國立陽明交通大學

113-1 電腦視覺 HW-1

Group 9 313551130 鍾名捷/313551133 陳軒宇/313553036 葉軒宇

1. Introduction

In our daily lives, environments are inherently 3D, but when we capture these scenes with a camera, they become 2D images. If we want to reconstruct the original 3D scenes from these 2D photographs, camera calibration becomes a necessary step. The goal of calibration is to find the effective projection transform, thereby yielding significant information regarding the vision system, such as focal lengths, camera pose, camera center, and more. The purpose of this assignment is to complete the camera calibration process without using the OpenCV library, relying solely on mathematical concepts. This includes solving for the homography matrix, intrinsic matrix, and extrinsic matrix. The detailed process will be discussed later.

2. Implementation procedure

There are three part in our implementation

2.1 Calculate homography matrix

We know that p_i is our pixel coordinate, P_i is world coordinate and H is homography matrix

$$p_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}, P_i = \begin{bmatrix} U_i \\ V_i \\ 1 \end{bmatrix}, H = \begin{bmatrix} f/s_x & 0 & o_x \\ 0 & f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix}$$

So we can rewrite the equation as

$$P_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \sim HP_i = \begin{bmatrix} h_1 P_i \\ h_2 P_i \\ h_3 P_i \end{bmatrix}$$

Where h1, h2, h3 are rows of H, so we can get 2 equation from each of them

$$\begin{cases} u_{i(h_3 P_i)} - h_1 P_i = 0 \\ v_{i(h_3 P_i)} - h_2 P_i = 0 \end{cases}$$

Cause we have n coordinates (n is the number of corners in a chessboard photo), so we can form it to following equation.

$$\begin{bmatrix} -P_1^T & 0^T & u_1P_1^T \\ 0^T & -P_1^T & v_1P_1^T \\ \dots & & \\ -P_n^T & 0^T & u_nP_n^T \\ 0^T & -P_n^T & v_nP_n^T \end{bmatrix} \begin{bmatrix} h_1^T \\ h_2^T \\ h_3^T \end{bmatrix} = Ph = 0$$

And our goal is to minimize $\|Ph\|^2$ subject to $\|h\|^2 = 1$, it can be solved by singular value decomposition

$$P = UDV^{-T}$$

And we set h equal to the last column of V(the last row of V^T), and resize it from 9*1 into 3*3.

$$h = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{14} \\ h_{15} \\ h_{16} \\ h_{17} \\ h_{18} \\ h_{19} \end{bmatrix}, H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

2.2 Solve Intrinsic matrix

After we get the homography matrix, we can calculate the intrinsic matrix for each photo. First we know the relationship between H and K(intrinsic matrix)

$$H = [h_1 \quad h_2 \quad h_3] = \begin{bmatrix} f/s_x & 0 & o_x \\ 0 & f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} = K[r_1 \quad r_2 \quad t]$$

So we can derive into two equation from above

$$r_1 = K^{-1}h_1 \ and \ r_2 = K^{-1}h_2$$

Because $r_1, r_2 \ and \ r_3$ are orthogonal to each other, so from basic mathematic we know that

$$\begin{cases} r_1^T r_2 = 0 \\ \|r_1\| = \|r_2\| = 1 \end{cases}.$$

It can be rewrite as

$$\begin{cases} {h_1}^T K^{-T} K^{-1} h_2 = 0 \\ {h_1}^T K^{-T} K^{-1} h_1 = {h_2}^T K^{-T} K^{-1} h_2 \end{cases}$$

We let

$$B := K^{-T}K^{-1} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

Where B is symmetric and positive definite. So K can be computed form B using Cholesky factorization.

Next, we define

$$\mathrm{b} = (b_{11}, b_{12}, b_{13}, b_{22},\ b_{23}, b_{33})$$

it leads to the system

$V_i b = 0\ , for\ each\ H_i$. And V = $\begin{bmatrix} V_1 \\ \vdots \\ V_n \end{bmatrix}$, Where n is the number of plane

In this session, our goal is to minimize least-square error

$$b = arg\min_{b} Vb$$

To complete this, we can apply SVD to V

$$V = UDV^T$$

Then we can get B by setting b to the last column of V. Next apply Cholesky factorization to B.

$$B = LL^H = LL^T = K^{-T}K^{-1} \rightarrow K = L^{-T}$$

Finally we get the intrinsic matrix K.

2.3 Solve Extrinsic matrix

After we get the intrinsic matrix, we can calculate extrinsic matrix by

$$\begin{cases} r_1 = \lambda K^{-1}h_1 \\ r_2 = \lambda K^{-1}h_2 \\ r_3 = r_1 \times r_2 \\ t = \lambda K^{-1}h_3 \\ \lambda = 1/\|K^{-1}h_1\| \end{cases}$$

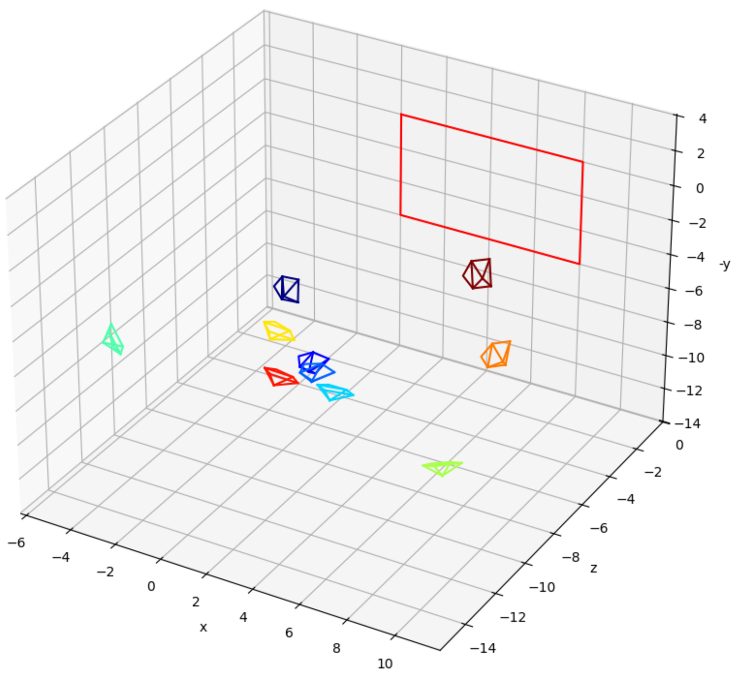
Than extrinsic matrix would be

$$Extrinsic\ Matrix = [R|t] = [r_1 \quad r_2 \quad r_3 \quad t]$$

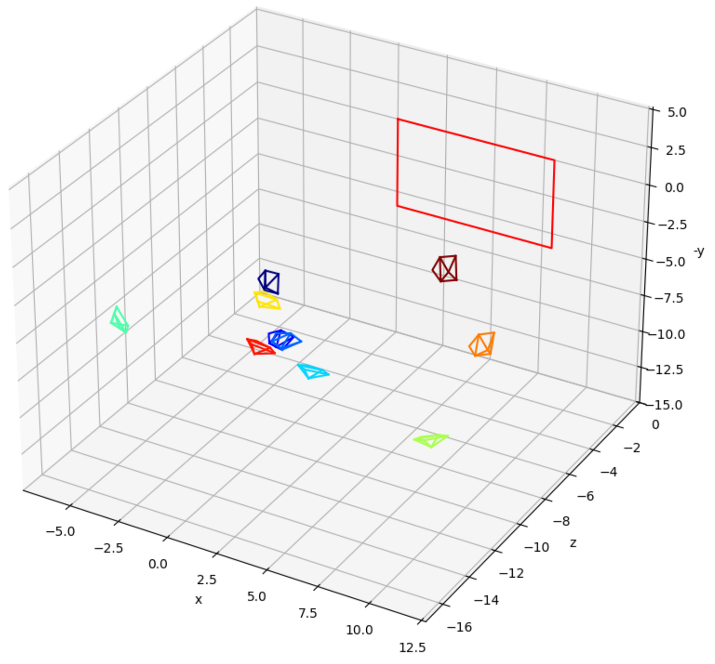
3. Eeperimental result

With image provided by the TA

Extrinsic Parameters Visualization, use_opencv=True

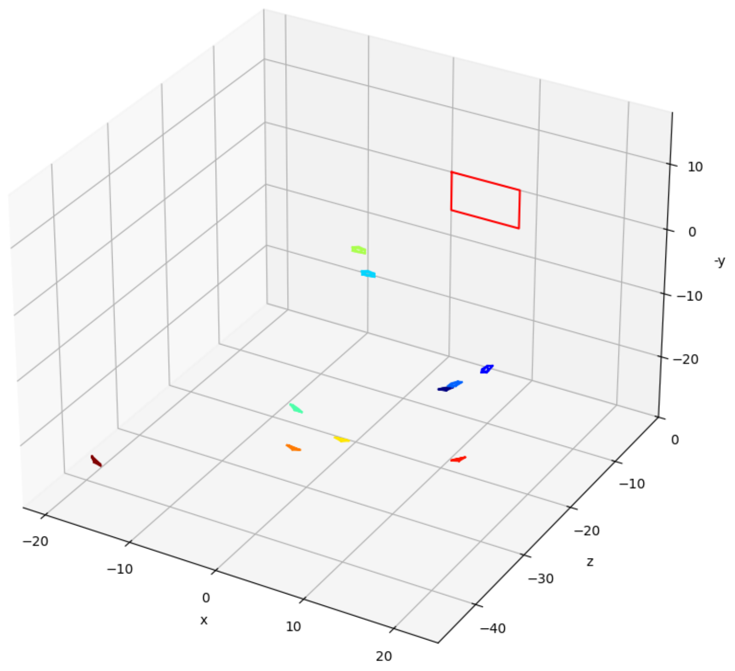


Extrinsic Parameters Visualization, use_opencv=False

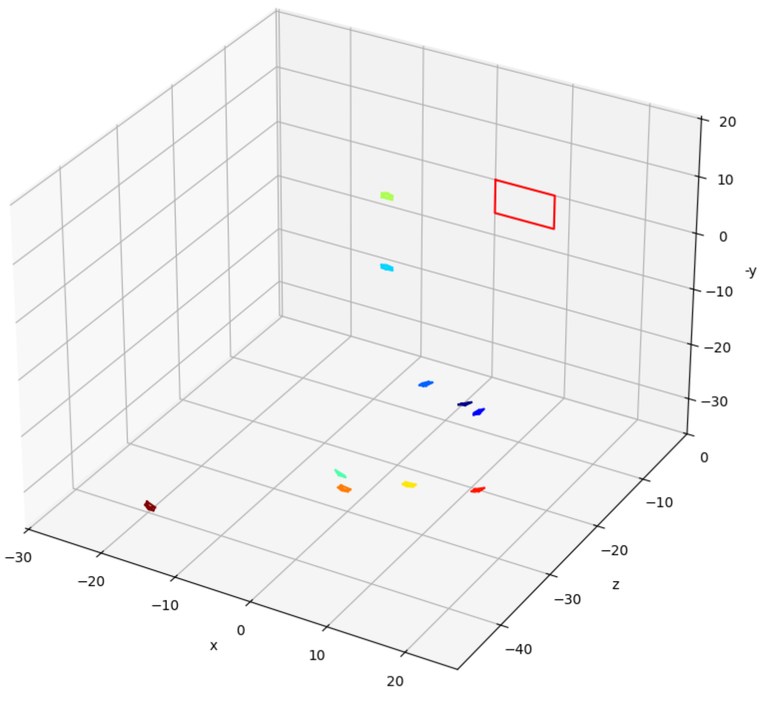


With our image

Extrinsic Parameters Visualization, use_opencv=True



Extrinsic Parameters Visualization, use_opencv=False



4. Discussion

We noticed that the intrinsic parameter matrix we calculated differed slightly from the one computed by OpenCV. The focal lengths and offsets were not significantly different, but OpenCV sets the skew to 0 by default, and the computed K matrix was an upper triangular matrix. But in our results, we observed that:

- The K matrix isn't perfectly upper triangular matrix
- The skew paprameter are non-zero

Initially, we thought that only rolling shutter effects when capturing moving objects would cause skew, which would result in the skew parameter in the intrinsic matrix being non-zero. However, we later found that skew can also occur if the object being photographed is not perfectly parallel to the optical axis of the camera. We also found that after performing Cholesky decomposition, the results of np.linalg.inv(L.T) and np.linalg.inv(L).T were different. The former resulted in an upper triangular matrix, while the latter had non-zero elements in the lower triangular part.

5. Conclusion

In this assignment, we implemented the camera calibration process from scratch without relying on OpenCV, following the mathematical derivation to calculate the homography matrix, intrinsic matrix, and extrinsic matrix. When we conducted experiments using photos taken with our mobile phones, the intrinsic parameter matrix we calculated still showed some discrepancies with the focal lengths and offsets from OpenCV. This could be due to OpenCV employing the Levenberg-Marquardt algorithm for non-linear optimization or considering distortion to achieve more accurate camera calibration.