CSS5330 Pattern Recognition and Computer Vision

Project 4: Calibration and Augmented Reality

Francis Jacob Kalliath

1. A short description of the overall project

The Calibration and Augmented Reality project helps us understand the various stages of camera calibration which includes finding the position or the camera i.e the rotation and translation of the target image. Then overlaying a virtual step graph on the target image.
This project was done on two different types of target image which are Chessboard and ArUco image. The extension also includes implementation of overlay of an object on the target image which changes its orientation when the pose of the target image is modified.

Video for Calibration and projection of 3D Object:
https://drive.google.com/file/d/1QG-dgm_ieBDKlofQMkq4Mn1JBMLVMvCS/view?usp=share_link

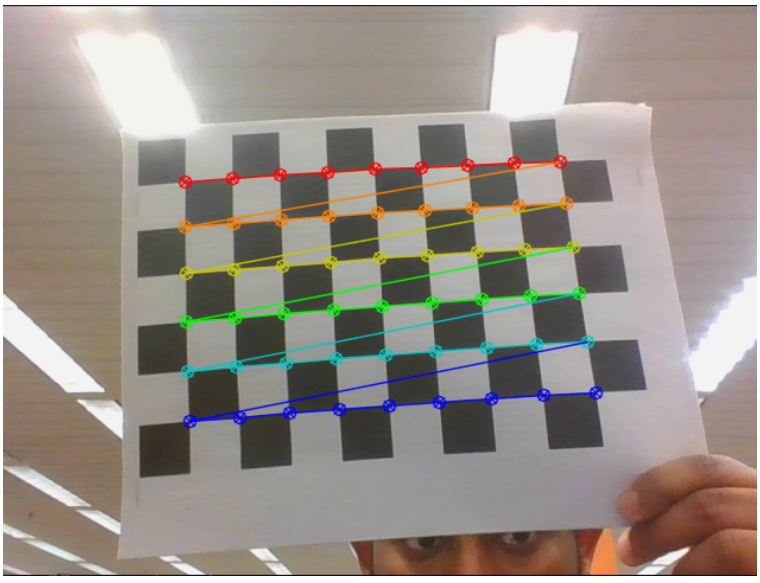
Video for Harris Corner detection:
https://drive.google.com/file/d/1KCk1bdZKkuAciwATj-c-jYgGON4UOcE1/view?usp=share_link

2. Images collected

TASKS:

Task 1

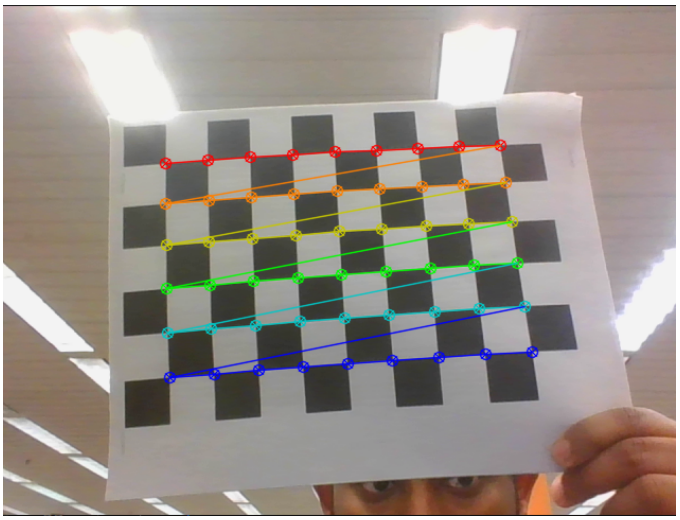
Detect and Extract Chessboard Corners



Task 2

Select Calibration Images

The chessboard corners are highlighted in the below image



Task 3

Calibrate the Camera

The camera matrix, distortion coefficients and re-projection error are displayed in the terminal.

```
calibrate for camera done
Matrix for Chessboard:
1048.18, 0, 79.3298,
0, 1016.28, 208.488,
0, 0, 1,
Distortion Coefficients for the Chessboard:
1.28381, -44.8459, -0.0305689, -0.0271895, 240.218,
Re-projection Error for Chessboard: 0.472246
```

Task 4

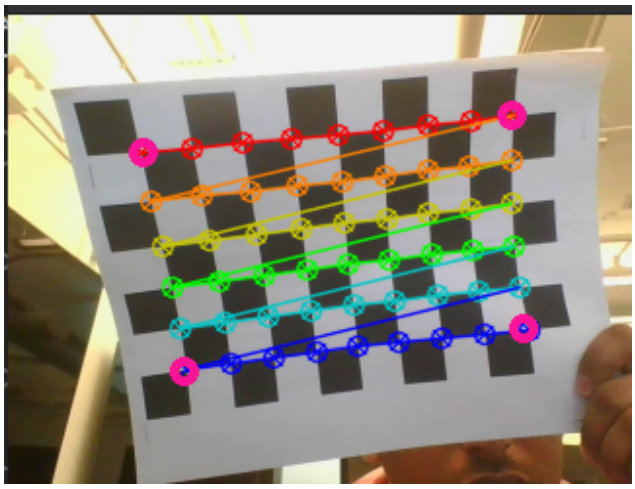
Calculate Current Position of the Camera

The rotation and translation are calculated in realtime and updated so that the projects can be drawn with the updated values.

```
rotationVector[-2.748103365372276;
 0.07953002257713972;
 0.08492162142529096]
translationVector[-0.7013313922688962;
 -3.068086060700905;
 22.54659578091854]
rotationVector[-2.747708373083724;
 0.08296843498479119;
 0.08542987568765074]
translationVector[-0.7138642334805804;
 -3.040499778265611;
 22.55293172404601]
```

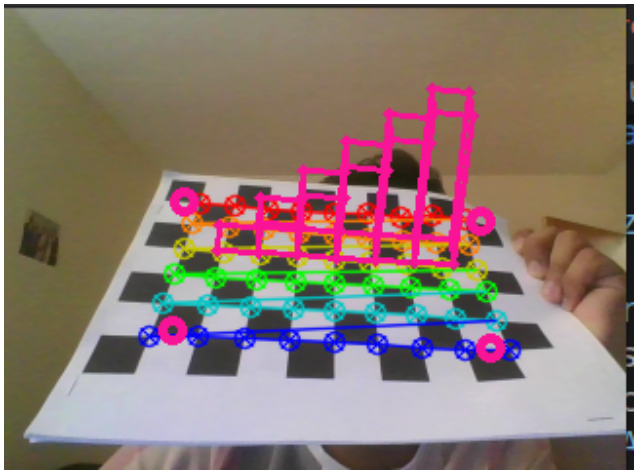
Task 5
Project Outside Corners or 3D Axes

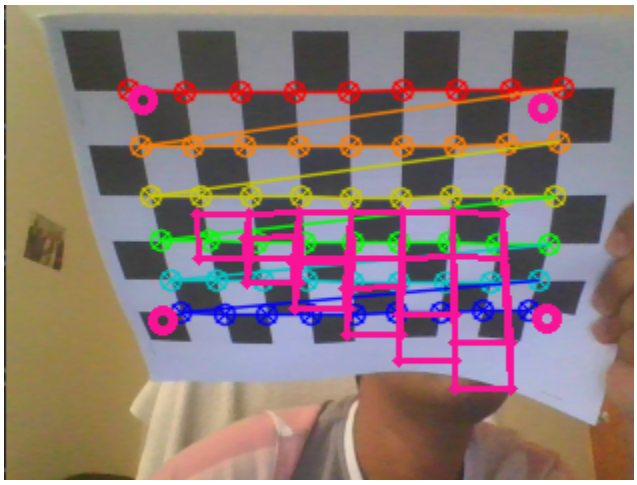
The corners of the chessboard are identified and a circle is drawn around it.



Task 6
Create a Virtual Object

The corners are taken as a reference, translation and rotation values are also used and then with all the information the projection is drawn on the target image which is a chessboard.

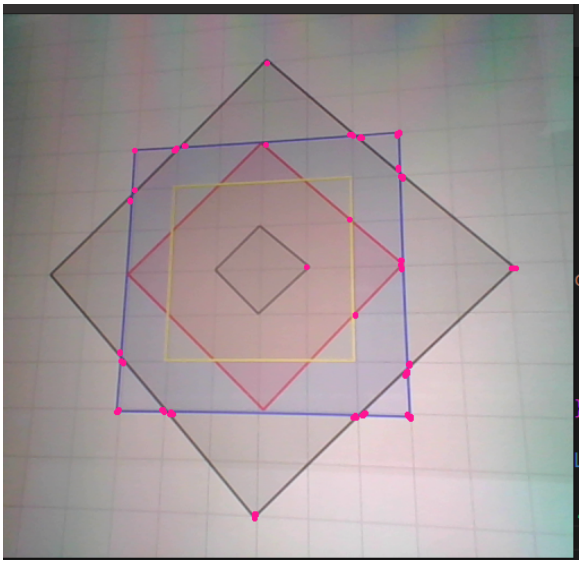


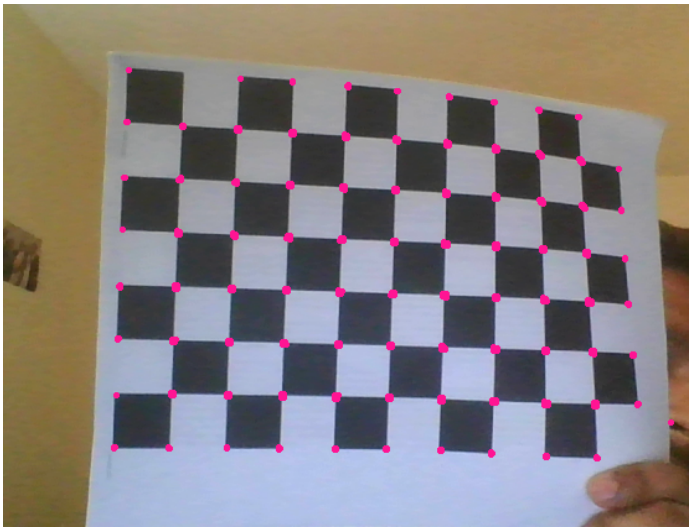


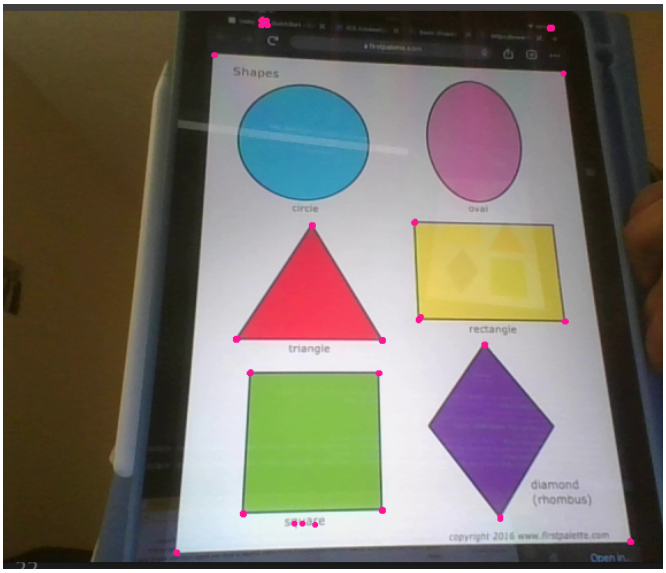
Task 7

Detect Robust Features

The Harris Corner Detector algorithm is used to detect feature points in images. Firstly we have to establish a correlation between the 2D coordinates of the points in the image and their corresponding 3D coordinates in the real world through camera calibration. Camera calibration involves estimating the intrinsic and extrinsic parameters of the camera, including the camera's position and orientation relative to the scene. Once the camera is calibrated, the 3D points can be projected and used to overlay virtual objects onto the real world scene.
After testing we understand that HarrisCorners has a higher capability of detecting corners, even in a complex image. Therefore, it could potentially be utilized for more intricate feature detection leading to better accuracy in projecting of objects.

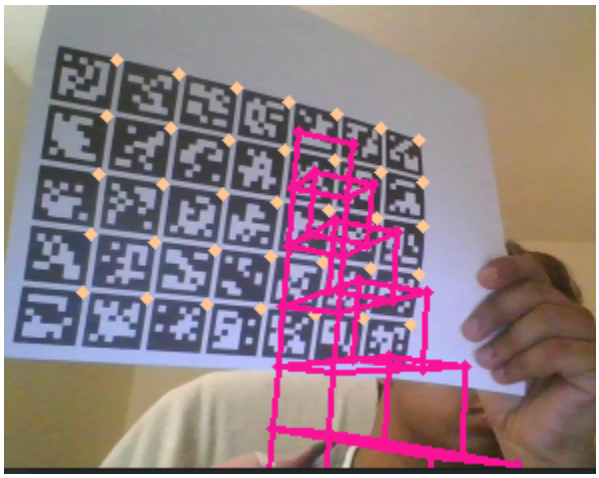


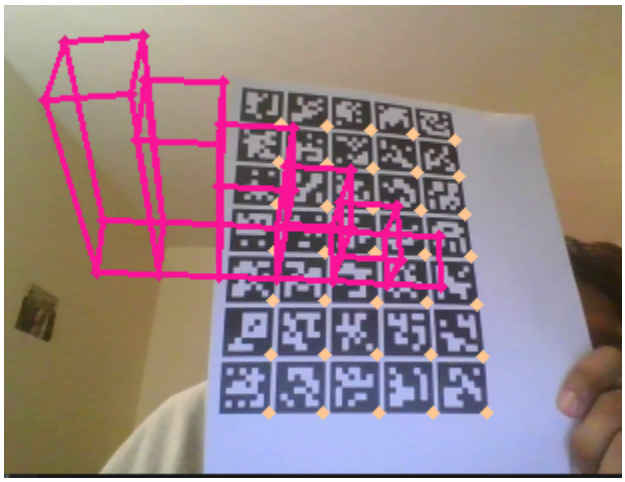



Shapes
circle
oval
triangle
rectangle
square
diamond
(rhombus)
copyright 2016 www.firstpalette.com

Extension 1

I have implemented a projection of the step graph on a target image as ArUco. To implement this I had to detect the features on the ArUco then calibrate it with the captured images. The rotation and translation of the target image is calculated and then the projection is overlaid on the target image.

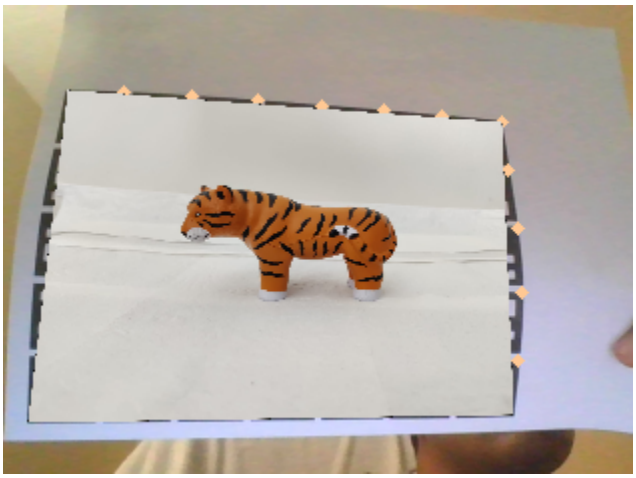




```
Aruco Matrix:
363.74, 0, 168.875,
0, 370.346, 186.449,
0, 0, 1,
Aruco Distortion Coefficients:
0.569457, -2.73143, 0.0538567, 0.00269727, 2.84564,
Aruco Re-projection Error: 0.859152
```

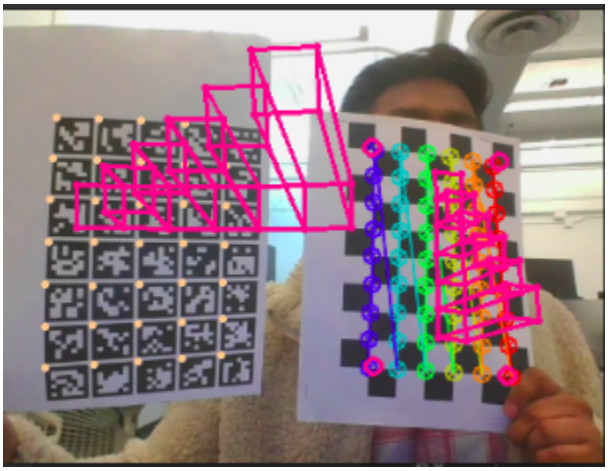
Extension 2

In the extension 2 we are able to overlay a picture on the ArUco target image. I have used findHomography() and then wrapPerspective() functions to find homography between the target and image and then wrap them together.



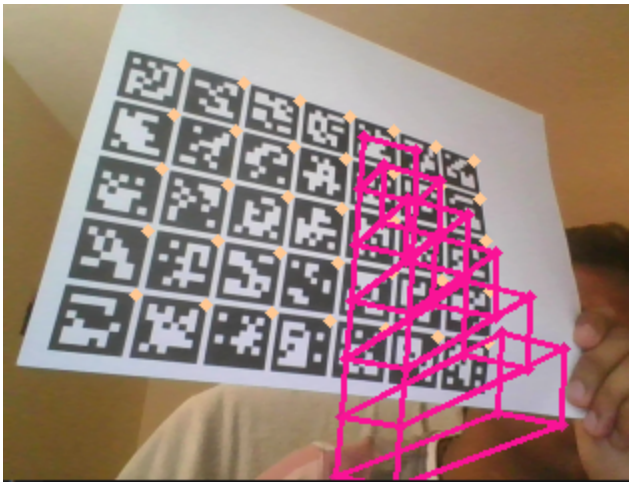
Extension 3

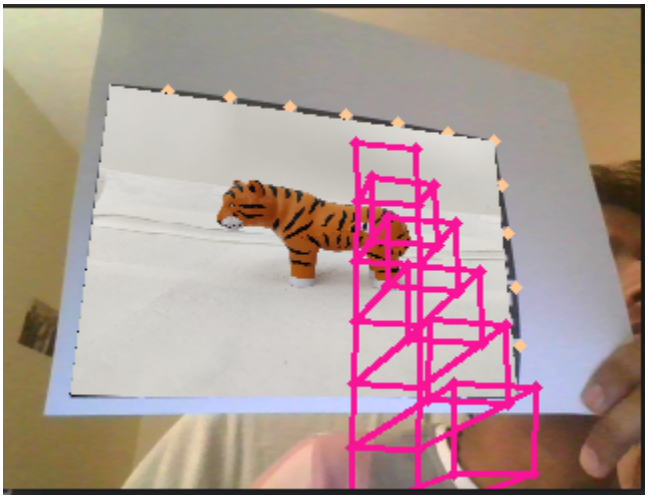
This project can simultaneously display the projections of both the target images of chessboard and ArUco



Extension 4

The extension can display the projections of the object on the overlaid image over the target image which is the ArUco image.





3. Reflection of the learnings

In this project I got hands-on exposure to Augmented Reality. Performing the task of calibrating the chessboard and the projecting the object on top of the chessboard give a from scratch step by step process of how the augmented reality projections are implemented on surfaces :
Some of the most important learning from this project are:

1) In this project I implemented a chessboard as the target image and then projected 3 dimensional objects on the chessboard. Here I understood the basic methodology of calibration and projecting objects.
2) In the next part of the project I implemented the projections on Aruco. Aruco was kept as the target image and then projections were done on the Aruco. This gave us an idea on how to implement projections on different kinds of surfaces.
3) In the last part of the project we implemented the Harris corner feature detector where the Harris corners show up on highlights on the target image.

4. Acknowledgement of the material

- HackerRank
- OpenCV Documentation / Tutorials
- Stackoverflow
- GeeksforGeeks
- W3 School
- Git
- Quora
- Gormanalysis