CS5330: Pattern Recognition and Computer Vision

Project 4: Calibration and Augmented Reality

Name: Krishna Prasath Senthil Kumaran

Email: senthilkumaran.k@northeastern.edu

Aim of the Project:

This project helps us understand, how to calibrate a camera, find the position of the camera and how to put a virtual object on the target image. The system can detect the chessboard as the target and place a virtual object on it, even if the target or the camera moves, the virtual object is within the target image.

I have tried using a normal webcam and my phone camera as a webcam as well.

Task 1: Detect and Extract Chessboard Corners

In task 1, the system can detect the target and extract corners using ***"findChessboardCorners", "cornerSubPix",*** and ***"drawChessboardCorners".*** Fig.1 shows the system detecting and drawing the chessboard corners.

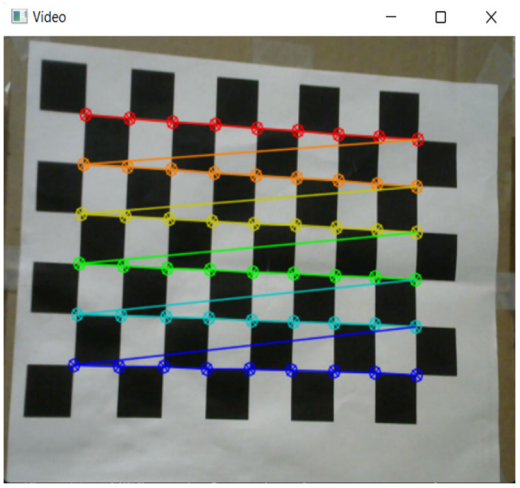

Fig.1 Detecting and drawing the chessboard corners.

Task 2: Select Calibration Images

In task 2, we select the calibration images by pressing the letter ***'s'***. Each time when we press this key a frame is captured from the live video. We have to keep on moving the camera to different positions and orientations every time before the key press. We need a minimum of 5 frames. Fig.2 shows the prompt displayed when we select the calibration images.

```
Capturing calibration images...
Capturing calibration images...
Capturing calibration images...
```

Fig.2 Capturing calibration images.

Task 3: Calibrate the camera

In task 3, after capturing the required number of calibration images we are ready to calibrate the camera. By pressing the letter ***'c'***, the system calibrates the camera. Once the camera is calibrated we can see the ***"camera matrix", "distortion coefficient",*** and ***"re-projection error"*** being printed out on the terminal. Fig.3 show all three being printed out.

```
Camera Calibrated!
Chessboard Camera Matrix:
575.255, 0, 165.155,
0, 556.078, 100.055,
0, 0, 1,
Chessboard Distortion Coefficients:
0.00120202, 0.25232, -0.0163527, 0.0134567, 0.477511,
Chessboard Re-projection Error: 0.304654
```

Fig.3 Camera matrix, distortion coefficient and re-projection error.

Task 4: Calculate the current position of the camera.

In task 4, using the camera parameters the system can calculate the current pose of the chessboard. Fig.4 shows the rotational and translational data being printed out in real-time on the terminal.

```
Rotational data of the chessboard:
[-3.0705212185761;
 -0.2071302907754813;
 -0.1554423570227839]
Translational data of the chessboard:
[0.9103163985446463;
 -5.905504156196728;
 59.31108353958815]
Rotational data of the chessboard:
[-3.077350299438706;
 -0.2066822087119761;
 -0.1435085633189931]
Translational data of the chessboard:
[0.9680697181263942;
 -5.942081273247136;
 59.46646132052258]
```

Fig.4 Rotational and translational data being calculated in real time.

Task 5: Project Outside Corners or 3D Axes

In task 5, the system is able to calculate the outermost 4 corners of the chessboard using the ***"projectPoints"*** function. Fig.5 shows the 4 corners being highlighted.

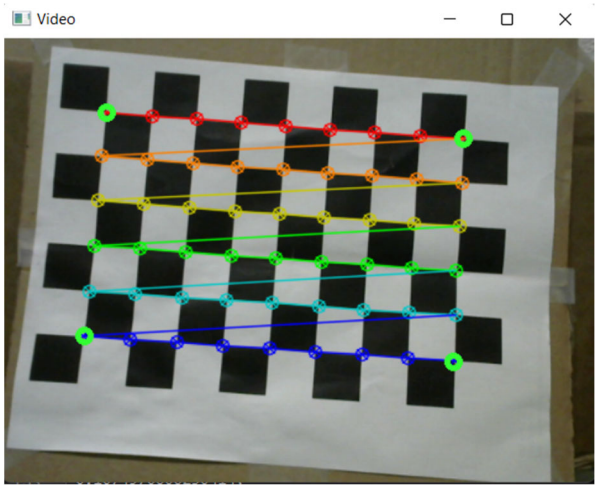

Fig.5 4 corners of the chessboard.

Task 6: Create a Virtual Object

In task 6, a virtual object is created and placed within the four corners of the chessboard. The sides of the virtual object are also filled. Fig.6 shows the virtual object being displayed.

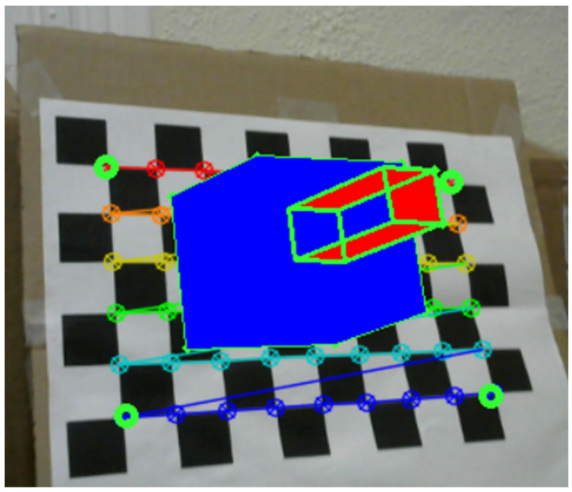

Fig.6 Virtual Object

Task 7: Detect Robust features

Harris corner has been implemented in this task. The program detects the features in a cube and chessboard.

We first figure out the 2D aspects of the image and the 3D coordinates of the real world. If we can project the 2D point into the real world, then we will be able to calibrate the camera. This will help us to place a virtual object on the desired shapes. Fig.7 shows the detection of feature points from the shapes shown to the program.

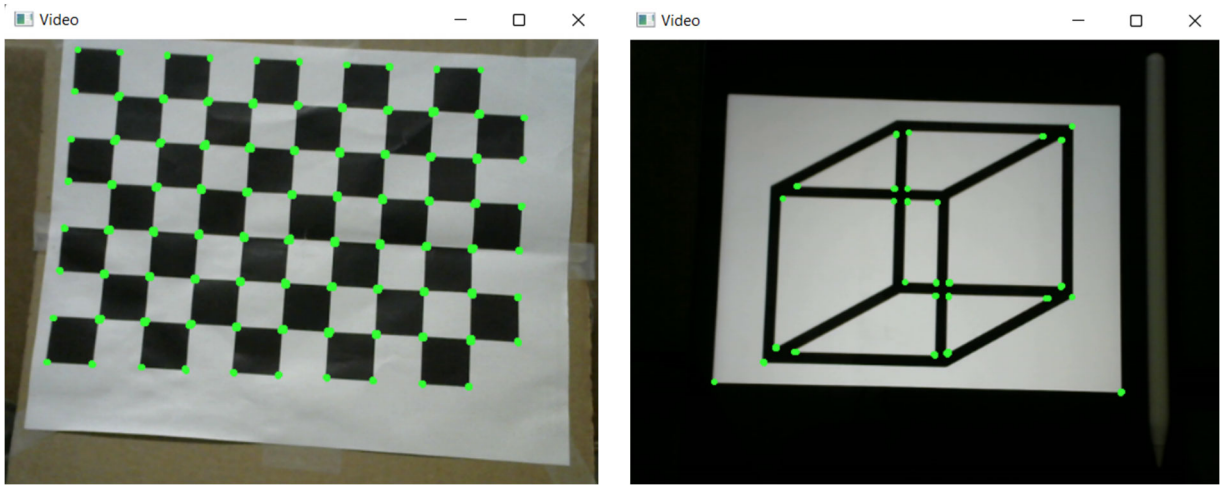

Fig.7 Detection of feature points using Harris corner.

Extension 1: Test out several different cameras and compare the calibrations and quality of the results

For this extension, I have used a Logitech webcam and my iPhone camera. Based on the results obtained for these two cameras, the webcam has lower reprojection error than the iPhone camera which shows the webcam has more accuracy in 3D to 2D point projection. Webcam also seems to have a lower distortion coefficient compared to the iPhone camera. The distortion coefficient might be more in the iPhone camera because of its lens. Fig.8 shows the calibration results of these camera models.

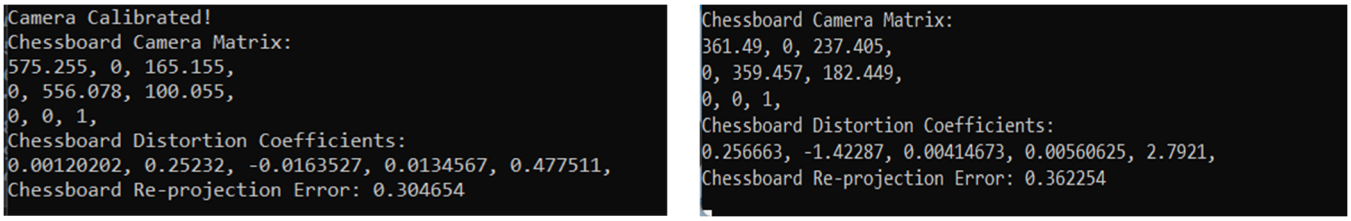

Fig.8 Left: Calibration result for a webcam; Right: Calibration result for iPhone.

Acknowledgement of resources or people consulted for this project:

- OpenCV Tutorials: https://docs.opencv.org/4.5.1/index.html
- Camera Calibration (theory): https://learnopencv.com/camera-calibration-using-opencv/
- Camera calibration functions: https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga549c2075fac14829ff4a58bc931c033d