IIAI30013

Computer Vision

HW3: Camera Calibration

Instructor: YuanFu Yang

yfyangd@nycu.edu.tw

國立陽明交通大學
NATIONAL YANG MING CHIAO TUNG UNIVERSITY

HW3: Camera Calibration

- Homework due: 4/29 23:59
- Late submissions will incur a penalty of one point for each day overdue.
- The assignment allows a maximum extension of 3 days (it will not be accepted if submitted later than 3 days).
- Submit files: code (HW3_Camera Calibration.ipynb) and report (6 questions), and submit them in both **.zip** and **PDF** file formats respectively.
- This assignment can be carried out using Colab or completed on your PC.

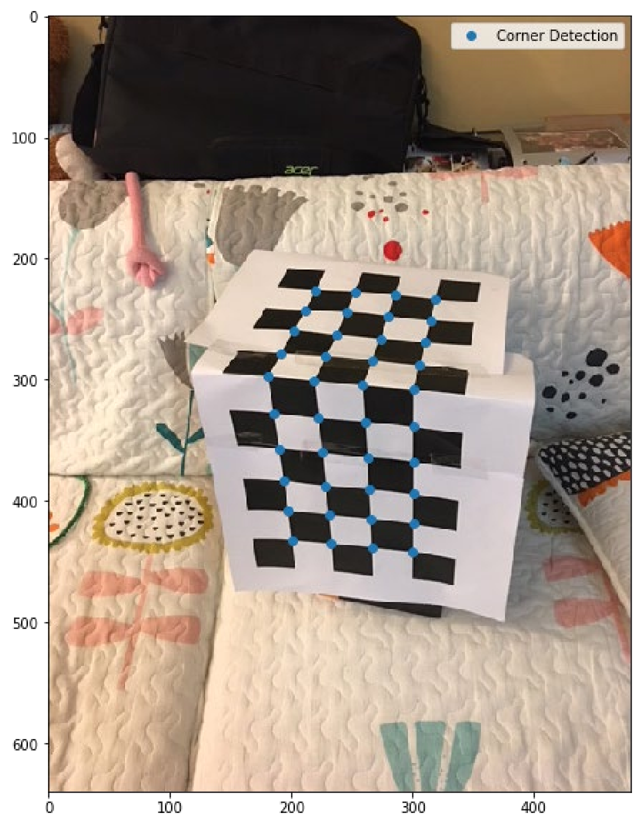
HW3: Camera Calibration

- Perform camera calibration from a set of 3D points & 2D points correspondences in the chessboard images. You can find these images in the zip file.

HW3: Camera Calibration

File Description

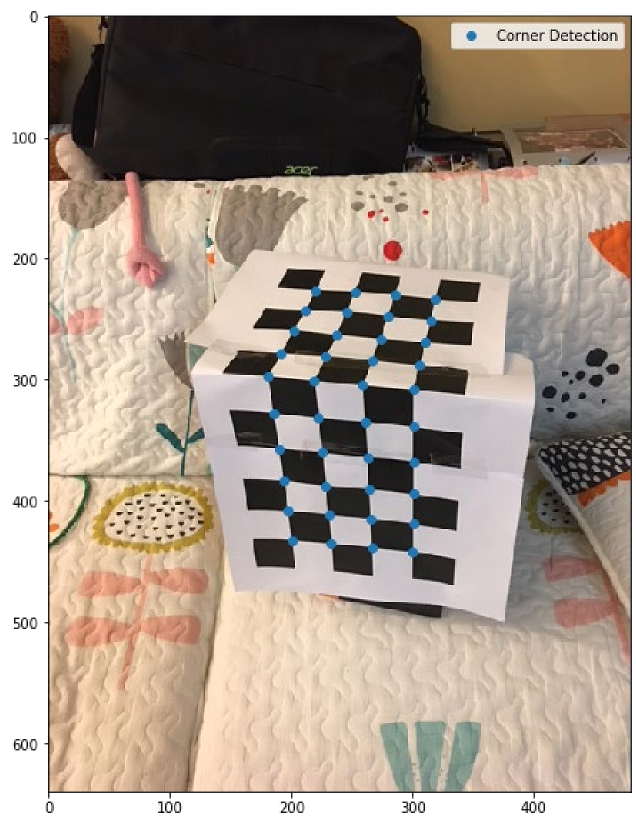
- **Point3D.txt**: 3D World coordinates, a total of 36 points, corresponding to the chessboard box (2D Image coordinates) in the image below.



HW3: Camera Calibration

File Description

- **image1.npy/image2.npy**: 2D Image coordinates, a total of 36 points, corresponding to the image below.



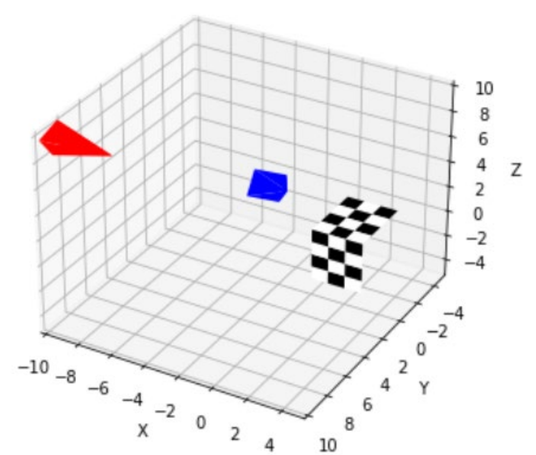
HW3: Camera Calibration

File Description

- **cliker.py**: Used for annotating corners in 2D image coordinates. Of course, you can also use your own method for corner annotation, such as the Harris corner detection taught in previous classes.

Note: When using clicker.py for corner labeling, it is essential to proceed **from left to right** and **from top to bottom** to align with the order of 3D world coordinates.

- **visualize.py**: Used to plot the positions of objects and cameras in 3D space.



HW3: Camera Calibration

Function 1: Project_Matrix(point2D, point3D)

Compute the projection matrix from a set of 2D-3D point correspondences by using the leastsquares (eigenvector) method for each image.

```
def Projection_Matrix(point2D, point3D):
    ########################################################################
    # TODO:                                                                #
    #   Using 2D coordinator and 3D coordinator,                           #
    #   , calculate the 3D to 2D projection matrix P                       #
    ########################################################################



    ########################################################################
    #                                                                      #
    #                        End of your code                              #
    #                                                                      #
    ########################################################################
    return M
```

HW3: Camera Calibration

Function 2: KRt(P)

Decompose the two computed projection matrices from (A) into the camera intrinsic matrices K, rotation matrices R and translation vectors t by using the Gram-Schmidt process.

```python
def KRt(P):
    ########################################################################
    # TODO:                                                                #
    #   Extract the intrinsic matrix (K), rotation matrix (R)              #
    #   , and translation vector(T) from the projection matrix.            #
    ########################################################################




    ########################################################################
    #                                                                      #
    #                           End of your code                           #
    #                                                                      #
    ########################################################################
    return K2, R1, T
```

HW3: Camera Calibration

Report

- A. Compute the projection matrix from a set of 2D-3D point correspondences by using the least-squares (eigenvector) method for each image (10%).
- B. Decompose the two computed projection matrices from (A) into the camera intrinsic matrices K, rotation matrices R and translation vectors t by using the Gram-Schmidt process. Any QR decomposition functions are allowed. The bottom right corner of intrinsic matrix K should be normalized to 1. Also, the focal length in K should be positive (10%).
- C. Re-project 2D points on each of the chessboard images by using the computed intrinsic matrix, rotation matrix and translation vector. Show the results (2 images) and compute the point re-projection root-mean-squared errors (10%).
- D. Plot camera poses for the computed extrinsic parameters (R, t) and then compute the angle between the two camera pose vectors (10%).
- E. Print out two “chessboard.png” in the attached file and paste them on a box. Take two pictures from different angles. For each image, perform the steps above (A ~ D) (40%).
- F. Instead of mark the 2D points by hand, you can find the 2D points in your images automatically by using corner detection, hough transform, etc. (20%).

HW3: Camera Calibration

Report

- A. Compute the projection matrix from a set of 2D-3D point correspondences by using the least-squares (eigenvector) method for each image.

Fig.1 Projection Matrix of chessboard_1

```
(array([[ 3.86524050e+01, -7.66400206e+00, -1.10228363e+01,
          8.92799191e+01],
        [-2.71766531e+00,  6.45039261e+00, -4.52114405e+01,
          3.01849673e+02],
        [ 4.20864519e-03, -6.57058041e-02, -5.25791651e-02,
          1.00000000e+00]]),
```

Fig.2 Projection Matrix of chessboard_2

```
array([[ 3.92908079e+01,  1.29659214e+01, -1.16148650e+01,
         1.79404276e+02],
       [-4.87169634e+00,  2.98031221e+00, -4.89285111e+01,
         2.99475624e+02],
       [ 3.63519029e-02, -6.03813844e-02, -5.40887370e-02,
         1.00000000e+00]]))
```

HW3: Camera Calibration

Report

- B. Decompose the two computed projection matrices from (A) into the camera intrinsic matrices K, rotation matrices R and translation vectors t by using the Gram-Schmidt process. Any QR decomposition functions are allowed. The bottom right corner of intrinsic matrix K should be normalized to 1. Also, the focal length in K should be positive.

Fig.3 intrinsic matrices K, rotation matrices R, translation vectors t of chessboard_1

```
K:  (array([[452.80407663,   0.93518602, 175.478448  ],
           [  0.        , 469.04223178, 318.52694972],
           [  0.        ,   0.        ,   1.        ]]),
R:   array([[ 0.99394262,  0.10005206, -0.04547142],
           [-0.09789359,  0.61796918, -0.78008393],
           [ 0.04994907, -0.77981002, -0.62402037]]),
T:   array([-2.26078377, -0.7166517 , 11.86820608]))
```

Fig.4 intrinsic matrices K, rotation matrices R, translation vectors t of chessboard_2

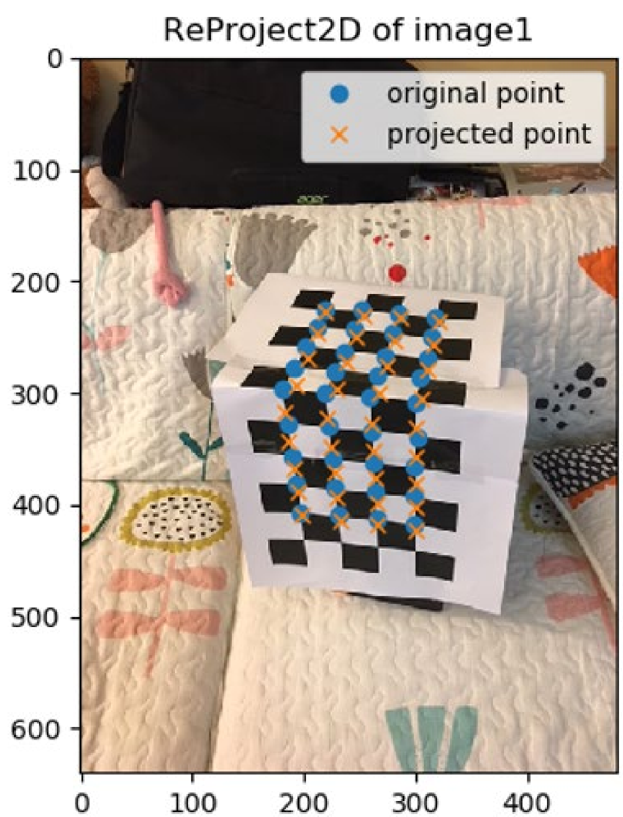
```
K:  (array([[455.83809493,  12.35960452, 161.3625426 ],
           [  0.        , 472.55226679, 308.05967669],
           [  0.        ,   0.        ,   1.        ]]),
R:   array([[ 0.83531043,  0.54751665, -0.04981979],
           [-0.36719729,  0.48816628, -0.7917448 ],
           [ 0.40917312, -0.67964638, -0.60881702]]),
T:   array([ 0.4394187 , -0.22428184, 11.25589263]))
```

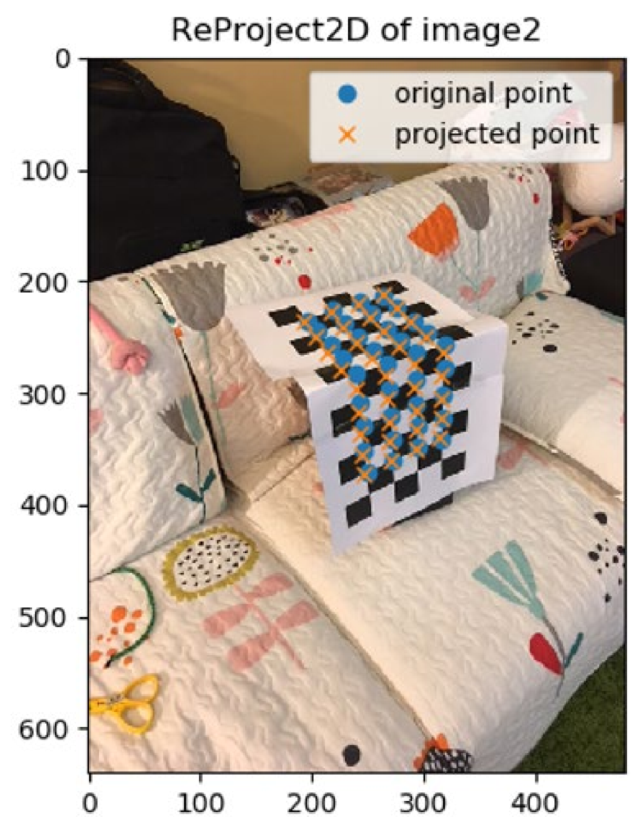
HW3: Camera Calibration

Report

- C. Re-project 2D points on each of the chessboard images by using the computed intrinsic matrix, rotation matrix and translation vector. Show the results (2 images) and compute the point re-projection root-mean-squared errors.

RMSE: 13.14



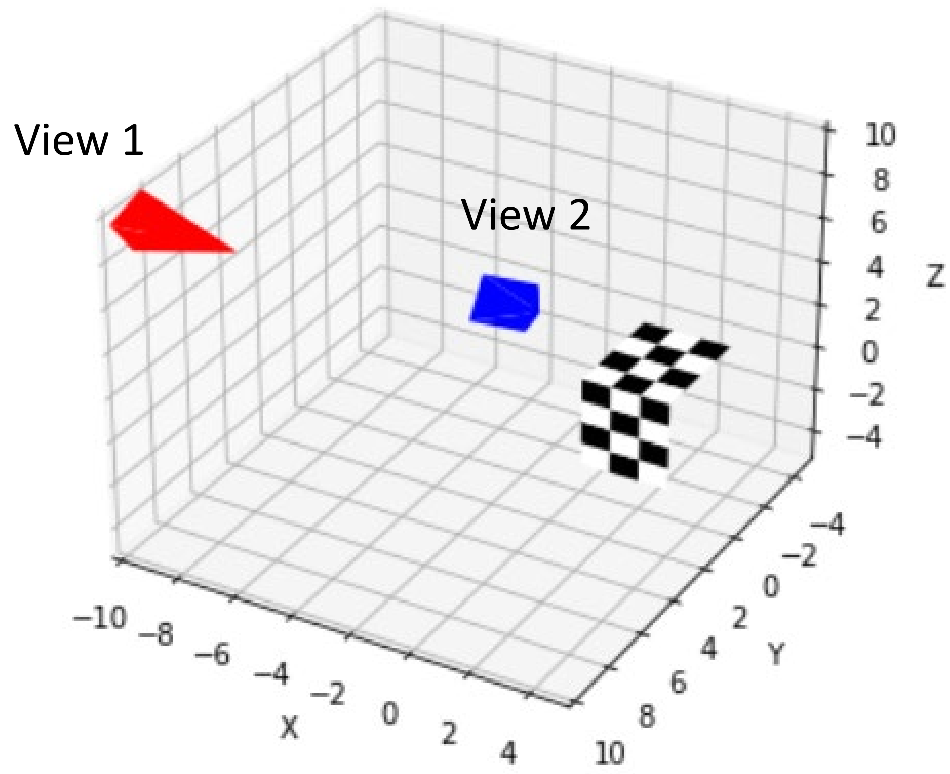


RMSE: 6.22

HW3: Camera Calibration

Report

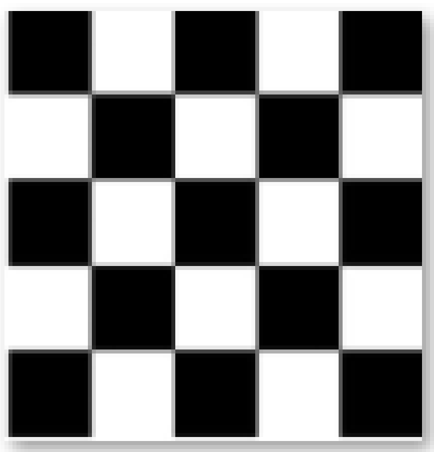
- D. Plot camera poses for the computed extrinsic parameters (R, t) and then compute the angle between the two camera pose vectors.



HW3: Camera Calibration

Report

- E. Print out two “chessboard.png” in the attached file and paste them on a box. Take two pictures from different angles. For each image, perform the steps above (A ~ D).

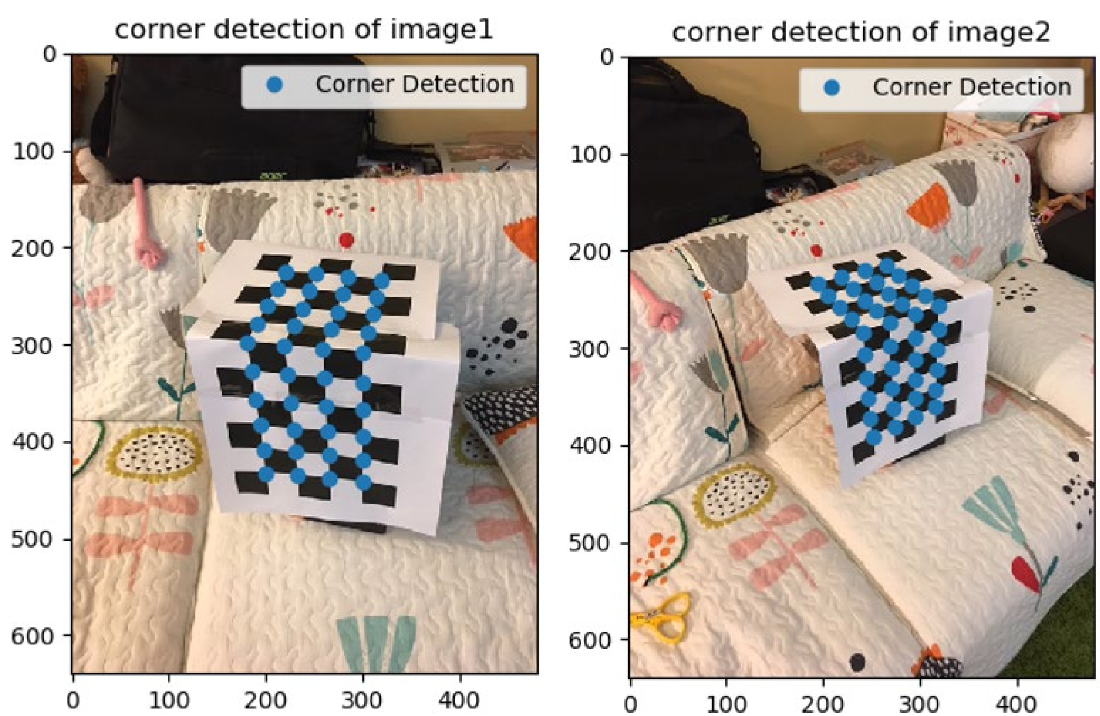


chessboard.png

HW3: Camera Calibration

Report

- F. Instead of mark the 2D points by hand, you can find the 2D points in your images automatically by using corner detection, hough transform, etc.



IIAI30013

Computer Vision

HW3: Camera Calibration

Q & A

Instructor: YuanFu Yang

yfyangd@nycu.edu.tw