Leaksmy Heng

CS5330

Project 4

March 01, 2025

Report

Short Description of Overall Project

This project was fun and challenging. The part where we were asked if the rotation and translation vector made sense were insightful because on top of having the code to work, we had to if the number were right. So that was great. Then the part where we have to project a virtual object in 3D world space above the board was also great because I was able to see the cube or triangle that I've put on to the 2D chessboard image.

Short Reflection of What I Have Learned

I've learnt a lot about camera calibration and pose estimation. It was interesting to see the 3D projection onto the 2D representation on the chessboard. I have fun with it especially on the cube that I projected on to the chessboard. Harris corner was also fascinating. I did have fun playing around with the parameters. However, it was frustrating because I have only 2 cores on my computer with about 10 GB left, so the video was lagging. But overall, I think I've learnt a lot of camera calibration.

Task1 Detect and Extract Target Corners

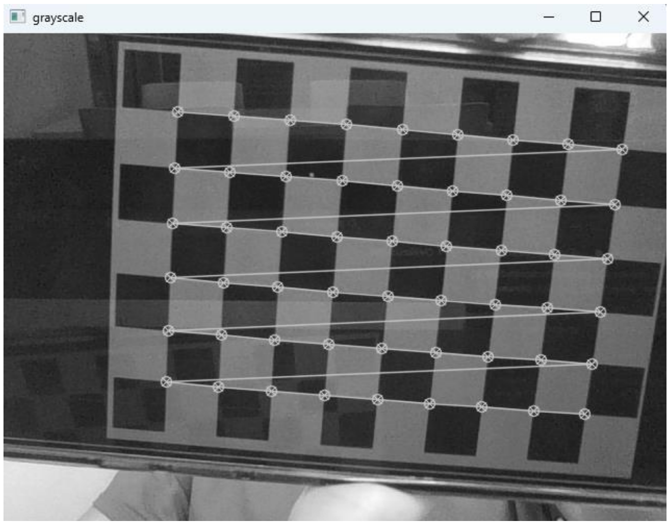
I was using checkerboard.png provided in the project as a target. To implement this task, first I had the program open the video frame from my laptop. I converted the RGB frame to grayscale frame. After that I utilized OpenCV findChessboardCorners function. The function takes 3 arguments:

- Image: I used grayscale image that I have converted from the frame
- Pattern size: 9x6 as instructed in the instructions
- Corners: used vector that stores 2D data (x, y) through std::vector<cv::Point2f> corners

The method outputs Boolean to indicate whether the chessboard corners are found. If the chessboard is found, I utilized OpenCV drawChessboardCorners to draw the corner. I showed it when I was testing but I commented it out in PROD. The function outputs the corners vector that we have initiated on top. I then used the corners output to print out the total number of corners found in chessboard.

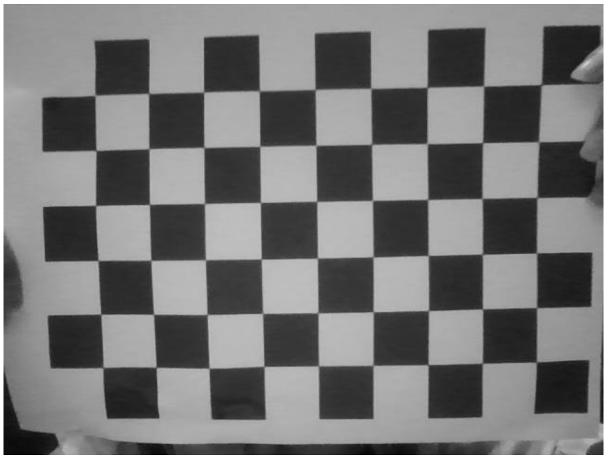
There are some limitations to implementing this including:

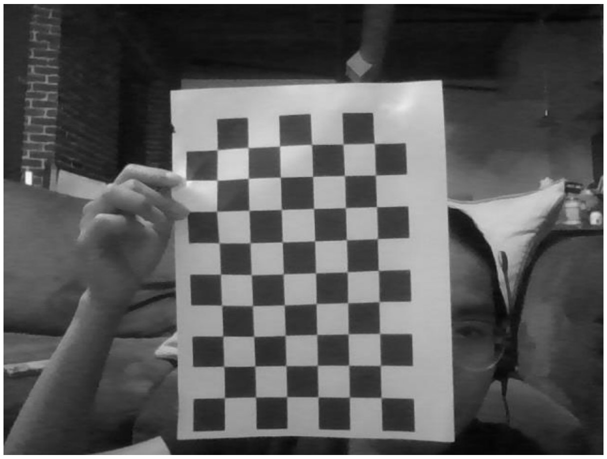
- If I tilted the chessboard paper to an extreme angle, the corners were not detected (i.e if it is vertical, it was not detectable when it’s close to 90 degrees. If it is horizontal, then it was not detectable when it’s close to 0 degree). I think it is because the corner is no longer in a regular grid.
- On top of that, if any of the parts of the chessboard were off the camera’s view field, the corner was also not found.
- Finally, if I deformed the chessboard paper from a planar object to a parabolic object, the corners were also not detected. This is only if I bent the paper until it reaches a certain curve and it became too distorted for the pattern to be detected, so on a certain parabolic angle, it is still detectable.

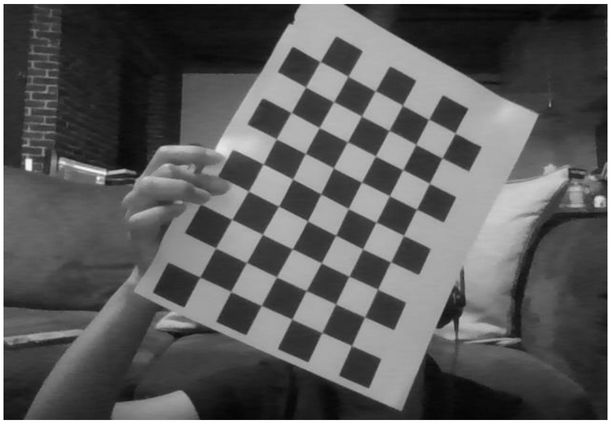


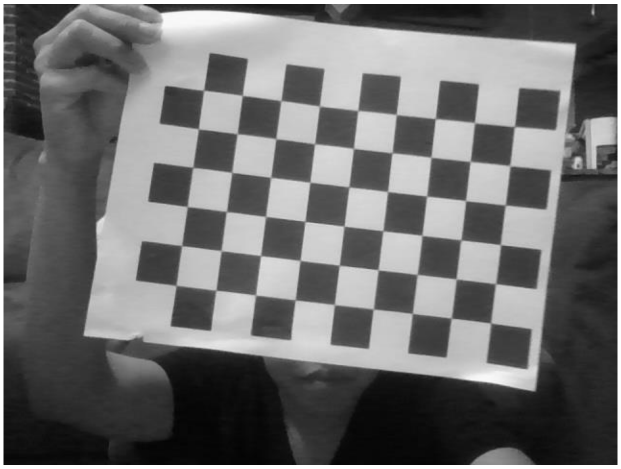
Task2 Select Calibration Images

Some of my calibrated photos include:







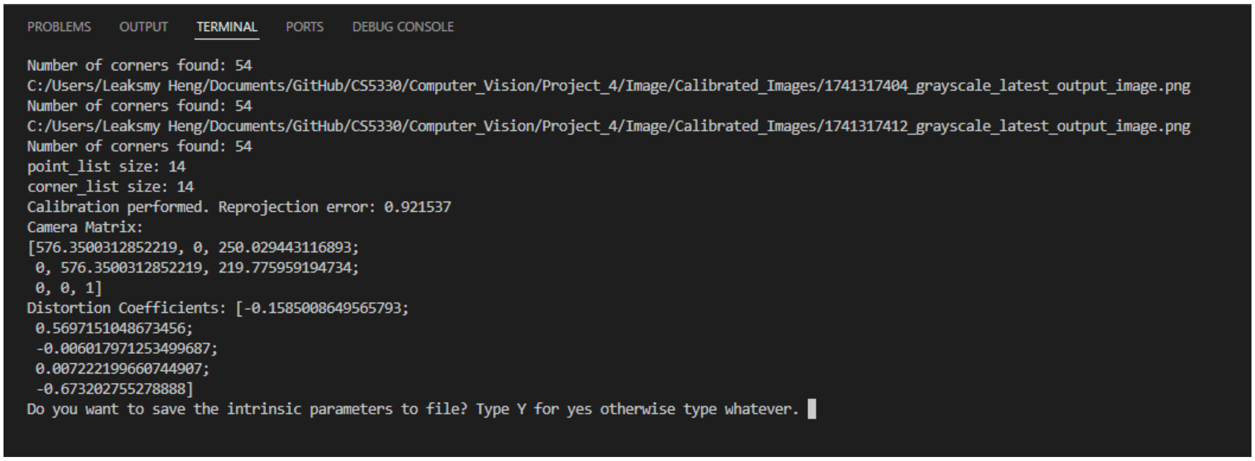


Implementation:

- If users press ‘s’ or ‘S’, the program will grab the **grayscale_latest_output_image.png** from the image directory.
- The program will call **calibration_image_selection** function which is used to calibrate the grayscale_latest_output_image.png. The corners and point_set vector received from **findChessboardCorners** are pushed to the corner_list and point_list vectors.
- To avoid errors on no chessboard image in the camera frame, the program detects and stores the latest image of when the chessboard was on the frame.

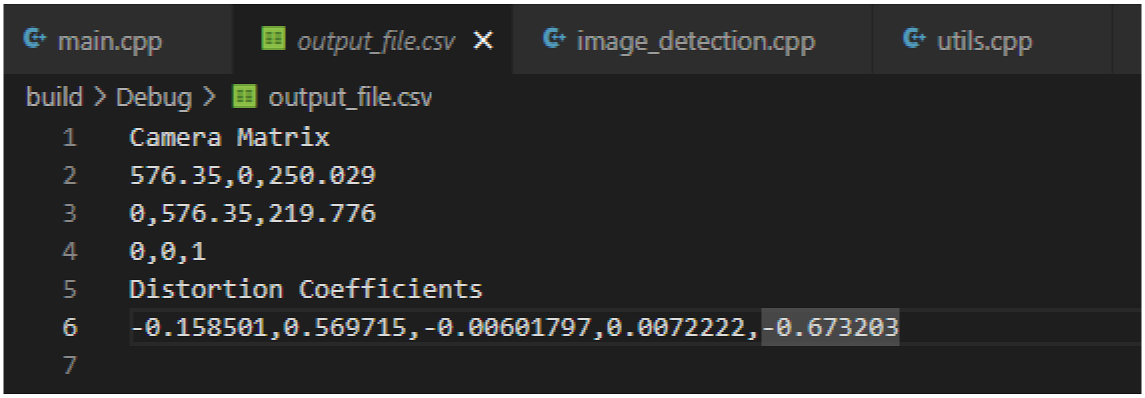
Task3 Calibrate the Camera

Calibration matrix, distortion coefficient and Reprojection error:



```
Number of corners found: 54
C:/Users/Leaksmy Heng/Documents/GitHub/CS5330/Computer_Vision/Project_4/Image/Calibrated_Images/1741317404_grayscale_latest_output_image.png
Number of corners found: 54
C:/Users/Leaksmy Heng/Documents/GitHub/CS5330/Computer_Vision/Project_4/Image/Calibrated_Images/1741317412_grayscale_latest_output_image.png
Number of corners found: 54
point_list size: 14
corner_list size: 14
Calibration performed. Reprojection error: 0.921537
Camera Matrix:
[576.3500312852219, 0, 250.029443116893;
 0, 576.3500312852219, 219.775959194734;
 0, 0, 1]
Distortion Coefficients: [-0.1585008649565793;
 0.5697151048673456;
 -0.006017971253499687;
 0.007222199660744907;
 -0.673202755278888]
Do you want to save the intrinsic parameters to file? Type Y for yes otherwise type whatever.
```

Calibration matrix and distortion coefficient were saved to file “output_file.csv”

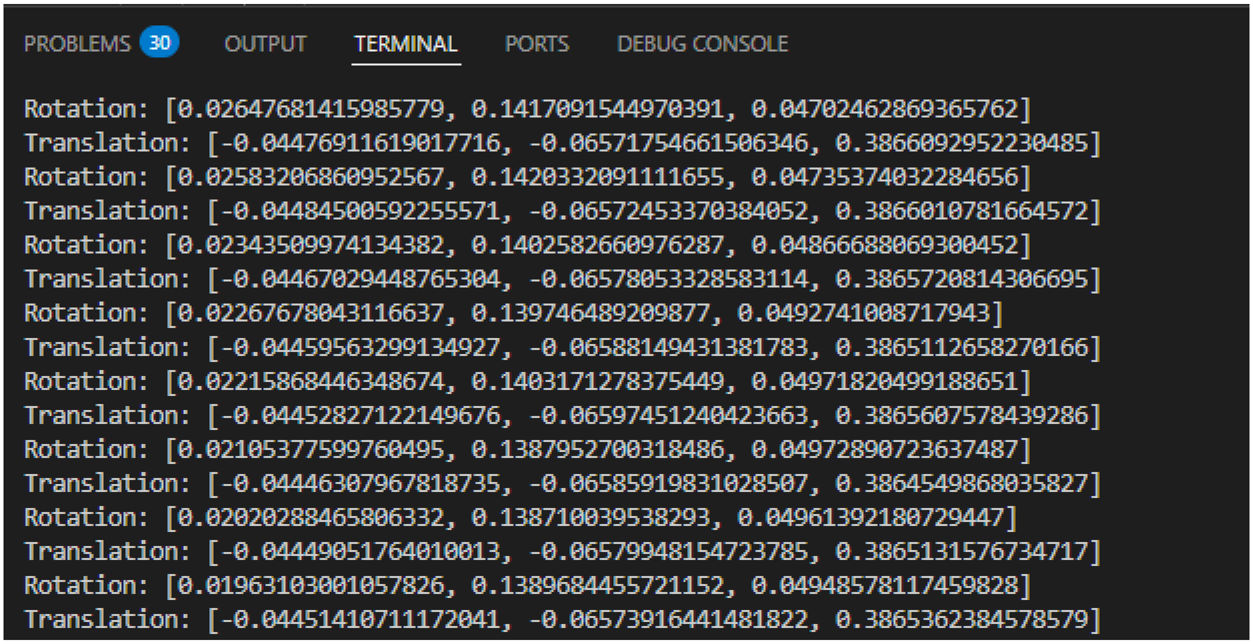


```
Camera Matrix
576.35,0,250.029
0,576.35,219.776
0,0,1
Distortion Coefficients
-0.158501,0.569715,-0.00601797,0.0072222,-0.673203
```

Implementation:

- In the previous task, when user press ‘s’ or ‘S’, and the chessboard image were save successfully to a folder, I have the variable called **number_of_calibrated_images** incremented by one.
- In this task, if users press ‘c’ or ‘C’, the program will check if the number_of_calibrated_images is greater than or equal to 5 or if the images in the Calibrated Image folder are greater than or equal to 5. If one of the conditions are met, the pgram called **camera_calibration** which further called opencv **calibrateCamera.**

Task4 Calculate Current Position of the Camera



```
Rotation: [0.02647681415985779, 0.1417091544970391, 0.04702462869365762]
Translation: [-0.04476911619017716, -0.06571754661506346, 0.3866092952230485]
Rotation: [0.02583206860952567, 0.1420332091111655, 0.04735374032284656]
Translation: [-0.04484500592255571, -0.06572453370384052, 0.3866010781664572]
Rotation: [0.02343509974134382, 0.1402582660976287, 0.04866688069300452]
Translation: [-0.04467029448765304, -0.06578053328583114, 0.3865720814306695]
Rotation: [0.02267678043116637, 0.139746489209877, 0.0492741008717943]
Translation: [-0.04459563299134927, -0.06588149431381783, 0.3865112658270166]
Rotation: [0.02215868446348674, 0.1403171278375449, 0.04971820499188651]
Translation: [-0.04452827122149676, -0.06597451240423663, 0.3865607578439286]
Rotation: [0.02105377599760495, 0.1387952700318486, 0.04972890723637487]
Translation: [-0.04446307967818735, -0.06585919831028507, 0.3864549868035827]
Rotation: [0.02020288465806332, 0.138710039538293, 0.04961392180729447]
Translation: [-0.04449051764010013, -0.06579948154723785, 0.3865131576734717]
Rotation: [0.01963103001057826, 0.1389684455721152, 0.04948578117459828]
Translation: [-0.04451410711172041, -0.06573916441481822, 0.3865362384578579]
```

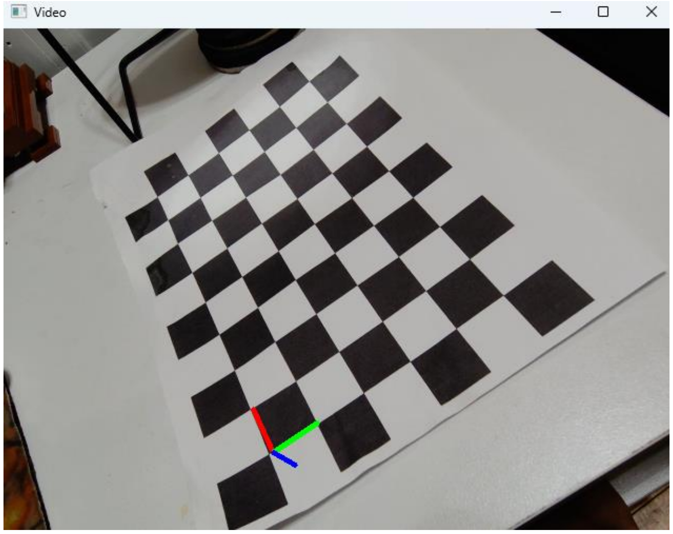
The number does make sense.

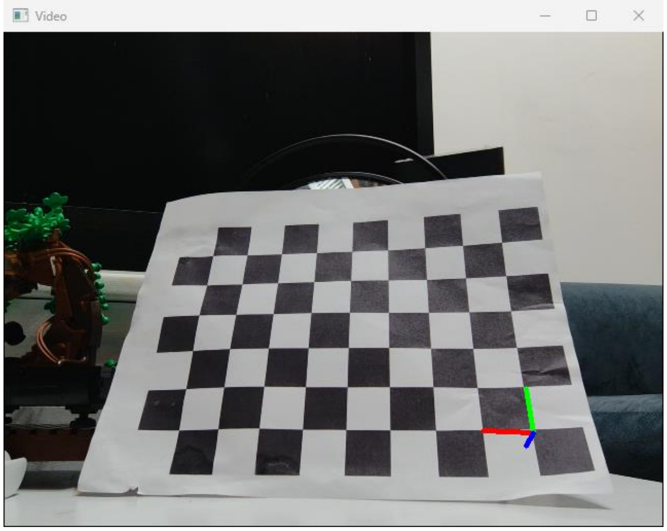
- In rotation, all x, y and z coordinates are positive which mean Z is in front of the camera, X is to the right and Y is down. When I tested this, I moved my chessboard paper in front of the camera in beginning then started to rotate it clocked wise. I think the clockwise rotation causes the Z value increases from 0.047 to 0.049. When I rotated the chessboard paper, my hands were not stable and might have moved downward a little bit which causes the Y axis value to be lower, decreasing from 0.1417 to 0.1389. As for X axis, it drops slightly from 0.026 to 0.019 suggesting that the paper was moved in a horizontal direction.

- In translation, for Z axis, there're only marginal changes in the value suggested that the chessboard move approximately about the same distance to the camera when I rotated it. The Y axis's changed minimally too suggesting that the vertical axis remained relatively stable. For X-axis, it drops marginally indicating that the chessboard photos was moving slightly to the left of the camera's view which are true because when I move it my hand was kind of move to left as I was trying to sneak and look at the screen from the right.

Task5 Project Outside Corners or 3D Axes

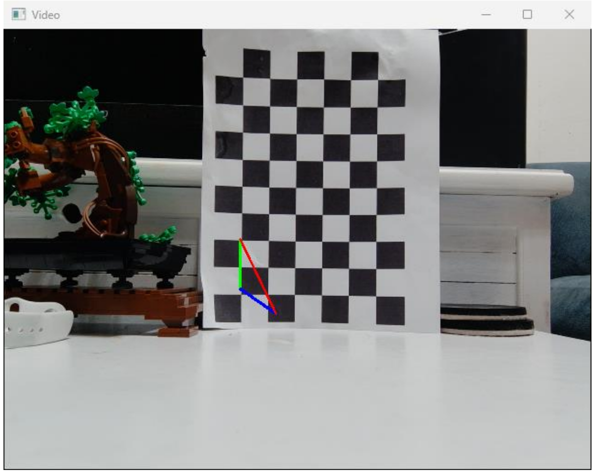
I believe the reprojected points/axes show up in the right places because X, Y and Z are close enough to the chessboard corners. This is due to the projection error is less than 1.

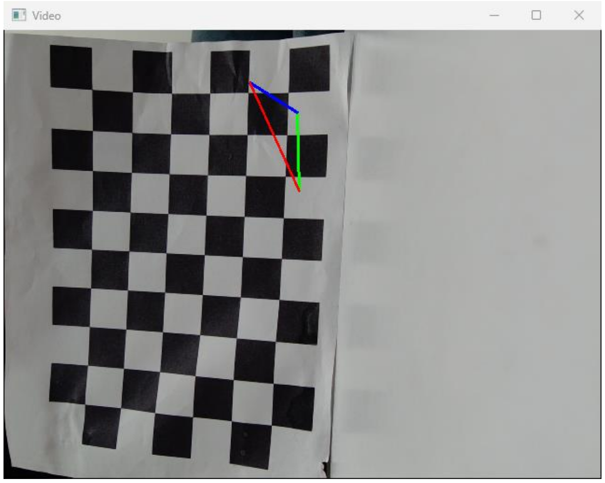




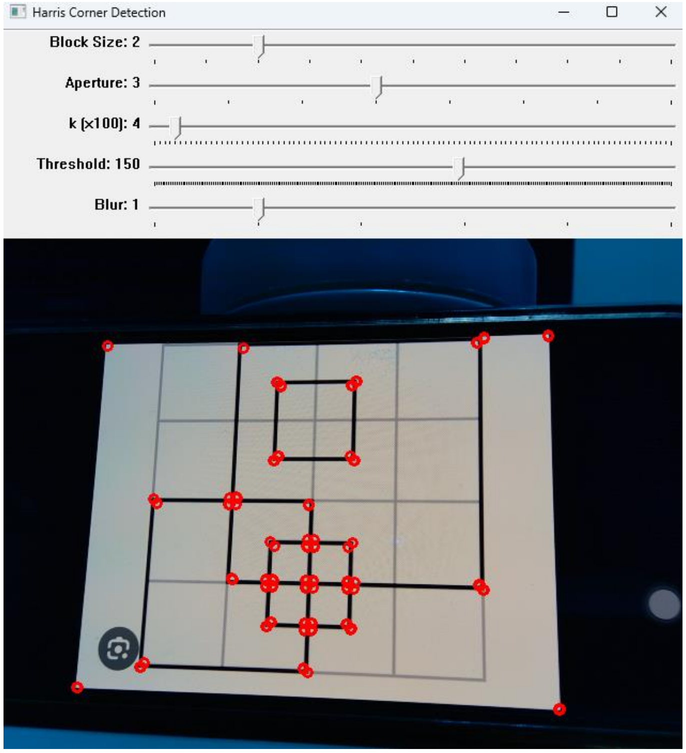
Task6 Create a Virtual Object

I used an asymmetrical triangle floating 5 centimeters above the chessboard. For base point, x and y axis are all 0, but in z axis, I have it 5cm above the chessboard plane. The green line represents the base point. The top point (red line) is offset by 5 centimeters on the x axis and raise 10 centimeters above the chessboard plane. The side point (blue line) is offset by 3 centimeters along the negative x axis, 4 centimeters above the positive y axis and 8 centimeters above chessboard plane.

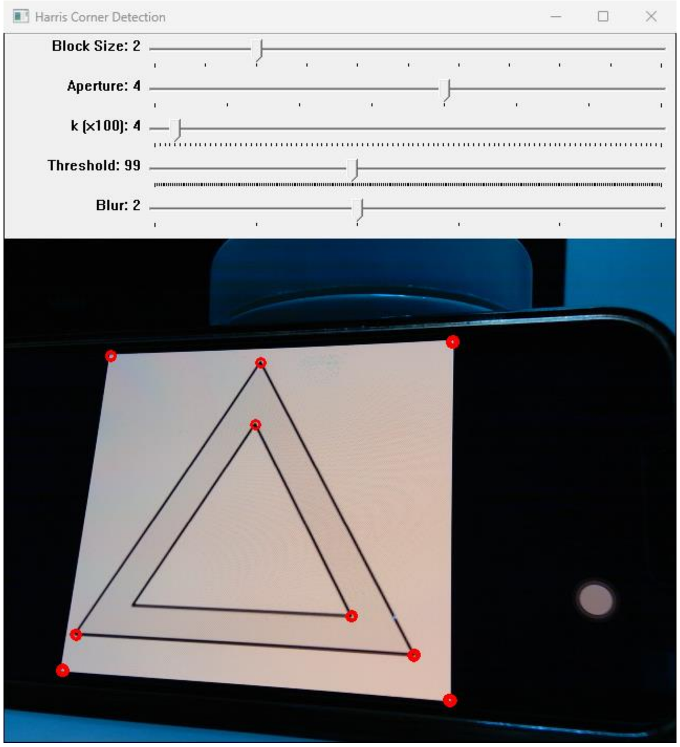




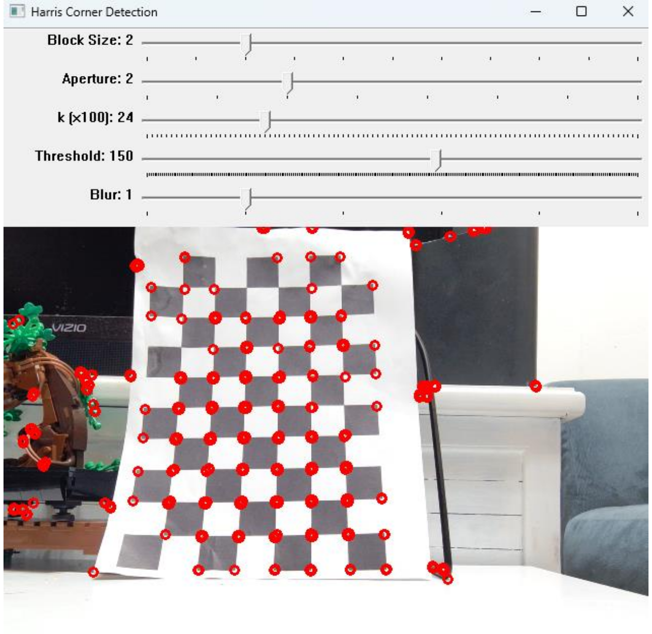
Task7 Detect Robust Features



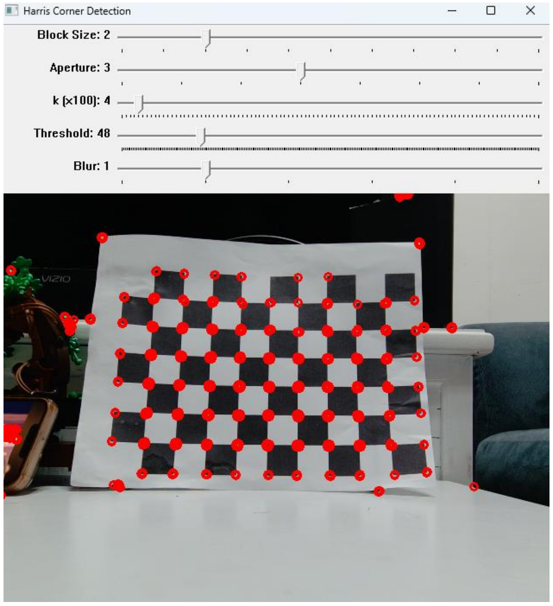
Block Size: 2
Aperture: 3
k (x100): 4
Threshold: 150
Blur: 1



Block Size: 2
Aperture: 4
k (x100): 4
Threshold: 99
Blur: 2



Block Size: 2
Aperture: 2
k (x100): 24
Threshold: 150
Blur: 1



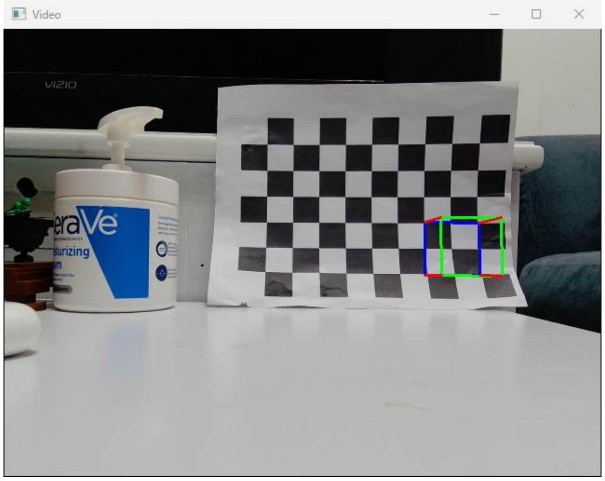
Block Size: 2
Aperture: 3
k (x100): 4
Threshold: 48
Blur: 1

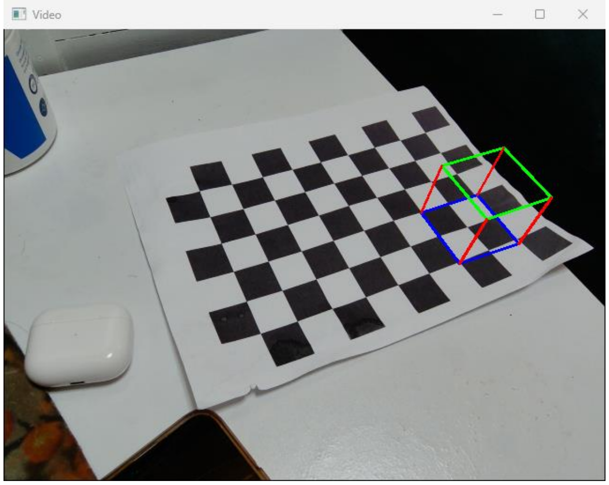
We might be able to use Harris corner as the basis of putting augmented reality into the image as:

- it could provide a feature point for tracking camera motion
- it could act as reference points to position and orient virtual objects in augmented reality

Extension

- The first extension is I created a dynamic trackbars on top of the video frame as seen in task 7 photos, so users could alter the parameters as need be.
- In task 6, I also added a cube rather than a triangle





Reference:

https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html#ga93efa9b0aa890de240ca32b11253dd4a

https://stackoverflow.com/questions/33503138/how-to-extract-video-frames-and-save-them-as-images-using-c

https://docs.opencv.org/4.x/dd/d1a/group__imgproc__feature.html#ga354e0d7c86d0d9da75de9b9701a9a87e

https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html#ga6a10b0bb120c4907e5eabbcd22319022

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga549c2075fac14829ff4a58bc931c033d

https://www.youtube.com/watch?v=bs81DNsMrnM

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga1019495a2c8d1743ed5cc23fa0daff8c

https://stackoverflow.com/questions/53277597/fundamental-understanding-of-tvecs-rvecs-in-opencv-aruco

https://www.reddit.com/r/computervision/comments/6wkb1x/opencv_solvepnp_function_outputs_rvec_tvec/#:~:text=Tvec%20is%20the%20translation%20vector,inverse%20transform%20of%20tvec%2C%20rvec

https://www.youtube.com/watch?v=Z_HwkG90Yvw&t=1s

https://comsci.blog/posts/intuitive-harris