

CV 2021 HW1 Report

Group 24

0610122 李思賢, 309553049 陳怡安

Introduction

Camera calibration is a procedure often used in computer vision to find out the camera parameters. We use the method introduced in class and refer to Zhengyou Zhang's paper: A Flexible New Technique for Camera Calibration to implement our calibration function. Then, we test it on the images given by TAs. We also printed out a chessboard image and took pictures of it from different angles using our cell phone as additional data, and compared the results of OpenCV to the results of our work.

Implementation Procedure

First, we use `cv2.findChessboardCorners` to find out the corner points of the chessboard. We obtain 49 pairs of real-world points and image points from each photo. Refer to Zhengyou Zhang's paper, we can get the homography matrix H by solving the equation :

$$\begin{bmatrix} M^T & 0^T & -uM^T \\ 0^T & M^T & -vM^T \end{bmatrix} x = Lx = 0 \quad H = \begin{bmatrix} h_{11} & h_{21} & h_{31} \\ h_{12} & h_{22} & h_{32} \\ h_{13} & h_{23} & h_{33} \end{bmatrix}$$

(u, v) is the image coordinate, M is the world coordinate matrix $[X, Y, 1]$, and $x = [h_{11}, h_{21}, h_{31}, h_{12}, h_{22}, h_{32}, h_{13}, h_{23}, h_{33}]^T$. We find x by using `np.linalg.svd` to find out the eigenvector of $L^T L$ associated with the smallest eigenvalue. Then rebuild the homography matrix H of the image.

Secondly, use the homography matrix H to find B . B is symmetric so let $b = [b_0, b_1, b_2, b_3, b_4, b_5]^T$ and we find out b by solving the equation :

$$B = \begin{bmatrix} b_0 & b_1 & b_3 \\ b_1 & b_2 & b_4 \\ b_3 & b_4 & b_5 \end{bmatrix}$$

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \quad v_{ij} = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix}$$

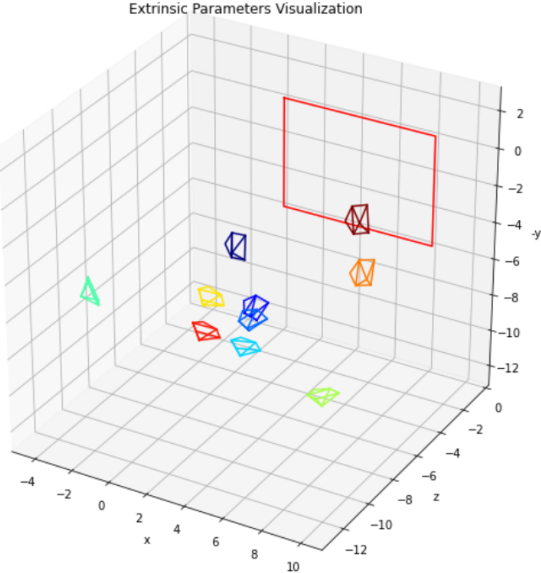
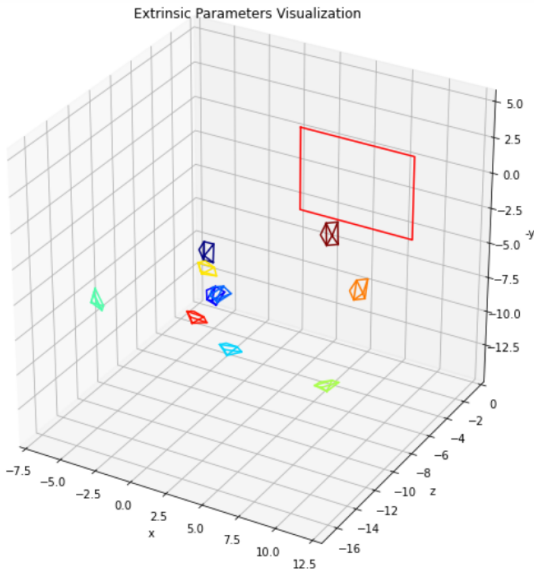
We know that $B = K^{-T}K^{-1}$ and B is positive definite, so we calculate the intrinsic matrix K by using Cholesky factorization to B . `np.linalg.cholesky` returns K^{-T} and we get the K by transport and inverse K^{-T} .

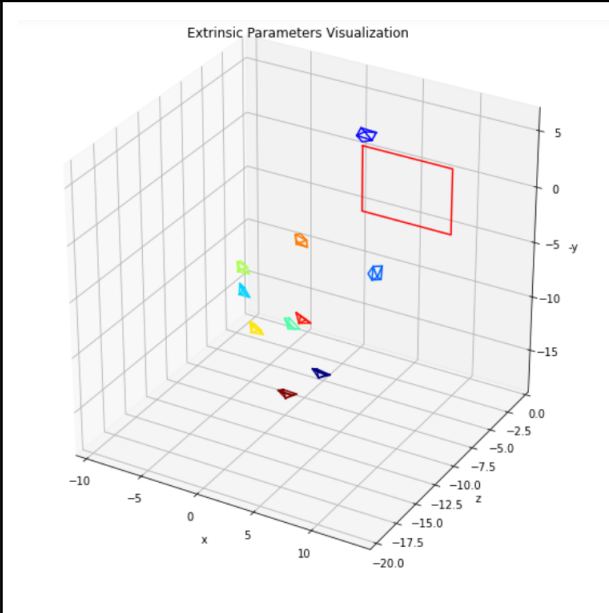
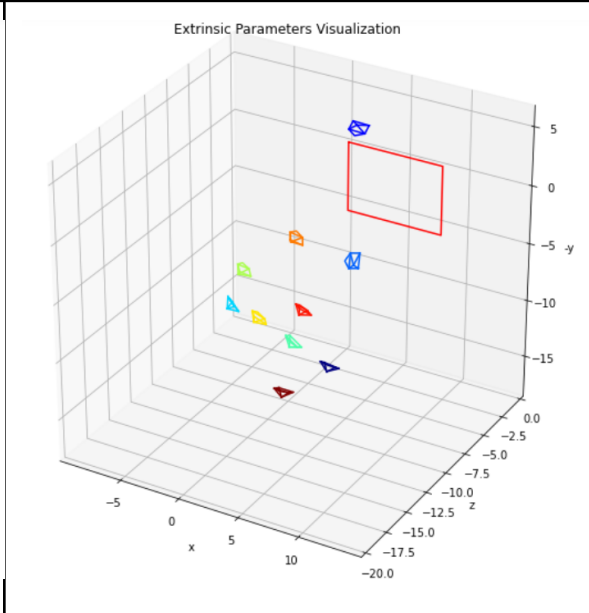
Then, we get extrinsic matrix $[R|t]$ for each image from K and corresponding H by solving the following equations:

$$\begin{aligned} r_1 &= \lambda K^{-1} h_1 \\ r_2 &= \lambda K^{-1} h_2 \\ r_3 &= r_1 \times r_2 \\ t &= \lambda K^{-1} h_3 \end{aligned} \quad \lambda = \frac{1}{\|K^{-1} h_1\|}$$

We obtain the rotation parameters and translation parameters. Finally, we rebuild the extrinsic matrix $[R|t]$ for every image and use the given code to plot the result figures.

Experimental Result

TA's Data	
OpenCV function	Our calibration function
$K :$ $\begin{bmatrix} 2701.93 & 0.00 & 1538.21 \\ 0.00 & 2738.09 & 1960.13 \\ 0.00 & 0.00 & 1.00 \end{bmatrix}$	$K :$ $\begin{bmatrix} 3400.3369 & -35.254288 & 1475.6844 \\ 0.00 & 3349.3596 & 1408.2289 \\ 0.00 & 0.00 & 1.00 \end{bmatrix}$
	

Our Data	
OpenCV function	Our calibration function
$K :$ $\begin{bmatrix} 3292.9153 & 0.00 & 1837.1595 \\ 0.00 & 3348.0628 & 1565.4485 \\ 0.00 & 0.00 & 1.00 \end{bmatrix}$	$K :$ $\begin{bmatrix} 3055.0142 & 9399.9611 & 1915.2863 \\ 0.00 & 3133.2603 & 1794.4666 \\ 0.00 & 0.00 & 1.00 \end{bmatrix}$
	

Discussion

According to our experimental result figures, we found a little displacement about camera positions between our function and OpenCV function. If we take a look at the intrinsic matrices, we can see that most of the elements have similar value using either OpenCV function or our calibration function.

However, there is a big gap between the skew factors in two cases. In our opinion, it's because OpenCV always sets the skew factor to 0, but we don't. For TA's data, we find the skew factor to be -35.25428. For our data, the skew factor is 9399.9611. Therefore our results are not exactly the same as OpenCV's results.

Conclusion

In this assignment, we refer to Zhengyou Zhang's paper to implement camera calibration function from scratch and use the given code to visualize the result. We test our function on two datasets to compare with that using cv2.calibrateCamera. We find they give similar results, but not exactly the same.

Work Assignment Plan

Code: 陳怡安

Photo: 李思賢

Report: 李思賢 70%, 陳怡安 30%

Reference

<https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr98-71.pdf>