CS5330 - Pattern Recognition and Computer Vision
(sec-01 spring 2023)

Northeastern University

Project - 4

Sreehari Premkumar

Description of the Project

Project 4 of Pattern Recognition and Computer Vision, has the usage of OpenCV along with C++ to implement an Augmented Reality Camera that will project any object into the video stream of the world. The project requires us to calibrate the camera to record intrinsic and parameter of the camera. After the calibration, we use the calibration details and camera pose to help in finding the perspective of the checkerboard in a new image. Then the perspective information was used to project an 3d(made up) object to the 2d image keeping the perspective in consideration to give an effect of the actual object being in the real world, when watching the video stream. Usage of opencv libraries to calibrate and projection and similar other libraries was

allowed. Overall, the project provides an excellent starting point for understanding camera calibration and Augmented Reality.

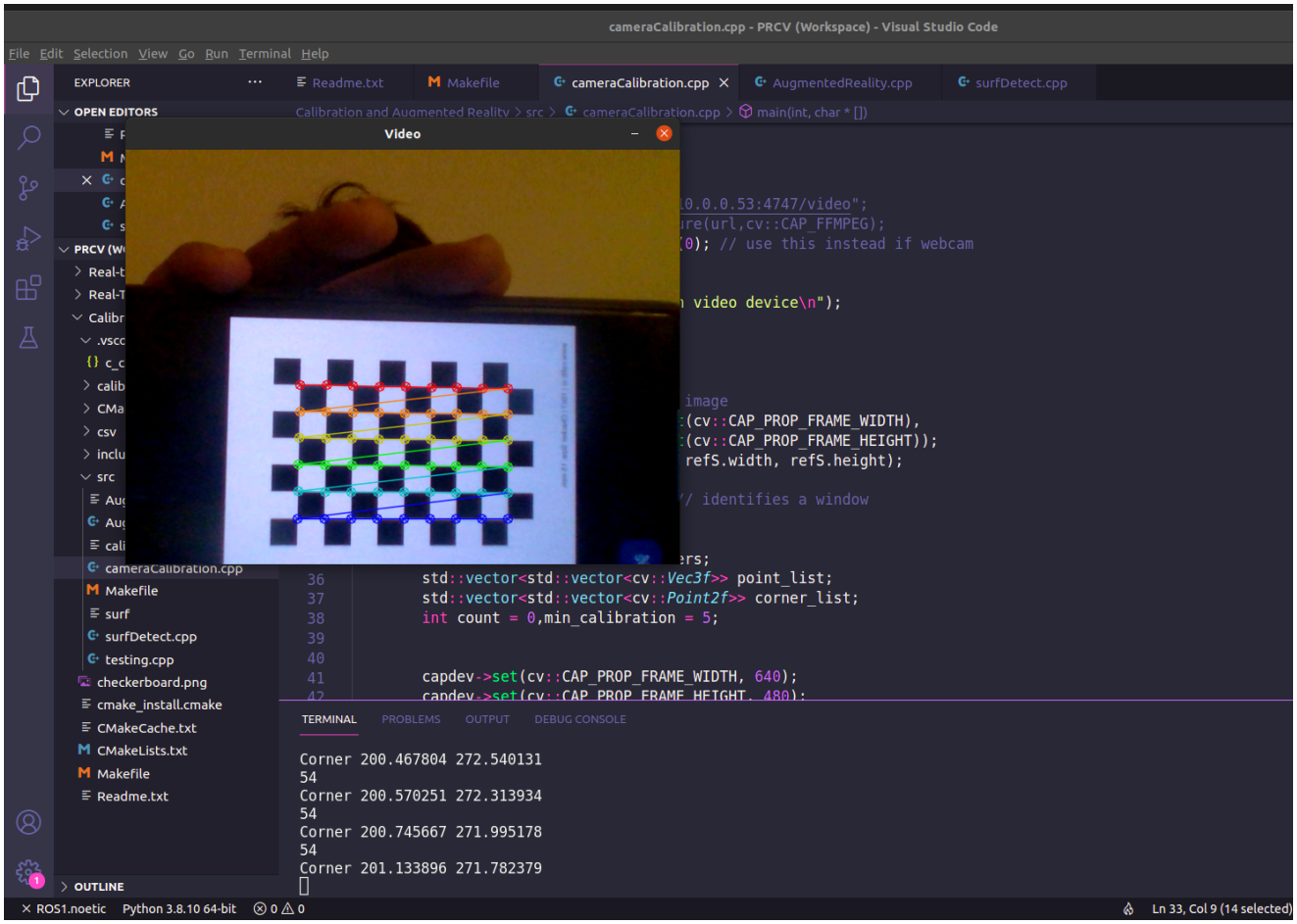
TASK - 1 - Detect and Extract Chessboard Corners

Input: The Input was a video streamed from the laptop webcam.

Can press ‘q’ to quit the program

Detected Corners :

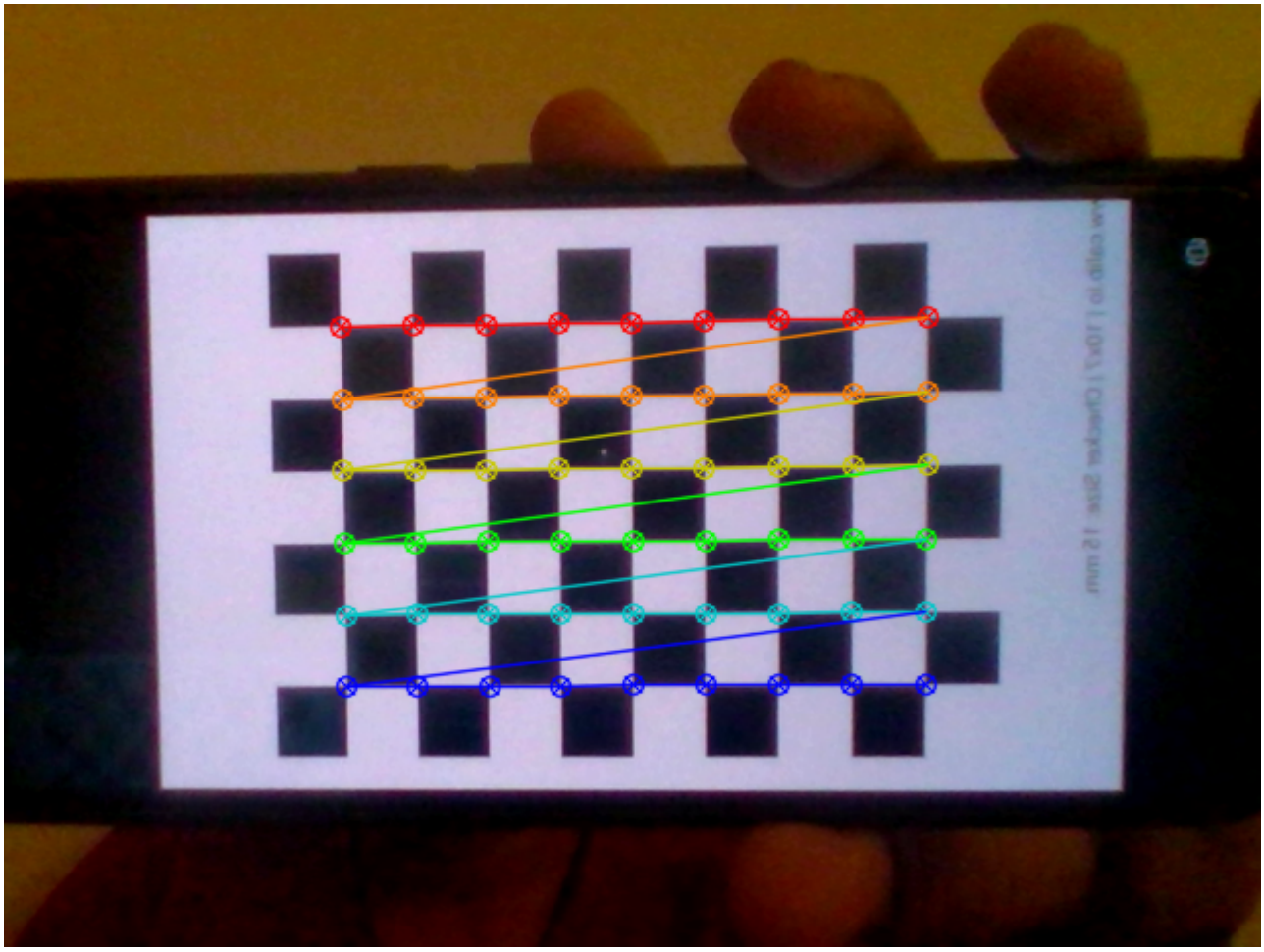
The detected corners are displayed, also the number of corners and location of the 1st corner is being printed, for verification and testing purposes.



Task 2 - Select Calibration Images

When a good calibration setup is achieved in the press ‘s’ to save the current setup as a calibration image. The image at that instance would be taken for calibration and the corresponding corner points and world coordinates of corners will be saved. The images will also be saved under the calibration folder.

One of the image saved for calibration :



Task 3 - Calibrate the Camera

When more than 5 images are saved for calibration, the system will automatically calibrate the camera every time a new calibration image is captured. The reprojection error is also calculated. The better the initial quality of the image the better the initial reprojection error.
Specifically for my system the error started at 0.3523 at 5 images, and went down to 0.1687 around 12 images. And finally to 0.1370 at 18 images.

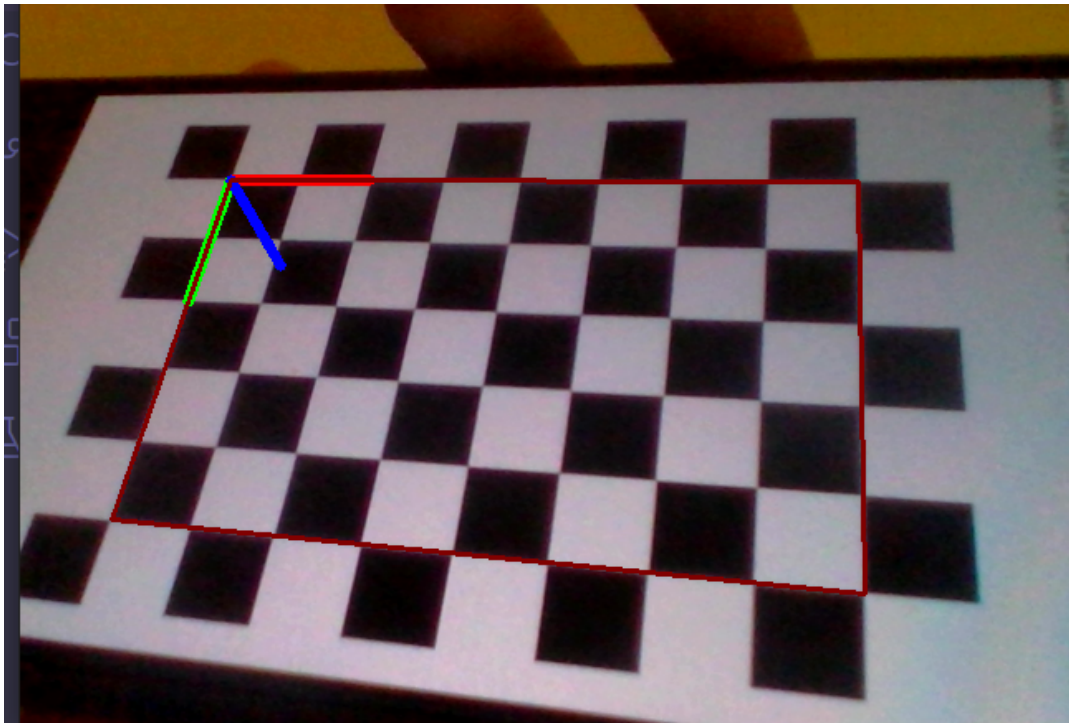
Then using the corner points and corresponding world coordinates, the intrinsic parameters of the camera is calculated. And finally the parameters along with distortion coefficients are saved in a yaml file named camera_parameters.yaml under the calibration folder.

Task 4 - Calculate Current Position of the Camera

In another program, again the checkerboard is detected. Then the previously saved camera_parameters is used to calculate the current position (rotation and translation) of the board, with respect to the world coordinate frame in the current image using solvePNP function provided by openCV.

Task 5 - Project Outside Corners or 3D Axes

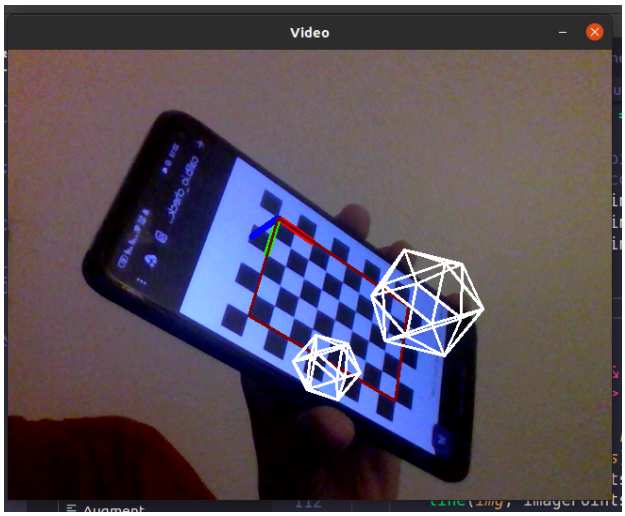
Output 3D Axis(R-X | G-Y | B-Z) + Bounding box of 4 Corners (Maroon):

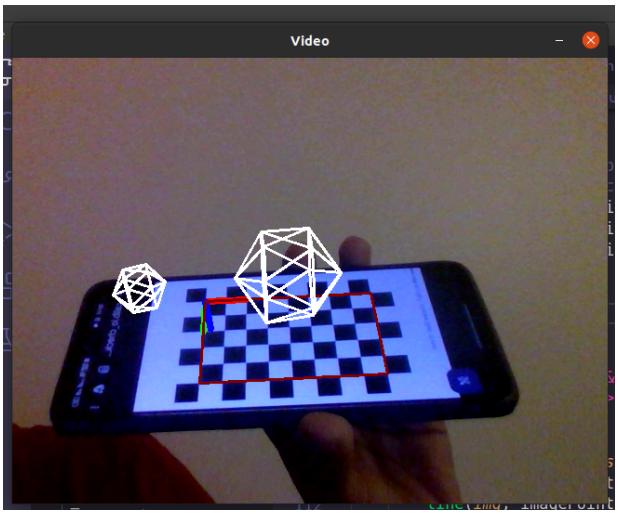


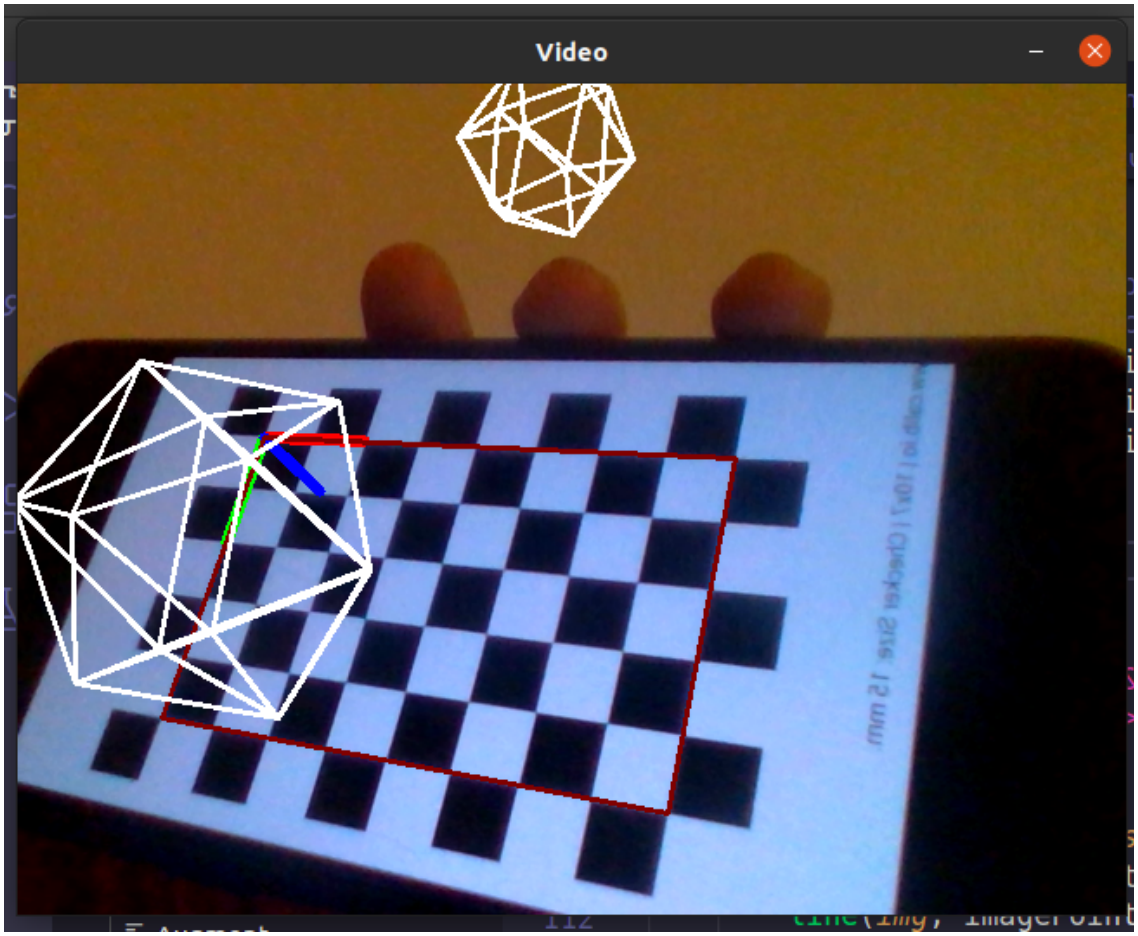
Task 6 - Create a Virtual Object

The object I chose was an icosahedron. Then I used 2 different sizes of the same. Then I updated the rotation value of all the points in the icosahedron about origin to show rotation. Then translated one first, then updated rotation about the origin to give a revolution effect.Finally after some adjustments with size and speed of rotation i was able to imitate the Earth and the Moon.

Output : (Images) (Video is in link at the end)



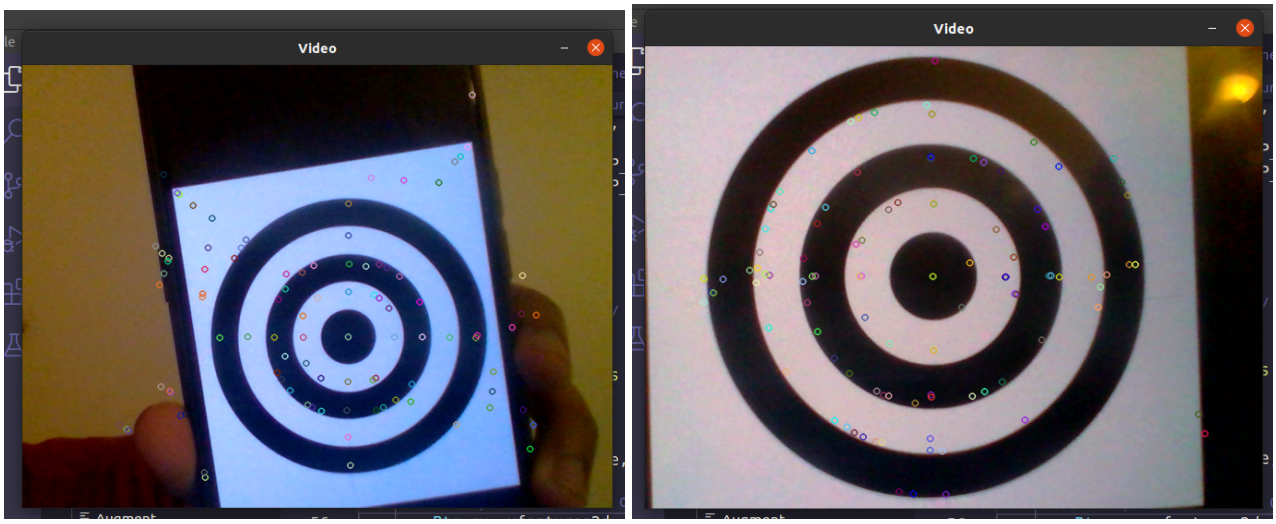




Task 7 - Detect Robust Features

In another program SURF feature algorithm in openCV was used to detect the features in a live video.

Output :



We can use the detected points and descriptor of each point as feature points. Take a pattern like the above bullseye, then use SURF to detect feature points and their descriptors and save them as a dataset, and assign a specific AR object with certain size, rotation and translation to those point sets. Then we take a new image, find a bullseye using the feature points and the descriptors and project the AR object according to the feature points.

Video Link

https://drive.google.com/file/d/1rJyELwbHYpchKoB17jVj5x3f6GR_iGSd/view?usp=sharing

Reflection

I have learned about camera calibration and making Augmented Reality. I used to think of it as some kind of rocket science but it was a lot easier than I expected. Detecting the checkerboard pattern, then finding the corner points, and building our own world frame with respect to the corner points, and then projecting an object from a different perspective; everything was made easy with the help of openCV. My next goal would be using a 3d model and projecting it onto a video stream to create some fun filters. This project gave me a lot of insights and ideas.

Acknowledgement

Calib.io website for 10x7 checker board, since the version provided was not showing up properly.

Referenced Stack overflow to help me with debugging the code.

Referred OpenCV official website to refer how to use library functions.