Computer Vision

Calibration

Name: hanxue Liang

Student ID: 16-912-156

Catalog

1. Calibration problem

In this exercise, we will calibrate our own camera. We need to calculate the intrinsic matrix and estimate the distortion coefficients of the camera. I will realize three methods below: the simple Direct Linear Transform algorithm and the Gold Standard algorithm as well as the Bouget's Calibration Toolbox.

The image of the calibration object are as follows:

Figure1. Image of the object (the size of the square is 27mm*27mm)

I use the 'getpoint.m' to select 6 points from this image and get their coordinates according the coordinate system I assign (shown in Figure2). The 2D coordinates (image coordinates) and the 3D coordinates are as follows:

Figure2: image with the selected points

2D coordinates:

x	1762.93	1122.88	1601.94	3164.76	2438.33	2658.22
y	1950.11	2071.84	948.79	1777.33	1235.45	2248.54

3D coordinates:

x	54	135	81	0	0	0
y	0	0	0	135	54	81
z	54	54	162	81	135	27

2. Direct Linear Transform

2.1 Data Normalization

First, for all points, scale the homogeneous vectors such that the last entry becomes 1, i.e., x = (x, y, 1) and X = (x, y, z, 1). Then, transform the input points so that the centroid of the points is at the origin.

$$\begin{cases} \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \\ \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \end{cases}, \begin{cases} \widehat{x_i} = x_i - \bar{x} \\ \widehat{y_i} = y_i - \bar{y} \end{cases}$$

$$\begin{cases} \bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \\ \bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i \\ \bar{Z} = \frac{1}{n}\sum_{i=1}^{n} Z_i \end{cases}, \begin{cases} \widehat{X_i} = X_i - \bar{X} \\ \widehat{Y_i} = Y_i - \bar{Y} \\ \widehat{Z_i} = Z_i - \bar{Z} \end{cases}$$

Then we need to scale the points so that average Euclidean distance from the origin is $\sqrt{2}$ for the image points, and $\sqrt{3}$ for the object points. I choose different scale factors for each coordinate direction separately.

$$\frac{1}{n}\mathbf{m_1}^2\sum_{i=1}^{n}(\widehat{x_i}^2) = \mathbf{1} \quad \frac{1}{n}\mathbf{m_2}^2\sum_{i=1}^{n}(\widehat{y_i}^2) = \mathbf{1}$$

$$\frac{1}{n}\mathbf{m_3}^2\sum_{i=1}^{n}\left(\widehat{X_i}^2\right) = \mathbf{1} \quad \frac{1}{n}\mathbf{m_4}^2\sum_{i=1}^{n}\left(\widehat{Y_i}^2\right) = \mathbf{1} \quad \frac{1}{n}\mathbf{m_5}^2\sum_{i=1}^{n}\left(\widehat{Z_i}^2\right) = \mathbf{1}$$

So we have:

$$T = \begin{bmatrix} \mathbf{m_1} & 0 & -\mathbf{m_1}\bar{x} \\ 0 & \mathbf{m_2} & -\mathbf{m_1}\bar{y} \\ 0 & 0 & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} \mathbf{m_3} & 0 & 0 & -\mathbf{m_3}\bar{X} \\ 0 & \mathbf{m_4} & 0 & -\mathbf{m_4}\bar{Y} \\ 0 & 0 & \mathbf{m_5} & -\mathbf{m_5}\bar{Z} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The normalized points are:

$$\begin{pmatrix} \widetilde{x_i} \\ \widetilde{y_i} \\ 1 \end{pmatrix} = T * \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

And

$$\begin{pmatrix} \widetilde{X_i} \\ \widetilde{Y_i} \\ \widetilde{Z_i} \\ 1 \end{pmatrix} = U * \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix}$$

The calculation is implemented in 'normalization.m' and I find the transformation matrix as follows:

$$\mathrm{T} = \begin{bmatrix} 0.0014 & 0 & -3.0709 \\ 0 & 0.0022 & -3.6812 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{U} = \begin{bmatrix} 0.0196 & 0 & 0 & -0.8839 \\ 0 & 0.0196 & 0 & -0.8839 \\ 0 & 0 & 0.0209 & -1.7874 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}$$

2.2 Direct Linear Transform

(1) Normalized camera matrix $\hat{P}$

With the normalized 2D and 3D points, I construct the projection equation with 6 points:

$$[M]_{18*18} * \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \lambda \end{bmatrix} = 0$$

I compose SVD decompose to M: $\mathrm{M} = \mathrm{U} * \mathrm{S} * V^T$,and $\hat{P}$ can be found from the last column of V^T, I calculate the $\hat{P}$ to be:

```
P_normalized =

   -0.1470    0.1905    0.0018    0.0056
    0.0241    0.0287   -0.3439   -0.0125
   -0.0444   -0.0372    0.0047    0.3410
```

(2) After get $\hat{P}$,I compute the denormalized camera matrix $\mathrm{P} = T^{-1} * \hat{P} * U$.

```
P =

   -3.8491    1.0350    0.2329  835.1086
   -1.2667   -0.9859   -3.1639  947.5180
   -0.0009   -0.0007    0.0001    0.4047
```

(3) Apply QR decomposition to the left 3*3 part of P, get K and R as following:

```
K =

   1.0e+03 *

    2.8622    0.0190    2.0092
         0    2.8814    1.1634
         0         0    0.0010

R =

   -0.0007    0.0009    0.0000
   -0.0001   -0.0000   -0.0011
   -0.0009   -0.0007    0.0001
```

The camera center is the 1-dimensional right null-space C of P. And because M is not singular (finite camera), So $C = \begin{pmatrix} -M^{-1}P_4 \\ 1 \end{pmatrix}$:

$$C = \begin{pmatrix} 285.3477 \\ 228.6723 \\ 113.9855 \\ 1 \end{pmatrix}$$

It can be noticed that the mean projection error for Direct Linear Transform is 1.7573 pixel*pixel, which is quite small considering that the image is 4000 * 3000 (pixel*pixel).

(4) Visualize the reprojected points: I visualized the selected reprojected points on the calibration object with the computed camera matrix:

Figure3. Image of the reprojected points (marked by blue circle)

(5) Run 'runDLT_denorm' to get results for DLT algorithm without normalization.

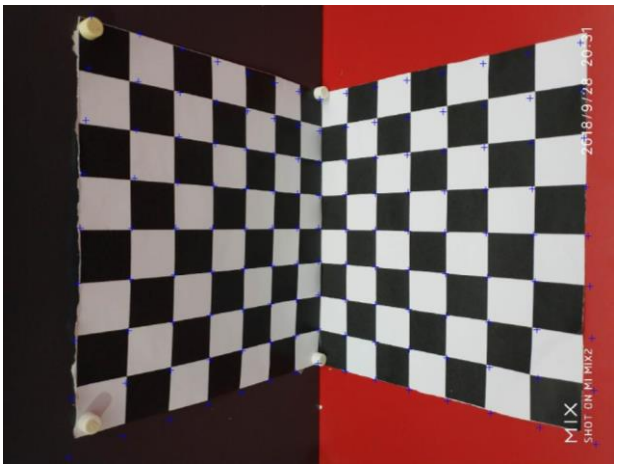

Figure4. Image of the reprojected points without normalization (marked by blue plus)

The mean projection error here is 2.3660, a bit larger than the Direct Linear Transform with normalization (1.7573). It improves the error by 25.7%.

I reselect another six points to do the same calculation as above and find that the mean projection error for Direct Linear Transform with normalization is 1.9653. By comparison, the error for DLT without normalization is 3.2301, improving the error by 39.15%. When calculate DLT without normalization, the select of the hand-clicked points can have a huge influence on the final result. And it's noticed that the introduction of normalization in DLT method could make the calculation of DLT more stable and help to find more accurate intrinsic parameters.

3. Gold Standard algorithm

3.1 Initialization with DLT

Similar to 2.2, I apply DLT algorithm and get the initial camera matrix $\hat{P}$:

```
P_normalized =

   -0.1470    0.1905    0.0018    0.0056
    0.0241    0.0287   -0.3439   -0.0125
   -0.0444   -0.0372    0.0047    0.3410
```

Then I denormalize it and get initial $\widetilde{P}$ for further optimization:

```
P =

   -3.8491    1.0350    0.2329  835.1086
   -1.2667   -0.9859   -3.1639  947.5180
   -0.0009   -0.0007    0.0001    0.4047
```

3.2 Visualization

Here I visualize the hand-clicked points on the image by using the initial camera matrix $\widetilde{P}$, the image is as follows:

Figure5. Image of the reprojected hand-clicked points without initial camera matrix (marked by red plus)

3.3 Optimization

Now I optimize $\hat{P}$ by using fminsearch and I get the $\hat{P}_{opt}$ for the normalized xy, XYZ as:

```
Pn_opt =

   -0.0025    0.0032    0.0000    0.0001
    0.0004    0.0005   -0.0058   -0.0002
   -0.0007   -0.0006    0.0001    0.0057
```

3.4 Decomposition

After denomalization and applying QR decompose, I get the camera matrix and intrinsic parameters of the camera as follows:

K =

1.0e+03 *

2.8627	0.0252	2.0158
0	2.8831	1.1550
0	0	0.0010

R =

1.0e-04 *

-0.1230	0.1479	0.0036
-0.0154	-0.0081	-0.1916
-0.1471	-0.1228	0.0170

3.5 Reprojection Error

After applying Gold Standard algorithm, I get the average reprojected error for the clicked points as 1.3776, which is a little smaller than that in DLT method (1.7573). It means that it reduces the average error by 27.6%. Then I reselect another six points and apply DLT and the Gold Standard algorithm. For this group, the average projection errors I obtain are separately 1.965(DLT) and 1.544(Gold standard algorithm), means reduction by 27.3%. Consequently, Gold Standard algorithm can be a good improvement from Direct Linear Transform method.

The reprojected points of all checkerboard corners on the calibration object with the computed camera matrix are visualized as following:

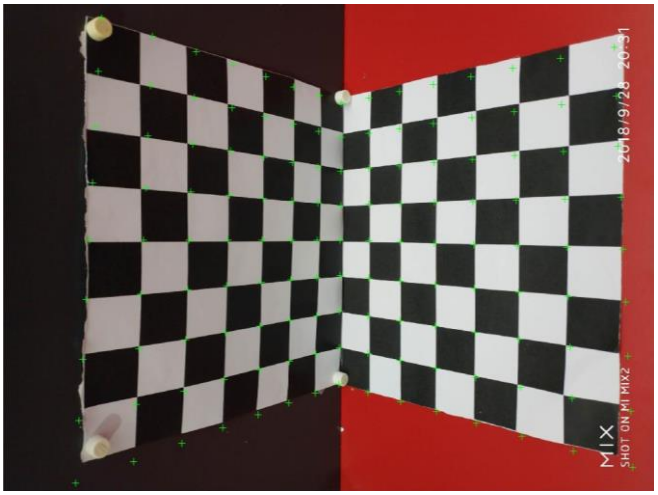

Figure6. Image of the reprojected all corner points without camera matrix obtained by gold standard algorithm (marked by green plus)

4. Bouget's Calibration Toolbox

After running Bouget's Calibration Toolbox based on 20 images

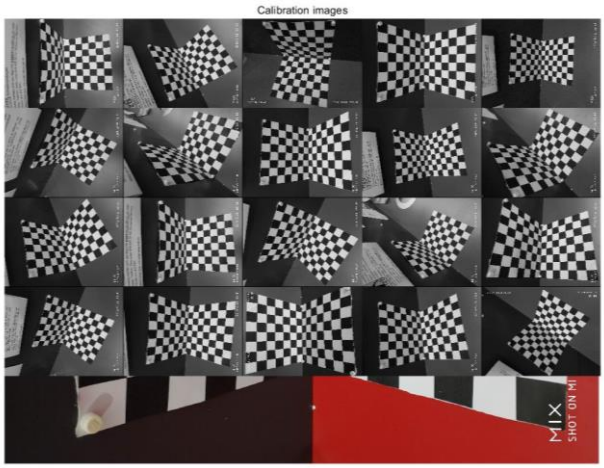

Figure7. Image of all the calibration images by Bouget's Calibration Toolbox

I get the intrinsic parameters of the camera as follows:

Calibration parameters after initialization:

Focal Length: fc = [3007.57753 3007.57753]

Principal point: cc = [1999.50000 1499.50000]

Skew: alpha_c = [0.00000] => angle of pixel = 90.00000 degrees

Distortion: kc = [0.00000 0.00000 0.00000 0.00000 0.00000]

Calibration results after optimization (with uncertainties):

Focal Length: fc = [3000.91361 3019.46295] +/- [32.90292 33.19826]

Principal point: cc = [2057.50402 1354.59318] +/- [38.76951 40.11662]

Skew: alpha_c = [0.00000] +/- [0.00000] => angle of pixel axes = 90.00000 +/- 0.00000 degrees

Distortion: kc = [0.06185 -0.26106 -0.00569 0.00614 0.00000] +/- [0.02994 0.07834 0.00486 0.00334 0.00000]

Pixel error: err = [0.84735 1.06397]

Therefore, the pixel error for each point is [0.84735 1.06397], which means $\sqrt{0.84735^2 + 1.06397^2}$=1.359 pixels error. This error is smaller than those of DLT method and Gold standard algorithm. The bigger errors of the latter two methods can be explained by the deficiency of selected points. Because Bouget's Calibration Toolbox does the calibration based on all 20 images while I calculate the intrinsic parameters based on two groups of points.

5. Bonus

5.1 Gold Standard algorithm with radial distortion estimation

(1) Firstly, I run Direct Linear Transform method to get the initial camera matrix $\hat{P}$:

```
P_normalized =

   -0.1470    0.1905    0.0018    0.0056
    0.0241    0.0287   -0.3439   -0.0125
   -0.0444   -0.0372    0.0047    0.3410
```

(2) In the second step, I introduce the radial distortion to the reprojection from the 3D points to the hand-clicked image points and try to minimize the cost function: $\sum_i^N \mathrm{d}\left(\hat{x_i}, \tilde{P}\hat{X_i}\right)^2$. I optimize $\tilde{P}$ by using fminsearch and I get the $\widetilde{P_{opt}}$ for the normalized xy, XYZ as:

```
Pn_opt =

   -0.1468    0.1906    0.0015    0.0058
    0.0239    0.0287   -0.3442   -0.0127
   -0.0444   -0.0370    0.0048    0.3410
```

And I also get the radial distortion $\widetilde{K}$ as:

```
rad =

   1.0e-03 *

    0.4435    0.4387
```

(3) In the third step, I add the radial distortion and reproject the hand-clicked points. I calculate the average reprojection error $\mathrm{d}\left(x_i, inv(T) * \widetilde{P_{opt}}\hat{X_i}\right)^2$. By running 'runGoldStandardRadial.m', the average error I obtain is 1.3238.

5.2 Image undistortion

With the radial distortion parameter obtained from 5.1, I do the image undistortion based on the hand-clicked points by running 'imageundistortion.m'.

6. Conclusion

By doing this exercise, I realize image calibration by implementing the Direct Linear Transform algorithm, the Gold Standard algorithm as well as the Bouget's Calibration Toolbox. It is found that Gold Standard algorithm could produce more accurate intrinsic parameter than the Direct Linear Transform algorithm. By comparison, Bouget's Calibration Toolbox can produce the most accurate estimation of intrinsic parameters among these three methods. Finally, I also find the radial distortion parameter of the camera and by introducing this distortion, and do the image undistortion based on the hand-clicked points.