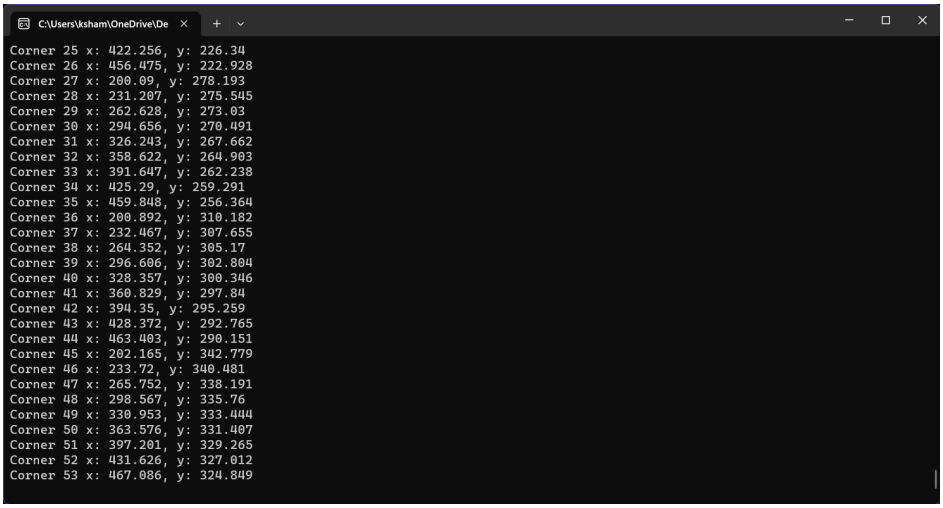
Project 4: Calibration and Augmented Reality

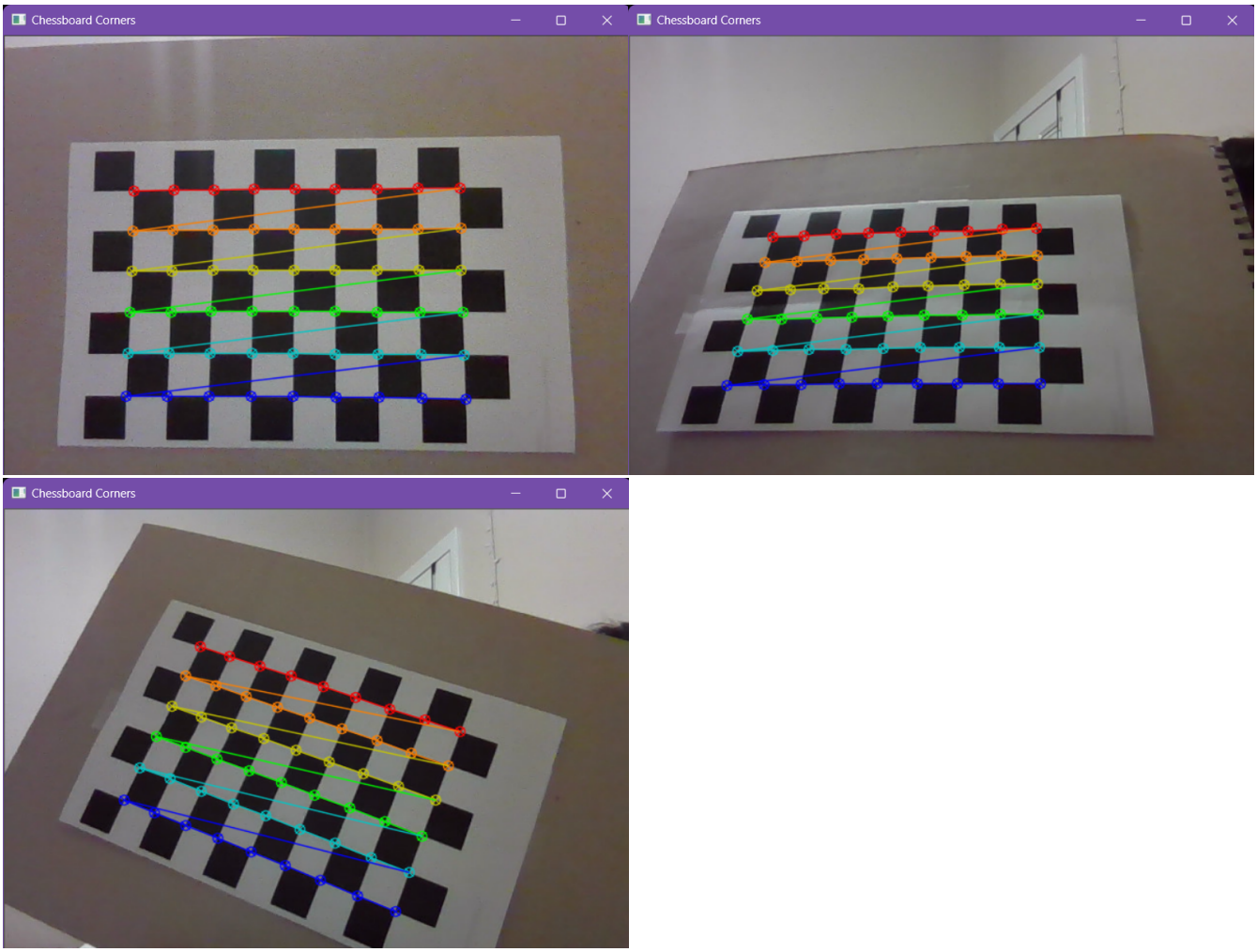
Overview

The purpose of this project was to develop an augmented reality system by adding a virtual object to the scene, learn about camera calibration, and put it into practice. The chessboard was the target for the the above purposes, and the following OpenCV functions were used: findChessboardCorners(), cornerSubPix(), drawChessboardCorners(), calibrateCamera(), solvePNP() , and projectPoints() to determine the corners, camera calibration, for camera pose, and to get 3D projection of the image points. In an effort to comprehend feature detection and extraction better, we also incorporated Harris Corner detection.

Task 1: Detect and Extract Chessboard Corners

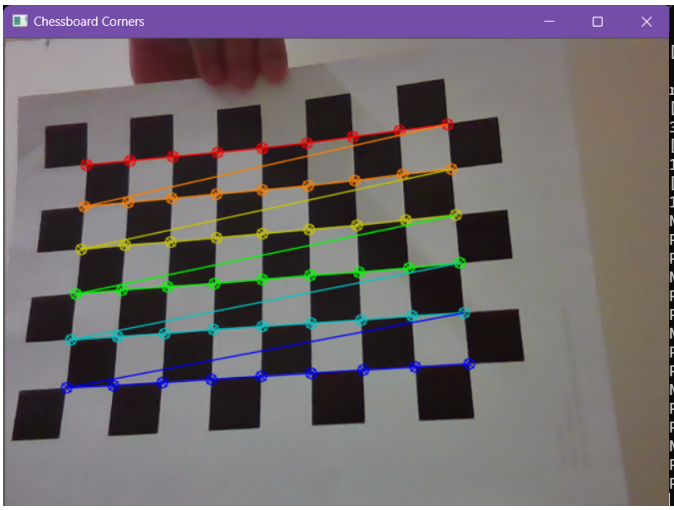
For a 9 x 6 checkerboard, the number of corners were discovered using findChessboardCorners; precise locations were then discovered using cornersubpix. The first corner and the number of discovered corners are shown below. When it locates a chessboard in the live stream, it draws the corners of the board. We are also finding the x-y coordinated of the detected corners.





Task 2. Select Calibration Images

On pressing the letter "s," the we save the calibration images together with the associated image coordinates. The chessboard corners are emphasized in the following calibration image, which was saved. The function drawChessboardCorners() was used to achieve this.




```
C:\Users\ksham\OneDrive\De
[ INFO:0] global C:\Users\Administrator\Desktop\OpenCV-vc16\opencv-
 (197) cv::`anonymous-namespace'::VideoBackendRegistry::VideoBacken
rity): FFMPEG(1000); GSTREAMER(990); INTEL_MFX(980); MSMF(970); DSH
[ INFO:0] global C:\Users\Administrator\Desktop\OpenCV-vc16\opencv-
396) cv::impl::getPluginCandidates Found 2 plugin(s) for GSTREAMER
[ INFO:0] global C:\Users\Administrator\Desktop\OpenCV-vc16\opencv-
175) cv::impl::DynamicLib::libraryLoad load C:\OpenCV\x64\vc16\bin\
[ INFO:0] global C:\Users\Administrator\Desktop\OpenCV-vc16\opencv-
175) cv::impl::DynamicLib::libraryLoad load opencv_videoio_gstreame
Number of corners: 54
First corner: 98.5516, 153.884
Frame saved.!
Number of corners: 54
First corner: 99.3454, 154.263
Frame saved.!
Number of corners: 54
First corner: 110.284, 158.19
Frame saved.!
Number of corners: 54
First corner: 99.1123, 146.495
Frame saved.!
Number of corners: 54
First corner: 99.7334, 145.874
Frame saved.!
```

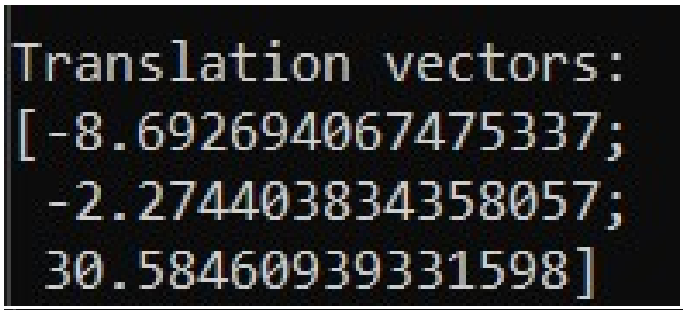
Calib_frame1	3/11/2023 6:33 PM	JPG File	46 KB
Calib_frame2	3/11/2023 6:33 PM	JPG File	45 KB
Calib_frame3	3/11/2023 6:38 PM	JPG File	44 KB
Calib_frame4	3/11/2023 6:38 PM	JPG File	44 KB
Calib_frame5	3/11/2023 6:38 PM	JPG File	44 KB
Calib_frame6	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame7	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame8	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame9	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame10	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame11	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame12	3/11/2023 6:45 PM	JPG File	44 KB
Calib_frame13	3/11/2023 6:45 PM	JPG File	44 KB

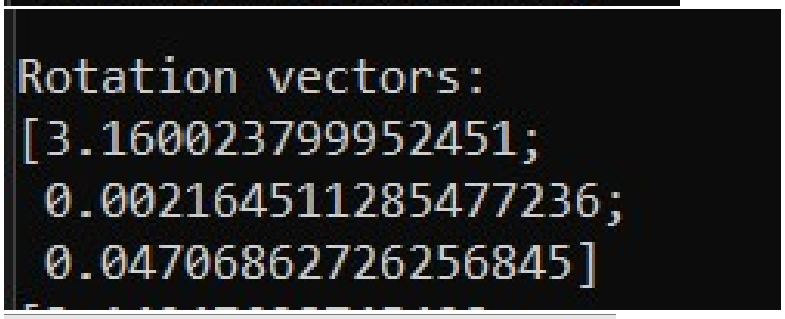
Task 3. Calibrate the Camera

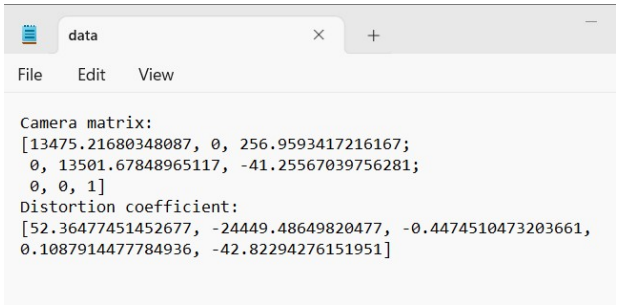
The code calibrates the camera on keypress "c" using the data stored via the calibration photos in the previous task. The re-projection error over 6 calibration pictures was 1.32748 . The camera matrix, distortion factors, and resulting Rotation and Translation vectors are displayed in the images below. The calibrateCamera() function was used for this. The camera_matrix and distortion_coefficianets are being store in a .txt file.

```
Camera_Matrix: [1436.981250534648, 0, 472.7364996486393;
 0, 1436.813128316594, 203.4803076151648;
 0, 0, 1]
```

```
Distortion_coefficient: [-3.264043921565127, 31.17366184266363, 0.01918118305919549, -0.08990874303440712, -145.6969000554476]
re-projection error: 1.32748
```

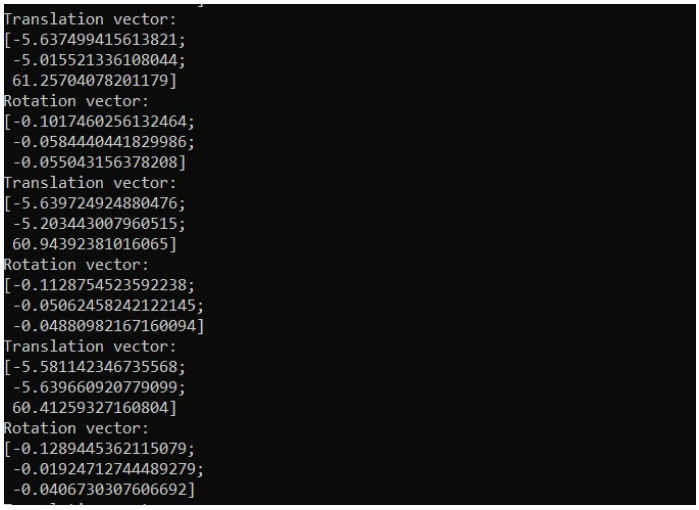






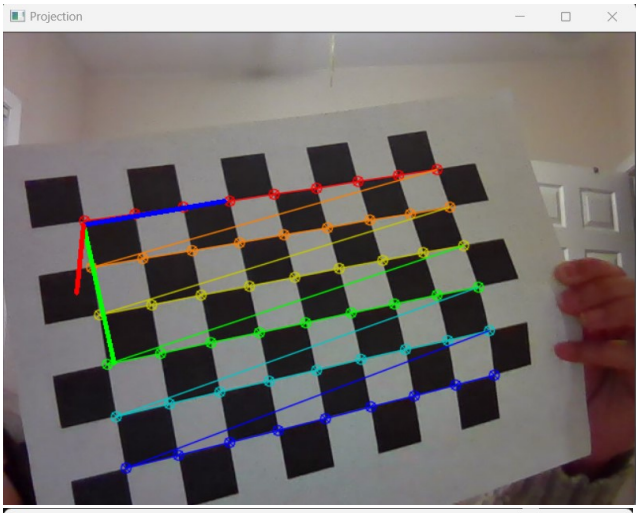
Task 4. Calculate Current Position of the Camera

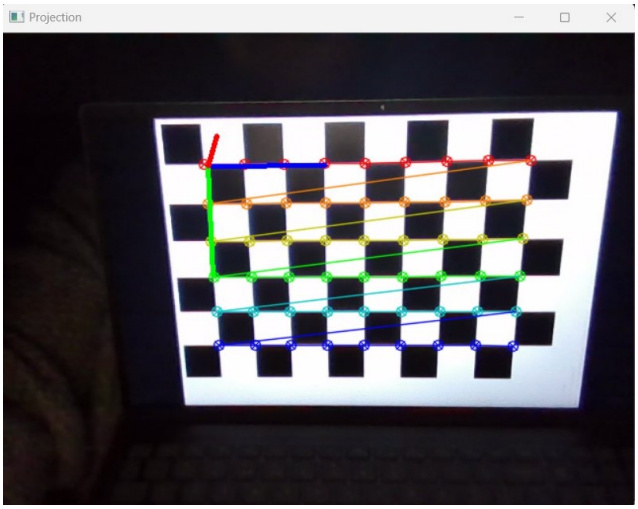
This task included figuring out the camera's pose in real-time using the previously determined camera calibration parameters. Use of the solvePNP() function was made for this. The application prints out the rotation and translation matrix related to each frame in real-time as seen in the accompanying image.

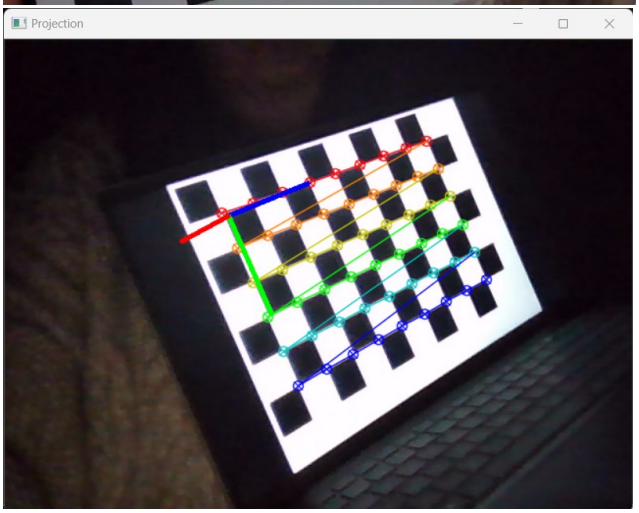


Task 5. Project Outside Corners or 3D Axes

The 3D axes that were projected from the chessboard's corner can be seen in the image below. The picture coordinates provided as arguments were used to find the 3D world points using the projectPoints() function.



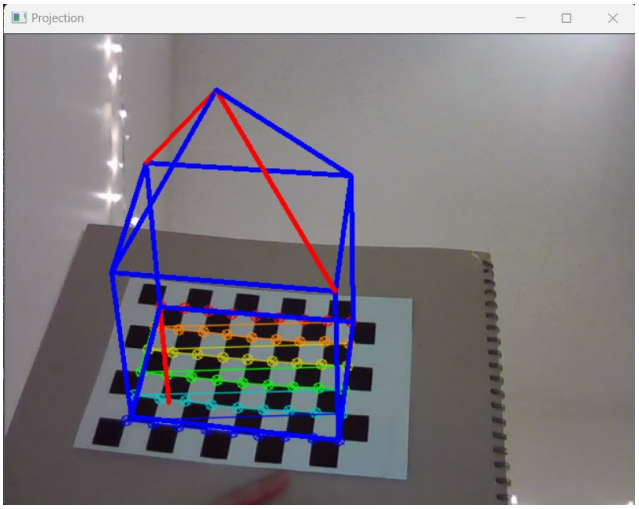




Task 6. Create a Virtual Object

In order to complete this task, we had to design a virtual object comprised of lines and project it onto the chessboard so that it could move about while maintaining its orientation. A screenshot of the virtual object is shown in the image, and a link displays a real-time video of the system.

Shape_projection_video



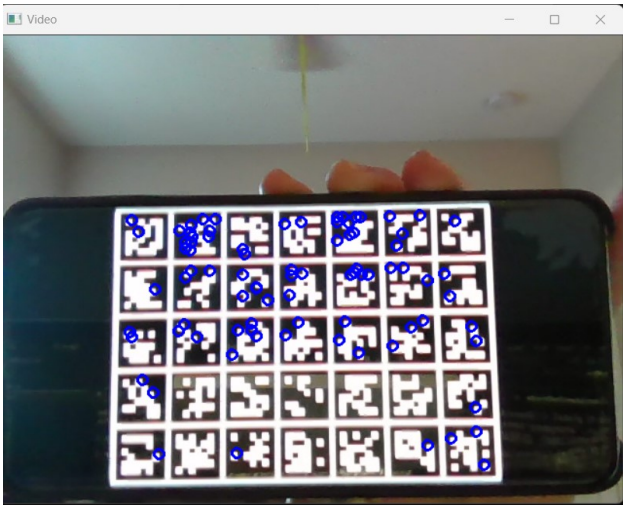
Task 7. Detect Robust Features

The chessboard and AruCo markers are used in the to demonstrate Harris corners detection using the detectHarris() function.

The Harris Corners feature, which indicates where the features are in a video stream, was used for the very final task. It appears that Harris Corner gives us more corner detection options and lets us use point locations to connect to global coordinates. Overall, I believe the modeling technique is quite traditional. It identifies a few unique locations to match the edge segments, classifies the image's structure, and represents its local properties.

Harris corners, is not particularly reliable or accurate, I would only use them to add augmented reality to still images rather than in real time.

Harris.mp4 **Harris_2.mp4**



Extensions:

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

I tested out using android device Samsung S10+ , used DroidCam for using the mobile phone camera instead of the webcam. The only code change is this line capdev = new cv::VideoCapture(1) . 1 phone camera 0 webcam.

The re-projection error is 0.407734 which is less that the webcam reprojection error i.e. 1.32748. This indicated better phone camera performance

```
Camera_Matrix: [590.9348981706075, 0, 340.4184082069932;
 0, 632.1408808949525, 248.2006287231645;
 0, 0, 1]
Distortion_coefficient: [0.5728290444699029, -2.235068719564988, -0.04786970847478635, 0.02664556708885635, 2.630682920237902]
re-projection error: 0.407734
```

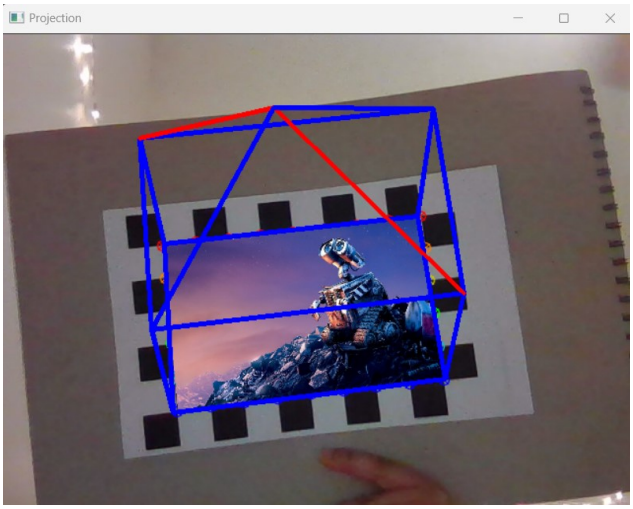
```
Translation vectors:
[-5.830020970533611;
 -2.032849738181914;
 17.08368490769297]
```

```
Rotation vectors:
[2.631736685459441;
 -0.1444697253128086;
 0.05669177740742847]
```

2. Changing the Target

We changed the target image here. The new target aligns well as the orientation changes. Below is the image and a video link of the same.

Target_change_video



Reflection

Reflecting on this project, I can confidently say that it was an amazing learning experience for me. The project provided me with a comprehensive understanding of the different components that make up augmented reality and how they work together to create the augmented experience. I was able to learn about important concepts such as camera calibration and pose calculation, which were previously foreign to me.

Although I found augmented reality to be complex initially, working on this project enabled me to develop a structured approach to tackling it. Additionally, I was able to put into practice the theoretical knowledge I had gained about combining OpenGL with OpenCV, which was both exciting and fulfilling.

Although I did not achieve my desired outcome due to time constraints, the project's potential for creative exploration left me intrigued and eager to continue exploring it. Moreover, the research helped me put into perspective the different coordinate systems used in augmented reality and how to convert them, which I found to be an essential aspect of this technology.

In conclusion, this project was a valuable learning experience that broadened my understanding of augmented reality and its various components. It was also an exciting creative outlet that has piqued my interest to continue exploring the possibilities of this technology.

Acknowledgements

I would like to express my gratitude to my TA, Mrudula Prasad, for her invaluable assistance in resolving some critical errors in my code. I would also like to acknowledge the numerous websites and online resources that I consulted during this project.

https://docs.opencv.org/4.x/d4/d94/tutorial_camera_calibration.html

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga93efa9b0aa890de240ca32b11253dd4a

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

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga6a10b0bb120c4907e5eabbcd22319022

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga3207604e4b1a1758aa66acb6ed5aa65d

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

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

https://docs.opencv.org/4.x/d9/d6d/tutorial_table_of_content_aruco.html

http://www.opengl-tutorial.org/beginners-tutorials/tutorial-7-model-loading/