Computer Vision HW1

> Group 11 Member: 110705017 何翊華, 313551097 鄭淮薰, 313551098 張欉齡

Introduction

A 2D calibration chessboard is commonly used in computer vision and photogrammetry for camera calibration, which involves estimating the camera's intrinsic and extrinsic parameters.

This chessboard pattern, typically a grid of black and white squares, provides a known geometric structure that helps identify key points on an image. By capturing images of this chessboard from various angles, calibration algorithms can analyze the positions and orientations of these points to accurately determine the camera's internal parameters, such as focal length, optical center, and lens distortion coefficients.

This process is essential for applications that require precise measurements and image rectification, such as 3D reconstruction, robotics, and augmented reality. The simplicity and regularity of the chessboard pattern make it a popular and reliable tool for 2D camera calibration tasks.

Implementation Procedure

The process includes three main steps:

1. Homography Calculation After detecting the chessboard corners, we compute the homography matrix H for each image. The homography matrix represents the transformation between the 3D points on the chessboard and their corresponding 2D projections in the image. Instead of using OpenCV’s findHomography function, the homography matrix is computed manually using Direct Linear Transformation (DLT) based on point correspondences. We set up a system of linear equations and solve them using Singular Value Decomposition (SVD).

```python
def compute_homography(objpoints, imgpoints):
    A = []
    for i in range(len(objpoints)):
        X, Y = objpoints[i][0], objpoints[i][1]
        u, v = imgpoints[i][0][0], imgpoints[i][0][1]
        A.append([-X, -Y, -1, 0, 0, 0, u*X, u*Y, u])
        A.append([0, 0, 0, -X, -Y, -1, v*X, v*Y, v])
    A = np.array(A)
    U, S, Vh = np.linalg.svd(A)
    H = Vh[-1].reshape(3, 3)
    return H / H[-1, -1]
```

2. Intrinsic Matrix Calculation Once we have the homography matrices for all images, we use them to compute the camera's intrinsic parameters. The intrinsic matrix K contains parameters like focal length and principal point, which describe the internal geometry of the camera. Before applying the above methods to compute K, we need to construct the matrix B, which is done by first calculating a set of linear equations. These equations are derived from the homography matrices H and are represented by a matrix V, which is solved using Singular Value Decomposition (SVD).We compute each element of V using the function v_{ij}, where v_{ij} is a function of the homography matrix:

$$v_{ij} = \begin{bmatrix} h_{0i}h_{0j} \\ h_{0i}h_{1j} + h_{1i}h_{0j} \\ h_{1i}h_{1j} \\ h_{2i}h_{0j} + h_{0i}h_{2j} \\ h_{2i}h_{1j} + h_{1i}h_{2j} \\ h_{2i}h_{2j} \end{bmatrix}$$

These equations represent the relationships between the homography matrix elements, We build the matrix V by calling v_{ij} (a function we defined)for pairs of indices and then use SVD to solve for the parameters in B.

```python
def compute_intrinsics(H_list):
    V = []
    for H in H_list:
        V.append(vij(H, 0, 1))
        V.append(vij(H, 0, 0) - vij(H, 1, 1))

    V = np.array(V)
    _, _, Vh = np.linalg.svd(V)
    b = Vh[-1]
    # Solve for B's elements using b
```

After getting b, To calculate the intrinsic matrix K for the camera, we implemented two different methods, each offering a unique approach to derive the intrinsic parameters from the homography matrices.

> Method 1: Cholesky Decomposition

In this method, we first compute the matrix B, which relates to the intrinsic parameters of the camera. To solve for K, we attempt to decompose B using Cholesky decomposition, which is only valid for positive definite matrices. Once decomposed, the intrinsic matrix K is obtained directly from this decomposition. However, this method can fail when B is not positive definite, often due to numerical instability. Though implemented, this method is commented out in the final version in favor of the second method

The corresponding code for Method 1 (commented out) is as follows:

```
# Method 1: Cholesky decomposition
try:
    L = np.linalg.cholesky(B)
except np.linalg.LinAlgError as e:
    print('B is not positive definite. Try -B.')
    L = np.linalg.cholesky(-B)
K = L[2, 2] * np.linalg.inv(L).T
```

> Method 2: Intrinsic Parameters Calculation

In this method, we directly compute the intrinsic parameters fx, fy, cx, and cy from the elements of the matrix B. By solving for these parameters from a set of linear equations derived from the homography matrices, we can calculate the focal lengths and the principal point. This method does not require matrix factorization, making it more robust when the matrix B is not positive definite. It was the chosen method for the final implementation due to its simplicity and reliability. The following equations are solved to calculate the intrinsic parameters:

- $\text{cy} = \frac{B_{12}B_{13} - B_{11}B_{23}}{B_{11}B_{22} - B_{12}^2}$
- $\lambda = B_{33} - \frac{B_{13}^2 + \text{cy} \cdot (B_{12}B_{13} - B_{11}B_{23})}{B_{11}}$
- $f_x = \sqrt{\frac{\lambda}{B_{11}}}$
- $f_y = \sqrt{\frac{\lambda \cdot B_{11}}{B_{11}B_{22} - B_{12}^2}}$
- $\gamma = -\frac{B_{12} \cdot f_x^2 \cdot f_y}{\lambda}$

- $c_x = \frac{\gamma \cdot \text{cy}}{f_y} - \frac{B_{13} \cdot f_x^2}{\lambda}$

  This method is implemented as follows:

  ```
  # Method 2: Intrinsic Parameters Calculation
  cy = (B12 * B13 - B11 * B23) / (B11 * B22 - B12**2)
  lamda = B33 - (B13**2 + cy*(B12*B13 - B11*B23)) / B11
  fx = np.sqrt(lamda / B11)
  fy = np.sqrt(lamda * B11 / (B11 * B22 - B12**2))
  gamma = -B12 * fx**2 * fy / lamda
  cx = (gamma * cy / fy) - (B13 * fx**2 / lamda)

  K = np.array([[fx, gamma, cx],
                [0,  fy,    cy],
                [0,  0,     1]])
  ```

  In summary, we used the function v_{ij} to construct the matrix V from the homography matrices and solve for the elements of the matrix B using SVD. We also implemented two methods for computing the intrinsic matrix K, with Method 2 being the preferred choice due to its robustness.

3. Extrinsic Matrix Calculation Once we have computed the intrinsic matrix K, the next step is to compute the extrinsic parameters for each image. The extrinsic matrix describes the camera's position and orientation in the world relative to the chessboard. This matrix consists of the rotation matrix R and the translation vector t. To calculate these parameters, we normalize the homography matrix and extract the columns to form the rotation matrix. We also compute the translation vector from the third column of the homography matrix.

```python
def compute_extrinsics(K, H):
    h1 = H[:, 0]
    h2 = H[:, 1]
    h3 = H[:, 2]

    lambda_ = 1/np.linalg.norm(np.dot(np.linalg.inv(K), h1))

    r1 = lambda_ * np.dot(np.linalg.inv(K), h1)
    r2 = lambda_ * np.dot(np.linalg.inv(K), h2)
    r3 = np.cross(r1, r2)
    r_matrix = np.array([r1, r2, r3]).T

    t = lambda_ * np.dot(np.linalg.inv(K), h3)
    r = R.from_matrix(r_matrix).as_rotvec()

    rvecs.append(r)
    tvecs.append(t)
```

In this implementation, we use the columns of the homography matrix to compute the rotation matrix and translation vector. The rotation matrix is converted to a rotation vector using Rodrigues' formula.

Experimental Result

> Note: All the result images are stored in the `output` folder, including `Figure1.png`, `Figure2.png`, and `Figure3.png`, which represent the use of data provided by the TA, semicircle data (my_data), and cycle data (my_data), respectively.

data

First, we used the images provided by the TA, which includes 10 images taken from different angles. Based on the implementation method mentioned above, we calculated the intrinsics and extrinsics and plotted the Extrinsic Parameters Visualization. As shown in Figure 1, the red box represents the chessboard, and the camera viewpoints from which the chessboard was captured are also indicated in the figure.

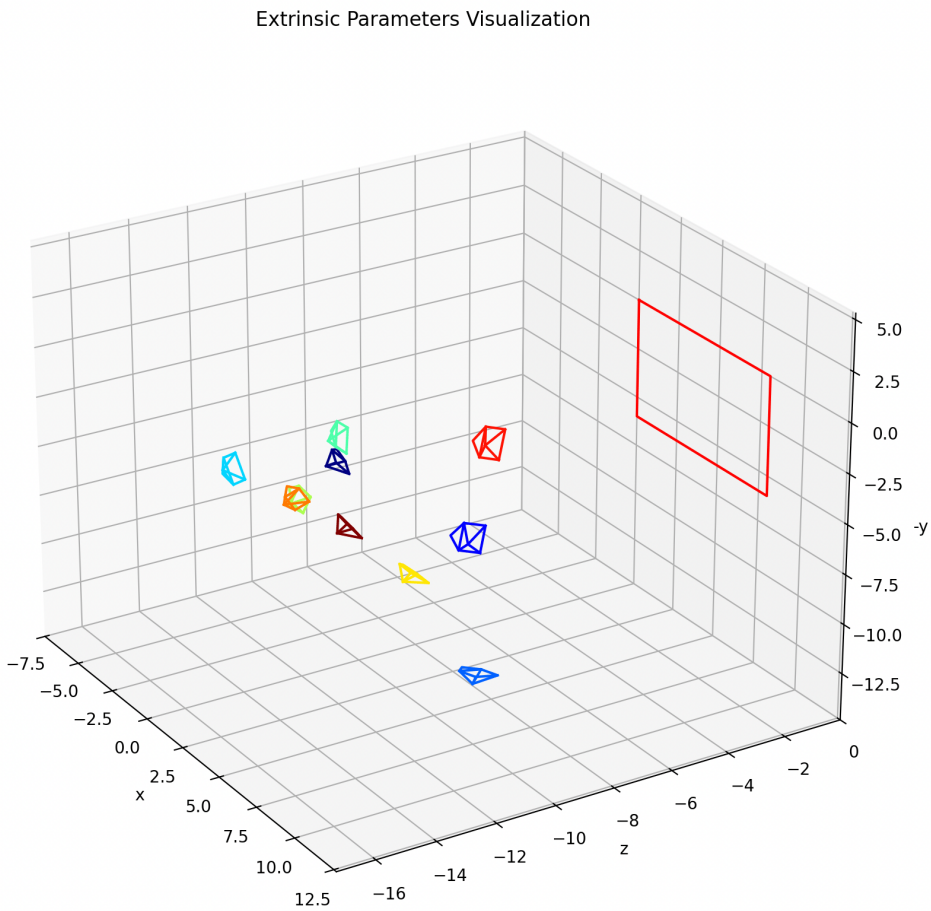


Figure 1: Extrinsic Parameters Visualization (use images in `data`)

my_data

In the next part, we conducted experiments using photos we took ourselves, keeping the camera focus fixed at 0.5. We captured two sets of images in total. For the first set, the camera followed a semi-arc trajectory, while for the second set, the camera circled around the chessboard for the shots.

1. Image sets 1

   - camera trajectory: semiarc
   - number of images: 12
   - folder path: Group11_HW1_code/my_data/semicircle

     To test our dataset, the folder path needs to be changed to `my_data/semicircle` and the corner values should be adjusted accordingly.

In Image Set 1, we captured a total of 12 photos, as detailed in the table below. The results in Figure 2 show that the calculated camera trajectory forms a semicircle, which accurately reflects the setup during the photo capture process.

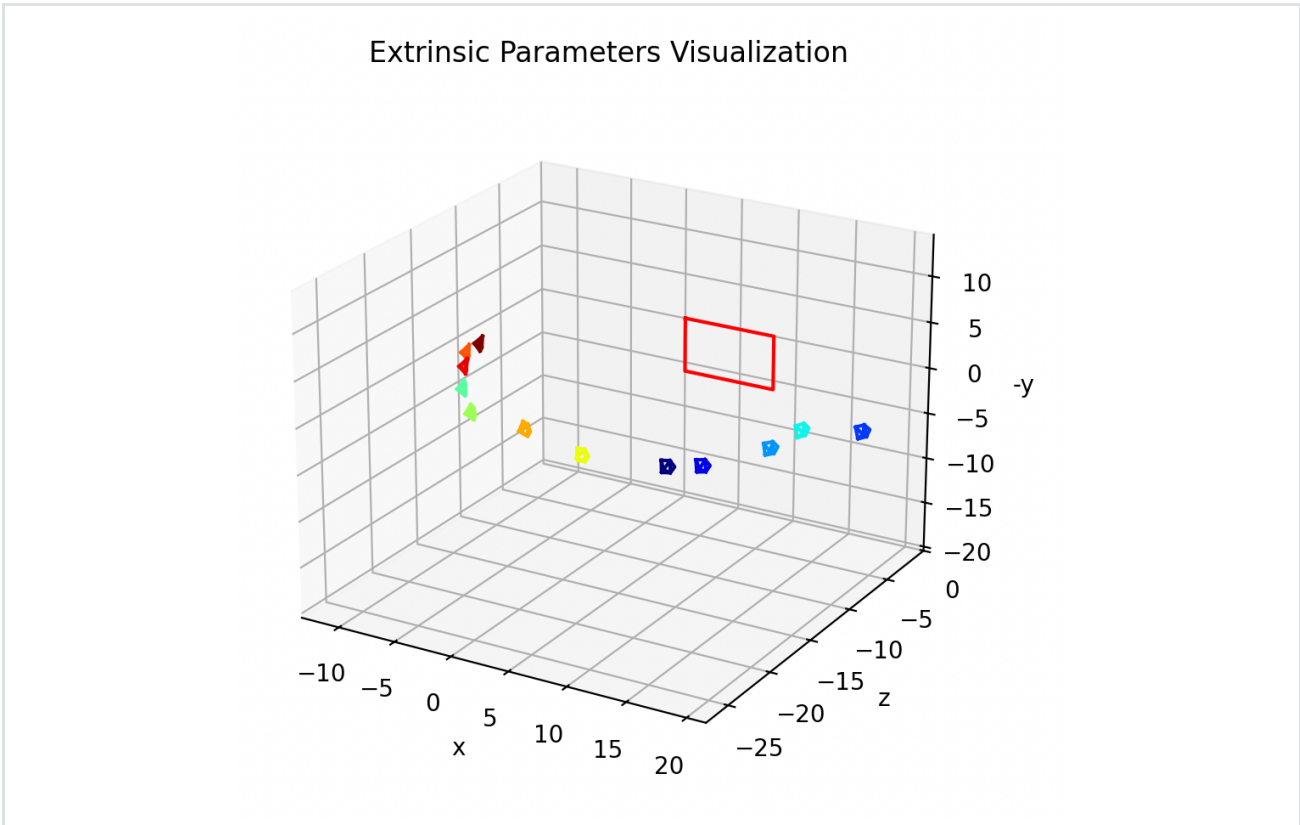
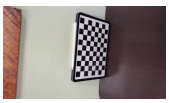


Figure 2: Extrinsic Parameters Visualization (use images in `my_data/semicircle`)

2. Image sets 2

> - camera trajectory: cycle
> - number of images: 130
> - folder path: Group11_HW1_code/my_data/cycle
>
> To test our dataset, the folder path needs to be changed to `my_data/cycle` and the corner values should be adjusted accordingly.

For Image Set 2, we used a video recording method, capturing footage while circling around the chessboard. We then extracted 5 frames per second from the 26-second video, resulting in a total of 130 images used as our data. As shown in Figure 3, the 130 camera viewpoints form a complete circle around the red box, which aligns with our recording method.

Frame 10	**Frame 20**	**Frame 30**	**Frame 40**	**Frame 50**	**Frame 60**
					
Frame 70	Frame 80	Frame 90	Frame 100	Frame 110	Frame 120
					

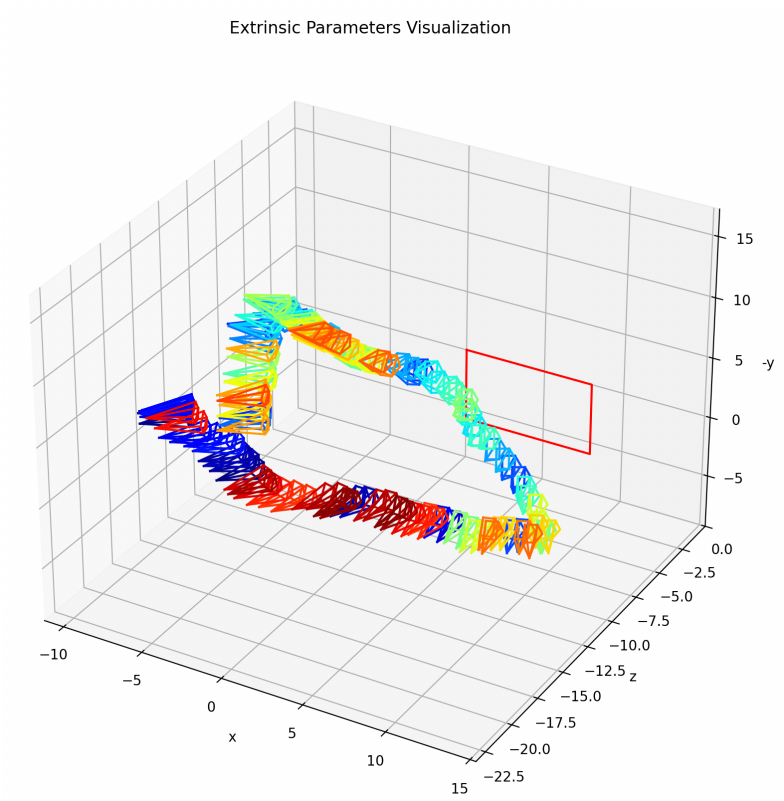
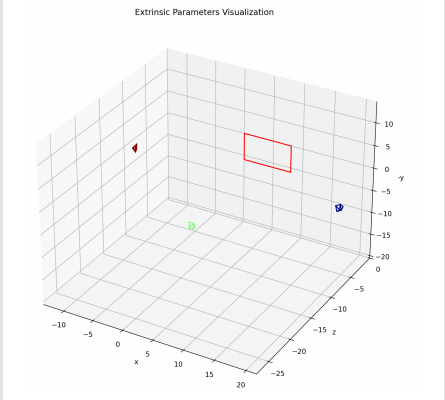
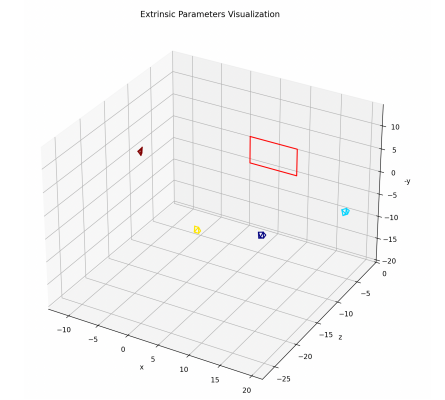
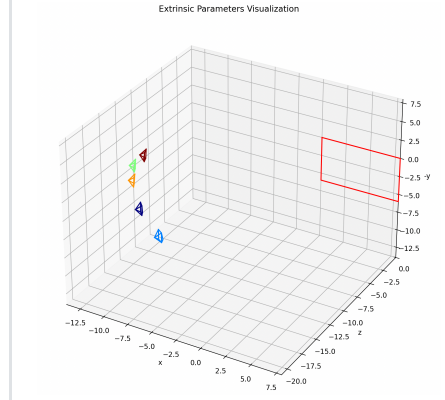


Figure 3: Extrinsic Parameters Visualization (use images in `my_data/cycle`)

Discussion

Minimum Photo Requirements for Successful Calibration

Through our experiments, we found that as long as each photo has a distinct angle, using 3 to 5 photos or more is sufficient to successfully compute the intrinsic and extrinsic matrices.

3 photos	4 photos	5 photo
		

However, if the number of photos is low and the camera angles are too similar, it may not be possible to compute the matrices successfully. For example, as shown in the images below, the lack of variation in angles can result in unsuccessful calibration.



Addressing Orientation Issues in Intrinsic Matrix Calculation Methods

In the Intrinsic Matrix Calculation part of the project, we implemented two different methods to calculate the intrinsic matrix K, These methods are:

1. Method 1: Cholesky Decomposition
2. Method 2: Intrinsic Parameters Calculation

Method 1 relies on the assumption that matrix B, which is derived from the homography matrices, must be positive definite. Cholesky decomposition is only valid for positive definite matrices, which means all the eigenvalues of B must be positive. However, in practice, we observed that sometimes B had negative values.

Upon further investigation, we noticed that in cases where B is not positive definite, all the eigenvalues of B are either negative or all are positive. This suggests that the problem lies in the orientation of the transformation rather than an intrinsic issue with the matrix. Specifically, if the direction of the transformation (i.e., how the object points are mapped to the image points) is reversed, it causes the matrix to flip its sign, making B negative.

To address this issue, we explored adding a negative sign to B when we detect that it is not positive definite. This correction involves checking if B has negative eigenvalues and, if so, multiplying B by –1. After applying this correction, the matrix becomes positive definite, and we can then proceed with Cholesky decomposition.

2D calibration chessboard vs 3D calibration rig

A 2D chessboard is relatively inexpensive, lightweight, and easy to print or create, whereas a 3D calibration rig often involves more complex construction and higher costs. The 2D chessboard can be easily positioned and oriented in different ways to cover various angles, which is sufficient for most camera calibration needs, especially in environments where precise 3D measurements are not critical.

Additionally, 2D calibration with a chessboard pattern is less computationally intensive compared to 3D calibration, making it faster and more efficient for scenarios where quick calibration is needed. This approach is particularly beneficial for applications like augmented reality, robotics, and general computer vision tasks where precise intrinsic parameters are essential, but the full spatial configuration of a 3D structure is not necessary.

Conclusion

In conclusion, our exploration of camera calibration using a 2D chessboard pattern has demonstrated the effectiveness and practicality of this method for estimating intrinsic and extrinsic parameters. We successfully calculated the homography, intrinsic, and extrinsic matrices using images taken from various angles, both from TA-provided datasets and our own experimental data. The results showed that the calculated camera trajectories closely matched the actual camera movements, validating our approach.

Overall, this project has provided valuable insights into camera calibration techniques and reinforced the utility of the 2D chessboard as a reliable calibration tool in computer vision.

Work Assignment Plan

- 110705017 何翊華
  - Code Implementation : Implementing the homography matrix, also responsible for debugging the code and ensuring that the final results are correct.
  - Dataset Design : Preparing and designing the calibration image dataset
  - Discussion and Collaboration
  - Report Writing
- 313551097 鄭淮薰
  - Code Implementation : Implementing the intrinsic and extrinsic parameter calculations, and calculating the rotation and translation vectors.
  - Mathematical Research : Focus on the mathematical theory behind camera calibration, particularly how to derive the intrinsic and extrinsic parameters from the homography matrix.
  - Discussion and Collaboration

    - Report Writing
- 313551098 張欀齡
    - Code Implementation : Implementing the intrinsic and extrinsic parameter calculations, and calculating the rotation and translation vectors.
    - Mathematical Research : Focus on the mathematical theory behind camera calibration, particularly how to derive the intrinsic and extrinsic parameters from the homography matrix.
    - Discussion and Collaboration
    - Report Writing