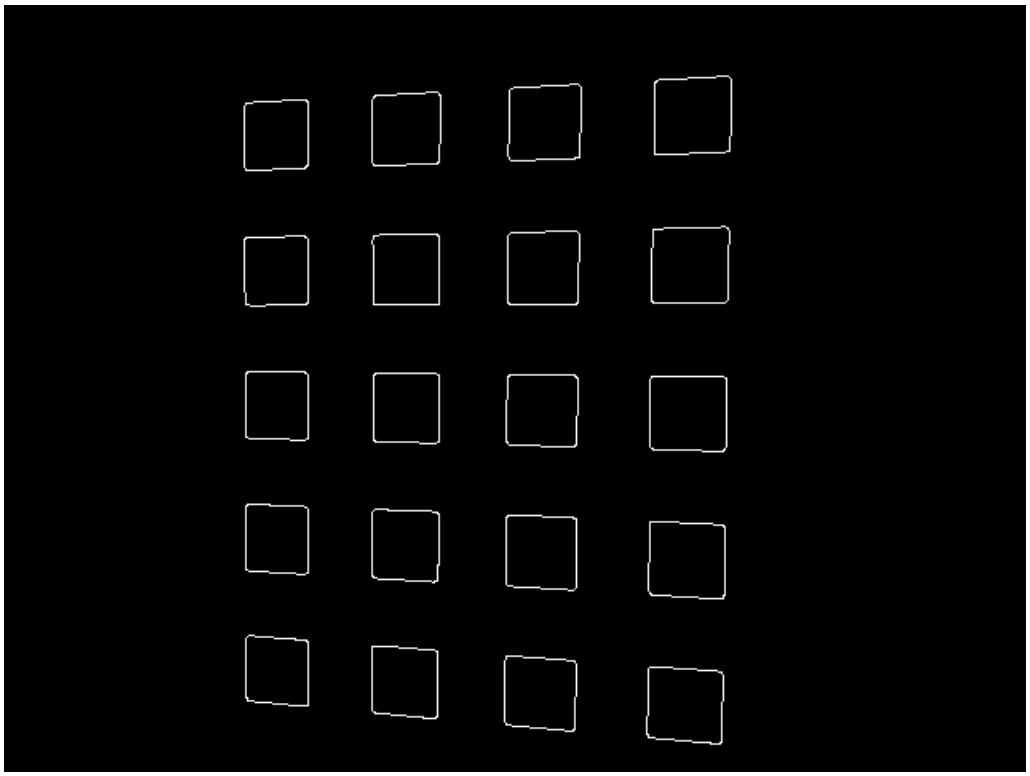
$$R = \begin{bmatrix} 0.98142 & -0.0038482 & 0.19183 \\ 0.0086514 & 0.99967 & -0.024207 \\ -0.19168 & 0.025417 & 0.98113 \end{bmatrix}$$

$$t = \begin{bmatrix} -99.009 \\ -103.88 \\ 326.84 \end{bmatrix}$$

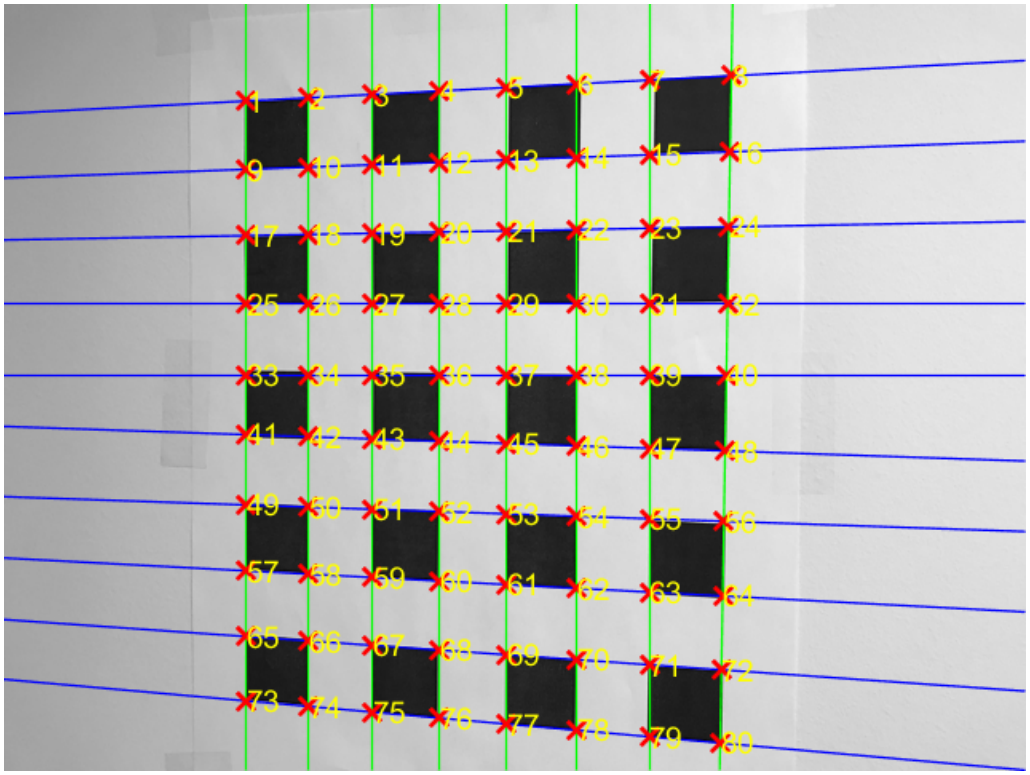
Table.9 is the quantitative comparison of different method via distance error.

Method	mean	var
LSE	1.3193	0.4613
LM	0.9315	0.3356
LM w radial Distortion	1.3193	0.4613

Table 9: Quantitative Comparison Between Different Method

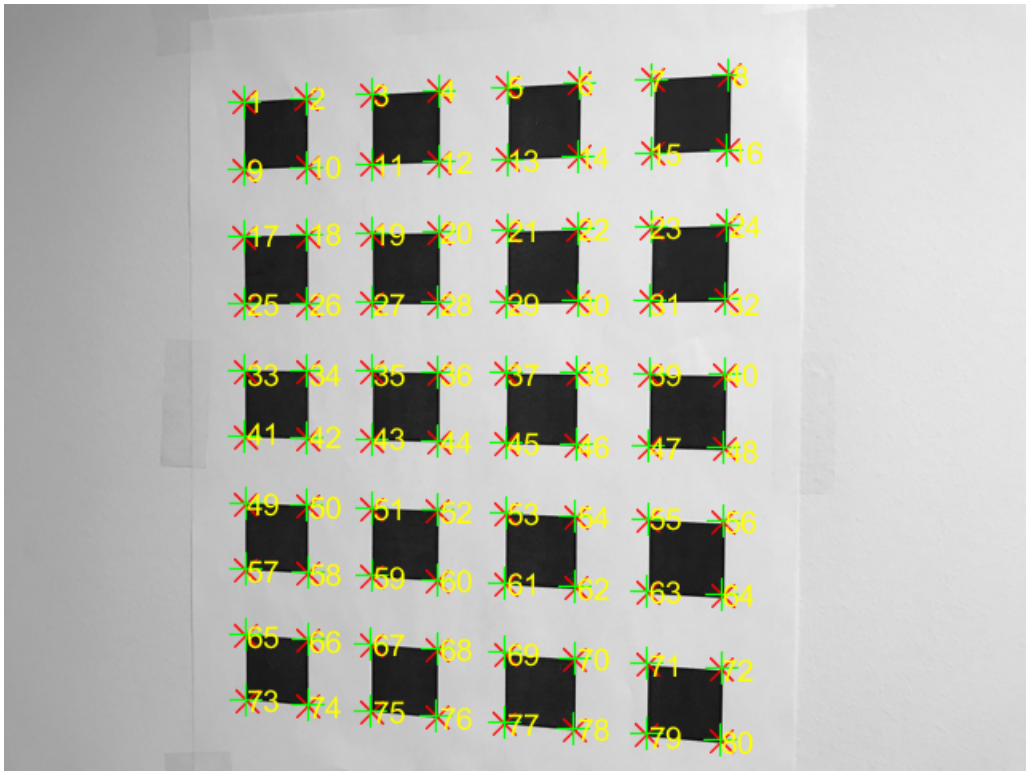


(a) Edges of 'IMG_1773.jpg'



(b) Corners of 'IMG_1773.jpg'

(c) LM vs LSE Re-projection of 'IMG_1773.jpg'



(d) With vs Without radial Correction Re-projection of 'IMG_1773.jpg'

Figure 9: Results of 'IMG_1773.jpg'

2.2.5 Re-projection to 'IMG_1758.jpg'

We select 'IMG_1758.jpg' as the Fixed Image and re-project by the parameters from LM method.

The extrinsic parameters of 'IMG_1758.jpg' are

$$R = \begin{bmatrix} 0.9999 & 0.0070 & -0.0133 \\ -0.0062 & 0.9981 & 0.0612 \\ 0.0137 & -0.0611 & 0.9980 \end{bmatrix}$$

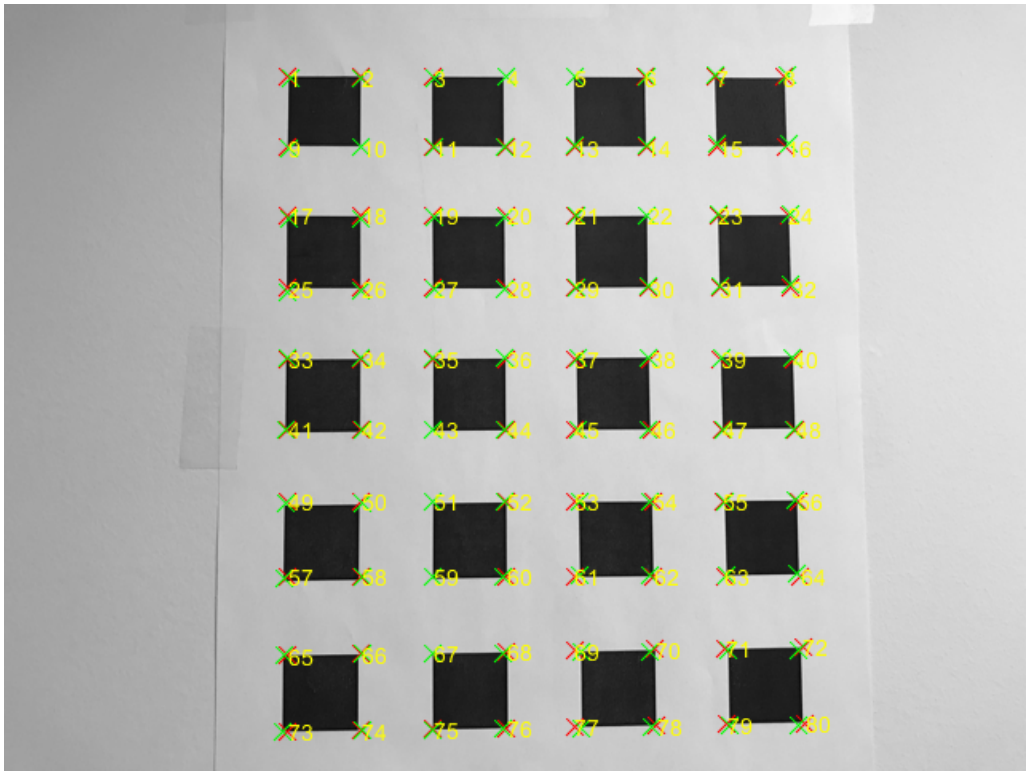
$$t = \begin{bmatrix} -79.9636 \\ -106.8870 \\ 307.2681 \end{bmatrix}$$

Table.10 is the quantitative comparison of different re-projections.

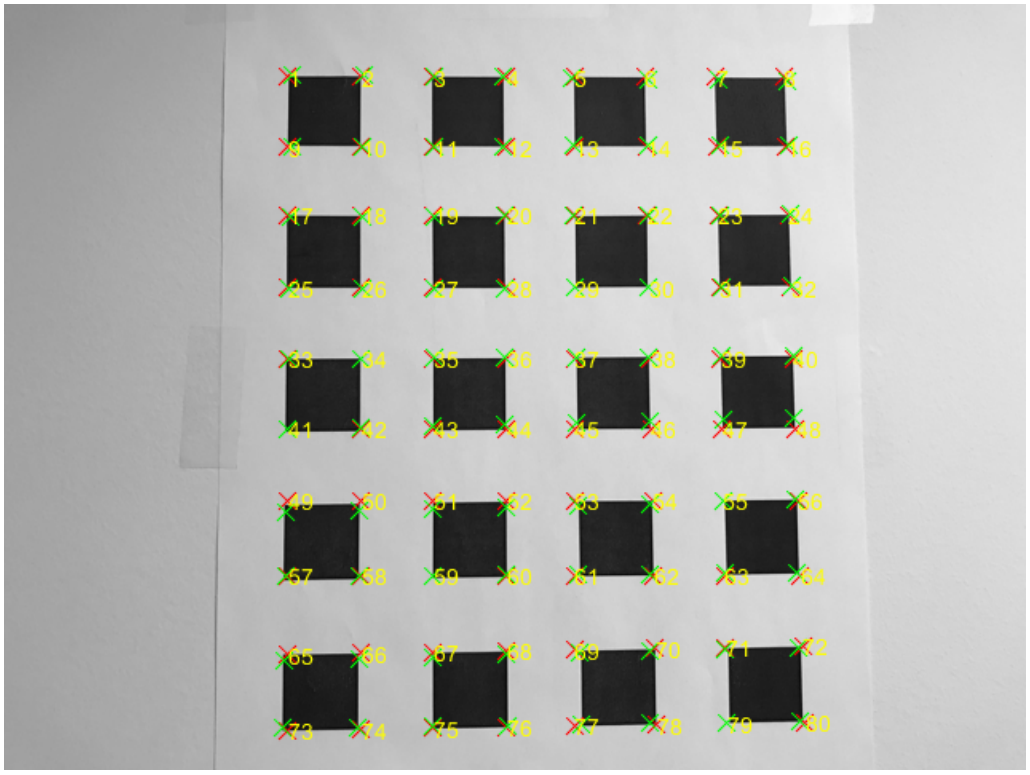
Target	mean	var
'IMG_1.jpg'	0.9644	0.1564
'IMG_3.jpg'	1.0067	0.5279
'IMG_12.jpg'	1.0904	0.4123
'IMG_15.jpg'	1.0054	0.2046

Table 10: Quantitative Comparison

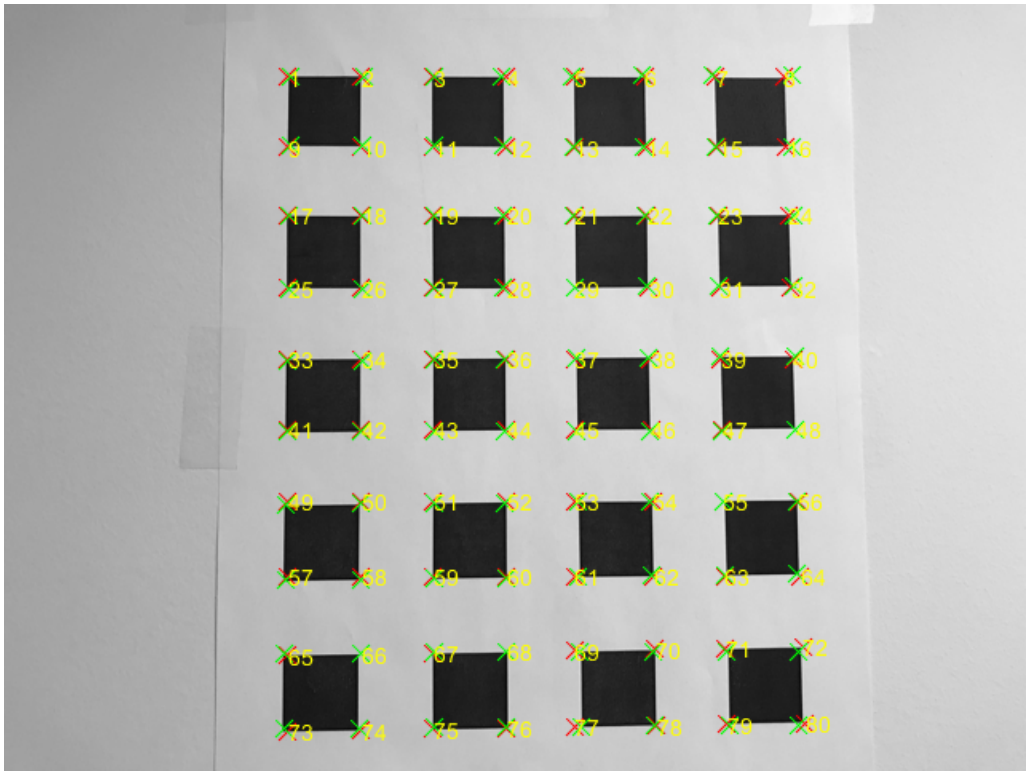
Fig.10 shows the results, where red '×' are the true corners and the green '×' are the re-projected corners.



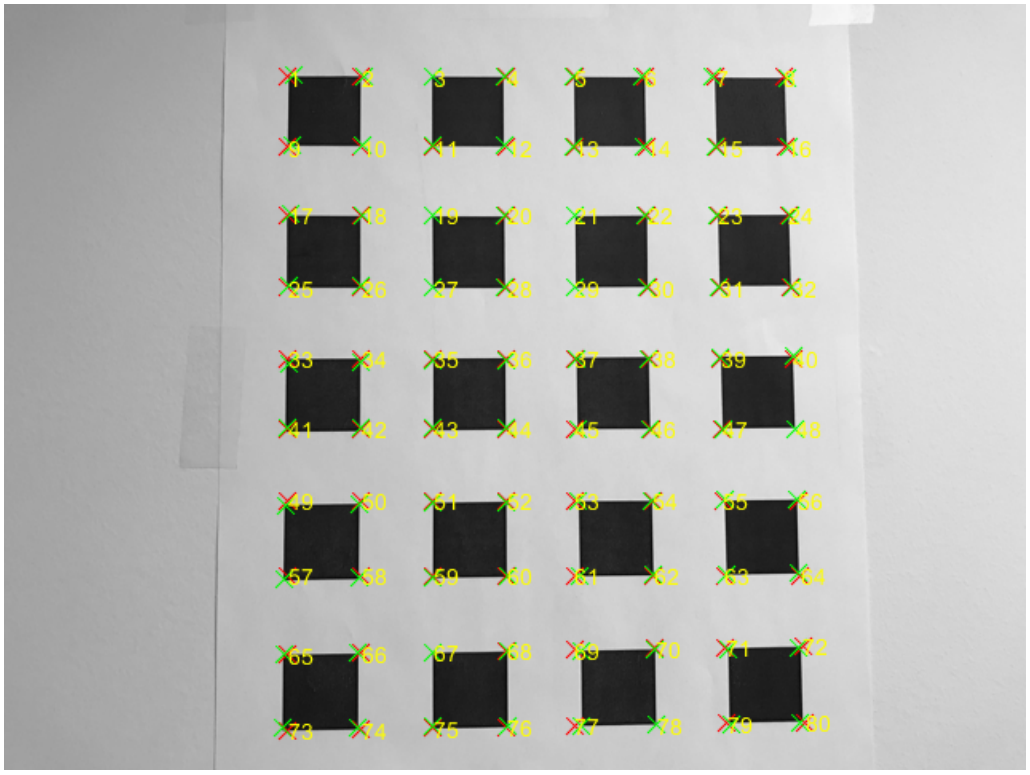
(a) Re-projection from 'IMG_1755.jpg' to 'IMG_1758.jpg'



(b) Re-projection from 'IMG_1759.jpg' to 'IMG_1758.jpg'



(c) Re-projection from 'IMG_1767.jpg' to 'IMG_1758.jpg'



(d) Re-projection from 'IMG_1773.jpg' to 'IMG_1758.jpg'

Figure 10: Re-projections to 'IMG_1758.jpg'

3 Source Code

The code is written in Matlab 2017b and Windows 10.

3.1 Main

```
%% ECE 661 2018 Fall Homework 8
% Ye Shi
% shi349@purdue.edu

close all;clear; clc;
grayread = @(x) rgb2gray(imresize(imread(x),[480 640]));
%% Import training images
folder = 'Dataset2'; % select dataset
dataset = struct('data',[],'corner',[]);
files = dir(folder);
files([1,2]) = []; % take out . and ..
num = numel(files);
% num = 2
for i = 1:num
    dataset(i).name = files(i).name;
    dataset(i).data = grayread([files(i).folder,'\',files(i).name]);
end
clear i;
%% Find corners in all images
for i = 1:num
    dataset(i).corner =findCorner(dataset(i).data,[folder,'results','\',files(i).
        name(1:end-4)]);
end
clear i
%% World Plane image corners
world.corner=zeros(80,2);
l = 0;
% imshow(zeros(800,600))
for i=1:10
    for j=1:8
        l=l+1;
        world.corner(l,:) =[(j-1)*25 (i-1)*25];
        world.label(l) = l;
    end
end
clear l i j
%% Find homography
% num = 2
for i = 1:num
    if numel(dataset(i).corner) == 80
        dataset(i).H = findH([world.corner,reshape(extractfield(...
            dataset(i).corner,'corner'),2,80)']);
    end
end

% combine the homographies
HvecSet = extractfield(dataset,'H');
Hnum = length(HvecSet)/9;
HvecSet = reshape(HvecSet,9,Hnum)';
clear Hnum

%% Find the intrinsic camer matrix K
K = findK(HvecSet)

%% Find the external camera calibration matrix R|t
for i = 1:num
    if ~isempty(dataset(i).H)
        [dataset(i).R,dataset(i).t] = findRt(dataset(i).H,K);
    end
end

%% LM refinement

options.Algorithm = 'levenberg-marquardt';
options.FunctionTolerance =1e-8;
p0= zeros(1,7+num*6+1);
```

```
66  p0(1:5) = [K(1,1) K(1,2) K(1,3) K(2,2) K(2,3)];
67  for i = 1:num
68      if ~isempty(dataset(i).H)
69          p0((i-1)*6+8:i*6+7) = [rotationMatrixToVector(dataset(i).R) ,...
70              dataset(i).t'];
71      else
72          p0((i-1)*6+8:i*6+7) = zeros(1,6);
73      end
74  end
75
76  % Without radical distortion
77  P = lsqnonlin(@dgeom,p0,[],[],options,world,dataset);
78  % Conduct to K
79  LMK = [P(1) P(2) P(3); 0 P(4) P(5); 0 0 1]
80  for i = 1:num
81      if ~isempty(dataset(i).H)
82          dataset(i).LMR= rotationVectorToMatrix(P((i-1)*6+8:(i-1)*6+10));
83          dataset(i).LMt = P(i*6+7-2:i*6+7)';
84      end
85  end
86
87
88  % With radical distortion
89  p(6) =  -6.8609e-09;
90  p(7) =   4.2824e-14;
91  p0(end) =1;
92  P = lsqnonlin(@dgeom,p0,[],[],options,world,dataset);
93  % Conduct to K
94  LMKr = [P(1) P(2) P(3); 0 P(4) P(5); 0 0 1]
95  LMk1 = P(6)
96  LMk2 = P(7)
97  for i = 1:num
98      if ~isempty(dataset(i).H)
99          dataset(i).LMRr= rotationVectorToMatrix(P((i-1)*6+8:(i-1)*6+10));
100         dataset(i).LMtr = P(i*6+7-2:i*6+7)';
101     end
102 end
103
104
105 %% Comparsion for projection
106 HC2R = @(P) (P(1:2,:)./P(end,:))';
107
108 for i = 1:num
109     if ~isempty(dataset(i).H)
110         % LSE
111         dataset(i).est = HC2R(K*[dataset(i).R dataset(i).t]* ...
112             [world.corner zeros(80,1) ones(80,1)]');
113         err = vecnorm(dataset(i).est - reshape(extractfield(...
114             dataset(i).corner,'corner'),2,80)',2,2);
115         dataset(i).estmean = mean(err);
116         dataset(i).estvar = var(err);
117
118         % After refine
119
120         dataset(i).LMest = HC2R(LMK*[dataset(i).LMR dataset(i).LMt]* ...
121             [world.corner zeros(80,1) ones(80,1)]');
122         err = vecnorm(dataset(i).LMest - reshape(extractfield(...
123             dataset(i).corner,'corner'),2,80)',2,2);
124         dataset(i).LMestmean = mean(err);
125         dataset(i).LMestvar = var(err);
126
127         % Radial Distortion
128         est_corners = LMKr * [dataset(i).LMRr dataset(i).LMtr] *...
129             [world.corner zeros(80,1) ones(80,1)]';
130         est_corners = est_corners./est_corners(end,:);
131         r = sum([est_corners(1,:);est_corners(2,:)].^2,1);
132         xrad = est_corners(1,:) + (est_corners(1,:)-240).*(LMk1.*r+LMk2.*r.^2);
133         yrad = est_corners(2,:) + (est_corners(2,:)-320).*(LMk1.*r+LMk2.*r.^2);
134         dataset(i).LMrest =[xrad;yrad]';
135         err = vecnorm(dataset(i).LMrest - reshape(extractfield(...
136             dataset(i).corner,'corner'),2,80)',2,2);
137         dataset(i).LMrestmean = mean(err);
138         dataset(i).LMrestvar = var(err);
```

```
139
140         end
141     end
142
143     %% Draw comparsion images
144     for i = 1:num
145         if ~isempty(dataset(i).H)
146             f = figure;
147             imshow(dataset(i).data)
148             hold on
149             scatter(dataset(i).est(:,1),dataset(i).est(:,2),60,'+','MarkerEdgeColor','b
                    ','MarkerFaceColor','b');
150             hold on
151             scatter(dataset(i).LMest(:,1),dataset(i).LMest(:,2),60,'x','MarkerEdgeColor
                    ','r','MarkerFaceColor','r');
152             hold on
153             for j = 1:80
154                 text(dataset(i).corner(j).corner(1),dataset(i).corner(j).corner(2),
                        int2str(dataset(i).corner(j).label),'Color','yellow','FontSize',7);
155                 hold on
156             end
157             hold off;
158             img = getframe();
159             imwrite(img.cdata,[folder,'results','\',files(i).name(1:end-4),'_LM.png'])
160             close(f)
161
162             f = figure;
163             imshow(dataset(i).data)
164             hold on
165             scatter(dataset(i).LMest(:,1),dataset(i).LMest(:,2),60,'x','MarkerEdgeColor
                    ','r','MarkerFaceColor','r');
166             hold on
167             scatter(dataset(i).LMrest(:,1),dataset(i).LMrest(:,2),60,'+','
                    MarkerEdgeColor','g','MarkerFaceColor','g');
168             hold on
169             for j = 1:80
170                 text(dataset(i).corner(j).corner(1),dataset(i).corner(j).corner(2),
                        int2str(dataset(i).corner(j).label),'Color','yellow','FontSize',7);
171                 hold on
172             end
173             hold off;
174             img = getframe();
175             imwrite(img.cdata,[folder,'results','\',files(i).name(1:end-4),'_LMr.png'])
176             %         w = waitforbuttonpress;
177             close(f)
178         end
179     end
180
181
182     %% Re-project to Fixed Image
183     if folder(end) == '1'
184         f = 3;
185         t = [1 23 4 7];
186     elseif folder(end) == '2'
187             f = 5;
188         t = [2 6 14 19];
189     end
190
191     % profermence matrix
192     M = zeros(4,2);
193     k= 0;
194     for i = t
195         k = k+1;
196        [ M(k,1),M(k,2) ]= reproj(dataset(f), dataset(i), LMK)
197     end
198
199
200
201     %%
202     save([folder,'results','\',datestr(datetime(),30)]);
```

3.2 Hough Corner Detector

```
%% ECE 661 2018 Fall Homework 8
% Ye Shi
% shi349@purdue.edu

function C = findCorner(I,filename)
% To find the cornor in a grayscale image I
if nargin <2
    filename = datestr(datetime(),30)
end
[h,w] = size(I);

E = edge(I,'canny',0.75); % canny edge detector with treshold 0.7

f = figure;
% hough transfom
[H,T,R] = hough(E,'RhoResolution',0.5);
MaxlineNum = 18;
lineNum = 0;
hpfactor = 0.31;
while lineNum <18
    hpfactor = hpfactor-0.01;
    P = houghpeaks(H,MaxlineNum,'Threshold',hpfactor*max(H(:)));
    lines = houghlines(E,T,R,P,'FillGap',300,'MinLength',70);
    lineNum = length(lines);
end


% separete lines by 2 orthogonal directions

LinesHC = zeros(lineNum,3);
for i = 1:lineNum
    l = LineHC(lines(i));
    LinesHC(i,:) =l;
end
LinesHC1 = sortrows(LinesHC);
LinesHC1 = LinesHC1(1:8,:);
LinesHC2 = sortrows(LinesHC,2);
LinesHC2 = LinesHC2(1:10,:);

% Plot the lines
for i = 1:8
    p1 = cross(LinesHC1(i,:),[0 1 0]);
    p2 = cross(LinesHC1(i,:),[0 1 -h]);
    xy = [p1(1:2)./p1(end) ; p2(1:2)./p2(end)];
    plot(xy(:,1),xy(:,2),'LineWidth',.5,'Color','green');
    hold on;
end
for i = 1:10
    p1 = cross(LinesHC2(i,:),[1 0 -w]);
    p2 = cross(LinesHC2(i,:),[1 0 0]);
    xy = [p1(1:2)./p1(end) ; p2(1:2)./p2(end)];
    plot(xy(:,1),xy(:,2),'LineWidth',.5,'Color','blue');
    hold on;
end

% find insection points
pnum = 0;
nl1 = length(LinesHC1);
nl2 = length(LinesHC2);
for i = 1:nl2
    for j = 1:nl1
        pnum = pnum+1;
        point = cross(LinesHC1(j,:),LinesHC2(i,:));
        point = point./point(end);
        point = point(1:2);
        % trim outrange points
        if sum(point>[1 1])==2 && sum(point<=[w h])==2
            points(pnum,:) = point;
        else
            pnum = pnum-1;
        end
    end
end
```

```
74  points = points(1:pnum,:);
75
76  % output and plot corners and label
77  cnum = min(80, length(points)); % fixed
78  C = struct('corner',[],'label',[]);
79
80  for i = 1:cnum
81      C(i).corner = points(i,:);
82      C(i).label = i;
83      plot(points(i,1),points(i,2),'x','LineWidth',1,'Color','red');
84      hold on
85      text(points(i,1),points(i,2),int2str(C(i).label),'Color','yellow','FontSize',7)
            ;
86      hold on
87  end
88  if cnum ==80
89      img = getframe();
90      imwrite(img.cdata,[filename,'_corners.png']);
91      close(f)
92      f = figure;
93      imshow(E);
94      hold off;
95      img = getframe();
96      imwrite(img.cdata,[filename,'_edge.png']);
97      close(f)
98  else
99      close(f)
100 end
101
102 end
103
104 function l = LineHC(line)
105 % transfer lines to HC
106 p1a = [line.point1 1];
107 p1b = [line.point2 1];
108 l = cross(p1a,p1b);
109 if l(end) == 0
110     l = [];
111 else
112     l = l./l(end);
113 end
114 end
```

3.3 LSE Homography Estimator

```
1  %% ECE 661 2018 Fall Homework 5
2  % Ye Shi
3  % shi349@purdue.edu
4
5  function H = findH(pairs)
6  % This function can find homography by SVD
7  % pairs are m (m>5) pairs of corresponding points by (x1,y1,x2,y2)
8  m = size(pairs,1);
9  if m < 5
10     disp("No necessary number of pairs (m<5)");
11     return
12 end
13 disp([num2str(m),'␣pairs␣points␣are␣used␣to␣find␣the␣homography.']);
14 A = zeros(m*2,9);
15 % Construct A
16 for i = 0:m-1
17     A(i*2+1:i*2+2,:) = [0 0 0 -pairs(i+1,1) -pairs(i+1,2) -1 pairs(i+1,4)*pairs(i
           +1,1) pairs(i+1,4)*pairs(i+1,2) pairs(i+1,4);
18         pairs(i+1,1) pairs(i+1,2) 1 0 0 0 -pairs(i+1,1)*pairs(i+1,3) -pairs(i+1,3)*
               pairs(i+1,2) -pairs(i+1,3)];
19 end
20 if rank(A)< 9
21     disp("Warning: Rank(A) is less than 9!");
22 end
23 [~,~,V] = svd(A); % V is order by the descending order of S
24 H = V(:,end); % H is the vector with smallest S
25 H = [H(1:3)';H(4:6)';H(7:9)'];
26 end
```

3.4 Zhang's Algorithm for Intrinsic Camera Matrix

```
%% ECE 661 2018 Fall Homework 8
% Ye Shi
% shi349@purdue.edu

function K = findK(HvecSet)
% Find the intrinsic camer matrix K
% HvecSet is homography vector [h11, h21,h31 , ... , h33;
%h11, h21,h31 , ... , h33; ...]
[m,n] = size(HvecSet);
if n ~=9
    error('Wrong matrix demension!');
elseif m <= 2
    error('Not enough homographies!');
end

% construct V matrix
Hcon = @(H) reshape(H,3,3);
Vrow = @(H,i,j) [H(1,i)*H(1,j) ,...
    H(1,i)*H(2,j)+H(2,i)*H(1,j) ,...
    H(2,i)*H(2,j) ,...
    H(3,i)*H(1,j)+H(1,i)*H(3,j)  ,...
    H(3,i)*H(2,j)+H(2,i)*H(3,j) ,...
    H(3,i)*H(3,j)];

V = zeros(2*m,6);
for k = 0:m-1
    H = Hcon(HvecSet(k+1,:));
    V(2*k+1:2*k+2,:) = [Vrow(H,1,2);
        Vrow(H,1,1) - Vrow(H,2,2)];
end

% SVD V to find the parameters in K
[~,~,T] = svd(V);
b = T(:,6); %B11 B12 B22 B13 B23 B33

y0 = (b(2)*b(4)-b(1)*b(5))/(b(1)*b(3)-b(2)^2);
l = b(6)-(b(4)^2+y0*(b(2)*b(4)-b(1)*b(5)))/b(1);
ax = sqrt(l/b(1));
ay = sqrt(l*b(1)/(b(1)*b(3)-b(2)^2));
s = -b(2)*ax^2*ay/l;
x0 = s*y0/ay-b(4)*ax^2/l;
K = [ax s x0; 0 ay y0; 0 0 1];
end
```

3.5 Extrinsic Camera Parameters

```
%% ECE 661 2018 Fall Homework 8
% Ye Shi
% shi349@purdue.edu

function [R,t] = findRt(H,K)
% Find the external camera calibration matrix
% R rotation
% t tranfer
% H homography
% K intrinsic camera matrix
K_inv = K^-1;
t = K_inv*H(:,3);
mag = norm(K_inv*H(:,1));
if(t(3)<0)
    mag = -mag;
end
r1 = K_inv*H(:,1)/mag;
r2 = K_inv*H(:,2)/mag;
r3 = cross(r1,r2);
R = [r1 r2 r3];
t = t/mag;
[U,~,V] = svd(R);
R = U*V';

% Convert to the R,t using Rodriguez formula
```

```
R =rotationVectorToMatrix(rotationMatrixToVector(R));
end
```

3.6 Distance Error Cost Function

```
%% ECE 661 2018 Fall Homework 8
% Ye Shi
% shi349@purdue.edu

function err = dgeom(p,world,dataset)
% Cost function for the LM

num = numel(dataset);
ax = p(1);
s = p(2);
x0 = p(3);
ay = p(4);
y0 = p(5);
% Intrinsic calibration matrix
K = [ax s x0;
     0 ay y0;
     0 0 1];
% Radial distortion
if p(end) ==1
    % Parameters for radial distortion
    k1 = p(6);
    k2 = p(7);
end

err = [];
for k = 1:num
    if ~isempty(dataset(k).H)

        R= rotationVectorToMatrix(p((k-1)*6+8:(k-1)*6+10));
        t = p(k*6+7-2:k*6+7)';
        est_corners = K*[R t] * [world.corner zeros(80,1) ones(80,1)]';
        est_corners = est_corners./est_corners(3,:);
        % Radial distortion
        if p(end) ==1
            r = sum([est_corners(1,:)-240;est_corners(2,:)-320].^2,1);
            xrad = est_corners(1,:) + (est_corners(1,:)-240).*(k1.*r+k2.*r.^2);
            yrad = est_corners(2,:) + (est_corners(2,:)-320).*(k1.*r+k2.*r.^2);
            est_corners = [xrad;yrad];
        end
        real_corners = reshape(extractfield(dataset(k).corner,'corner'),2,80);
        err = [err est_corners(1:2,:)-real_corners];
    end
end

end
```

3.7 Reprojection

```
%% ECE 661 2018 Fall Homework 8
% Ye Shi
% shi349@purdue.edu

function [mu,nu] = reproj(fixed, target, LMK)
% reproject target image to fixed image by LM calibration
Hf = LMK*[fixed.LMR(:,1:2) fixed.LMt];
Ht = LMK*[target.LMR(:,1:2) target.LMt];
H = Hf*pinv(Ht);
est_corners  = H*[reshape(extractfield(target.corner,'corner'),2,80);ones(1,80)];
est_corners = [est_corners(1:2,:)./est_corners(end,:)]';
fixed_corners = reshape(extractfield(fixed.corner,'corner'),2,80)';
err = vecnorm(fixed_corners-est_corners,2,2);
f = figure;
imshow(fixed.data)
hold on
scatter(fixed_corners(:,1),fixed_corners(:,2),60,'x','MarkerEdgeColor','r','
    MarkerFaceColor','r');
hold on
```

```
19  scatter(est_corners(:,1),est_corners(:,2),60,'x','MarkerEdgeColor','g','
        MarkerFaceColor','g');
20  hold on
21  for j = 1:80
22      text(fixed.corner(j).corner(1),fixed.corner(j).corner(2),int2str(fixed.corner(j
            ).label),'Color','yellow','FontSize',7);
23      hold on
24  end
25  hold off;
26  img = getframe();
27  imwrite(img.cdata,[target.name(1:end-4),'To',fixed.name(1:end-4),'.png'])
28  close(f)
29  mu = mean(err);
30  nu =var(err);
31  end
```