Ishan Tripathi
Naveen Verma

Project 4: Calibration and Augmented Reality

Project Description

This project aims to calibrate a camera and then use the calibration to detect a target and then generate virtual objects in the scene relative to the target. This target object moves and orients itself when the camera is moved, or the target is moved.

The program works in four modes:

Normal Mode: The program by default runs in normal mode. Normal mode is to be used to perform camera calibration. It performs following activity in normal mode:

- Detect and extract chessboard corners using OpenCV functions.
- Draw the chessboard corners using OpenCV function.
- This mode will also display following on the window title:
    - Number of corners detected.
    - Number of frames used for calibrations till now.
    - Current RMS(re-projection error) error
- A user can **press 's' at a particular frame to use that frame for calibration**.
    - This will save the image that is used for the calibration as x.jpg along with the extrinsic values in x.txt . Here x is a numeric value corresponding to the number of times calibration is done. For example 10^{th} calibration will be stored as 9.jpg and 9.txt .
    - This will also update the RMS error and all the previous calibration results saved in previous txt files.
- A user can **press 'c' to save the intrinsic parameters of the camera in camera.txt**, this needs to be done after multiple times calibration has been done and error is less than half pixel. This camera.txt is used when the program is run in other modes.

Pose Mode: This mode is to use the camera calibration performed in Normal mode using camera.txt and then based on that detect the chessboard, grab the corners and using solvePNP get the Rotation and Translation of the board.

- To run Pose mode the program needs to run with 'pose' command line argument.
- This mode will display the Rotation and Translation on the window title in real time.
- It will display the outside corner of the chessboard onto image plane in real time.
- It will display 3D axes on the board attached to the origin.

Play Mode: This mode is to create virtual objects in 3D world space relative to the target. The program uses the camera calibration performed in Normal mode using the intrinsic values saved in camera.txt and then calculates the Rotation and Translation and then uses the projection function to project.

- To run program in play mode the program needs to run with 'play' command line argument.
- If the camera is able to detect the target, it will display "Have Fun!" on the video title. If it is unable to detect the target it will turn into grayscale and will display "No pattern found" on the video title.

- The play mode allows user to create multiple 3D objects relative to the target, following are the objects which can be created when the key is pressed in play mode:
    - **Key ‘s’ will create a sphere.**
    - **Key ‘c’ will create a cylinder.**
    - **Key ‘o’ will create a Cone.**
    - **Key ‘u’ will create a Cube.**
    - **Key ‘v’ will display the axes on the target.**
- There are a lot of other fun features in the play mode which are mentioned in the Extension Section.

Harris Mode: The program can be run in Harris mode by running program with ‘harris’ command line argument. In this mode the program will detect Harris corners in the frame and show it in the video stream.

Ishan Tripathi
Naveen Verma

Images

Image 1: Chessboard corners along with Number of corners detected, Number of frames used for calibration till now and current RMS error on the window title.

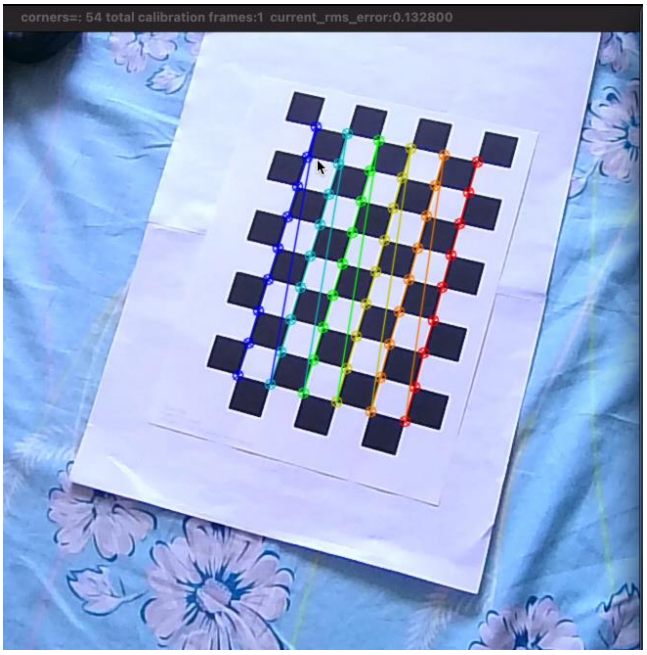
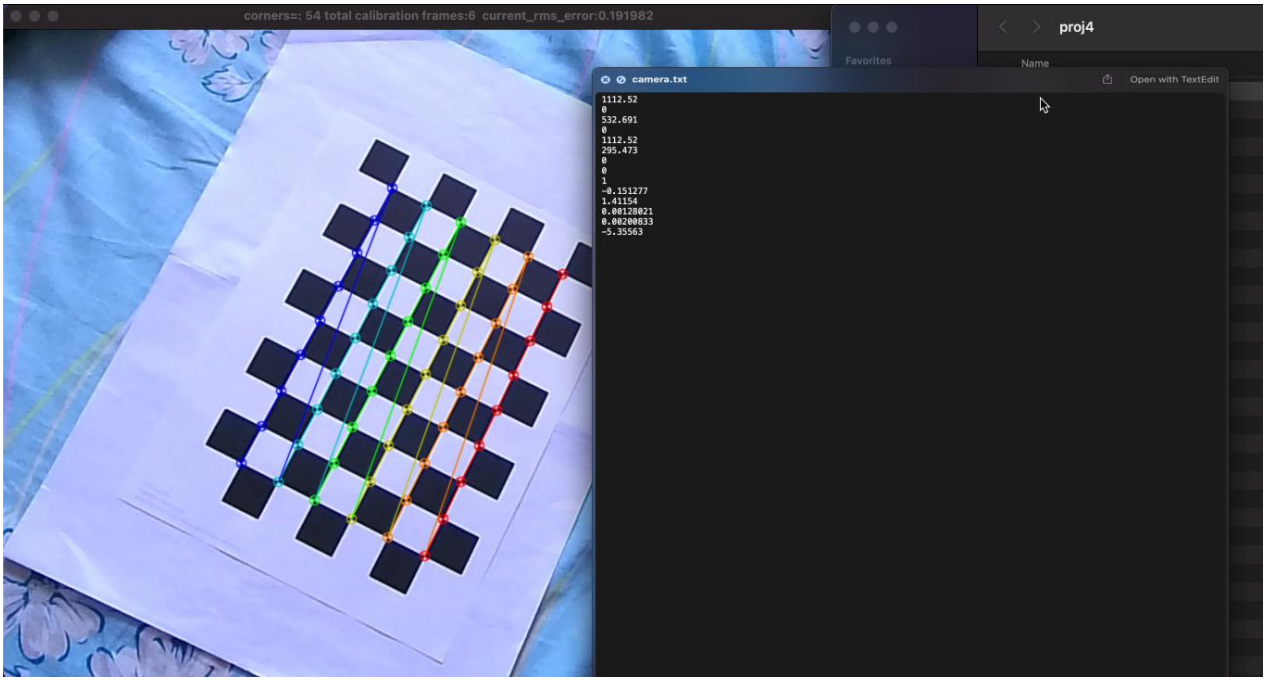
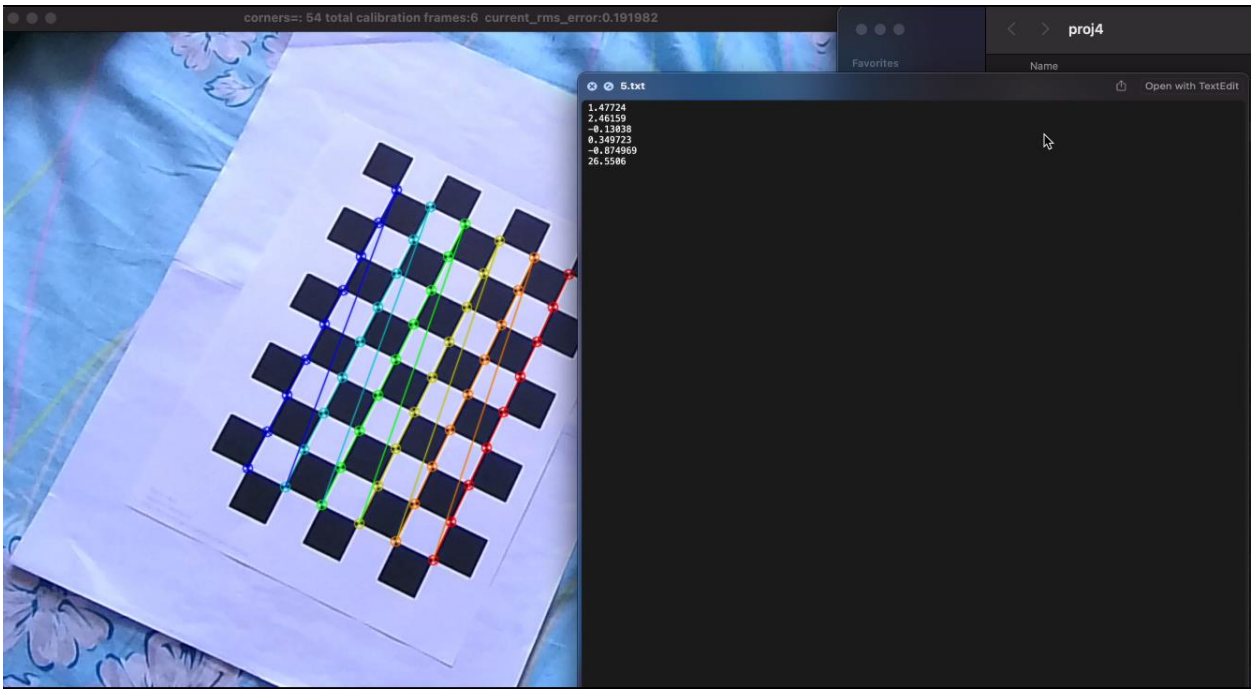


Image 2: Calibration Files and Error estimate (***Include the error estimate in your report.**)



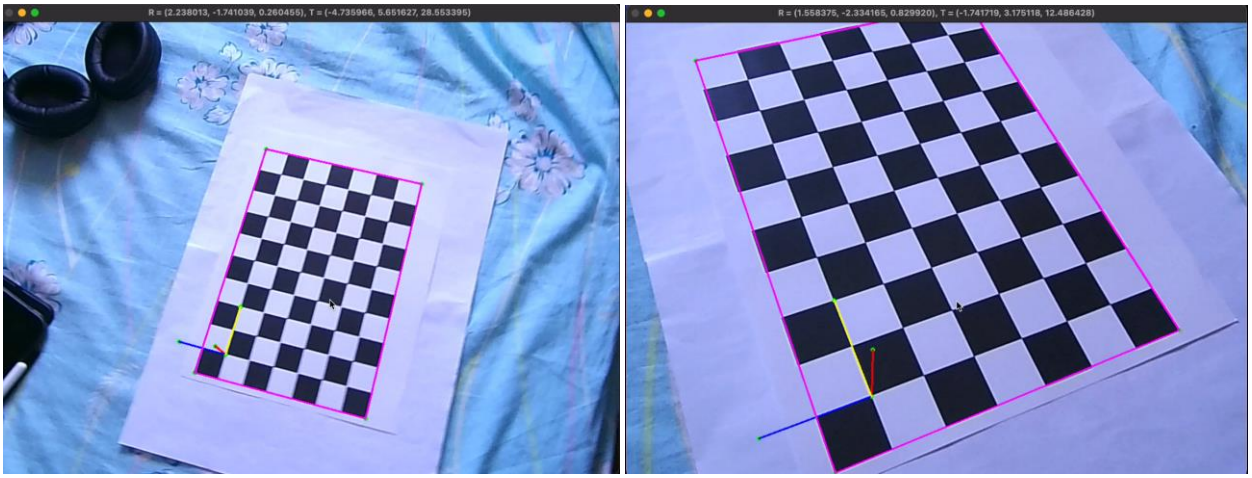
Camera.txt is created on pressing ‘c’ and it contains the intrinsic parameters of the camera.



5.txt is created on pressing 's' the 6th time while calibrating and this contains the extrinsic parameters.

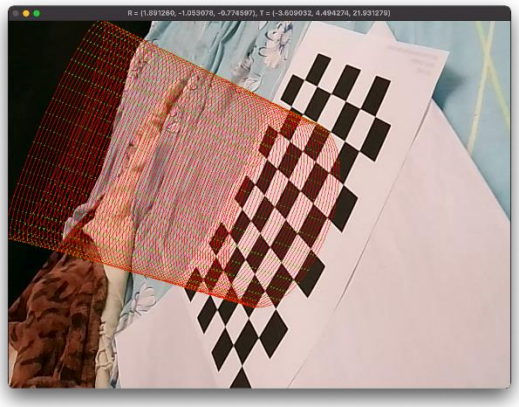
Error Estimate: 0.191982
Number of Frames used for calibration = 6

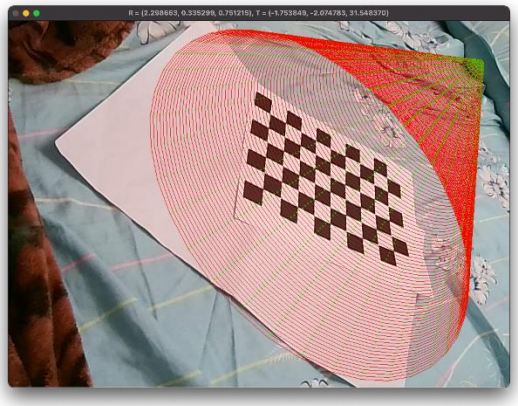
Image 3: (pose mode) Outside corner of the chessboard onto image plane in real time along with 3D axes on the board attached to the origin. Rotation and Translation data in real time can be seen on the window title.

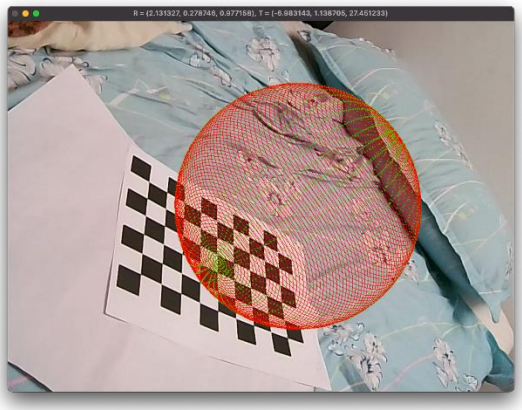


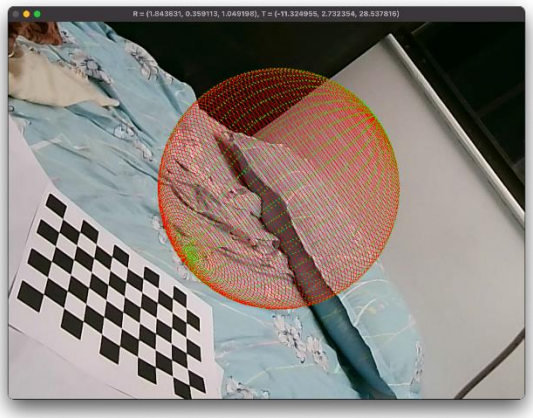
Ishan Tripathi
Naveen Verma

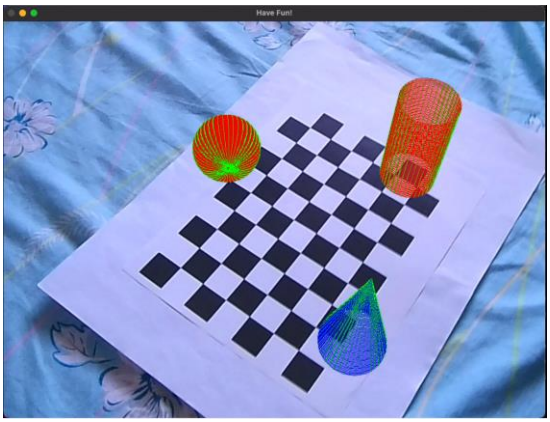
Image 4: Virtual objects

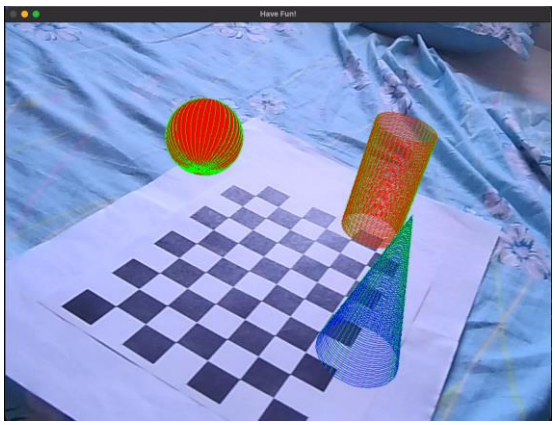






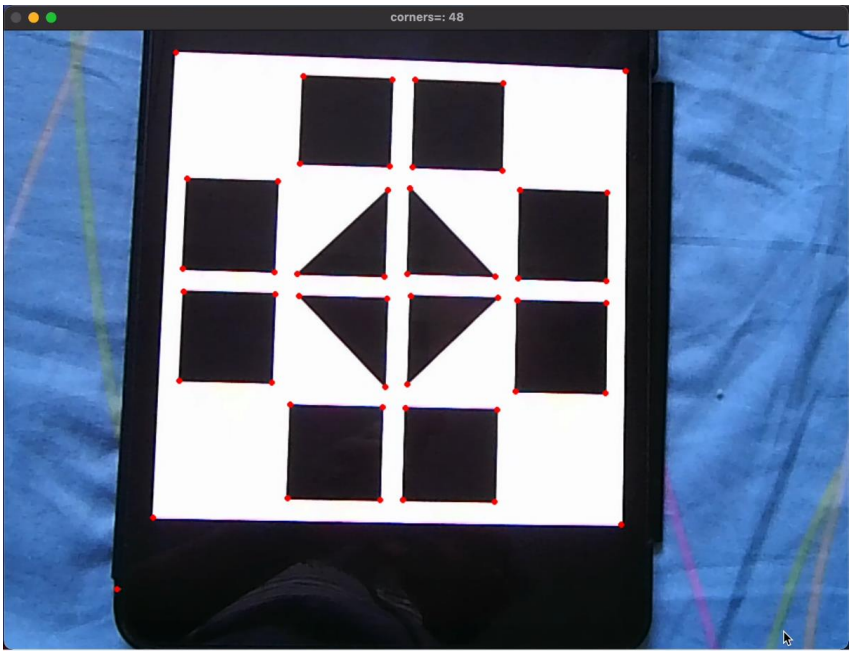






Ishan Tripathi
Naveen Verma

Image 5: Harris Corners



Video Links:

Camera calibration: https://drive.google.com/file/d/1am3iPhiCtGuVTav-yDBtYCPDx3zwdEkn/view?usp=sharing

Pose mode: https://drive.google.com/file/d/1Ns1Km6zk1HBUYVANh3LVfth-JhCjI-Qf/view?usp=sharing

Virtual Objects: https://drive.google.com/file/d/1hZq8UvSJKCmvC21ErEqySvnEXgOj__G9/view?usp=sharing

Harris corner: https://drive.google.com/file/d/1e1J57czhczhMflYMtV8R-iFuQ_p9zwL9/view?usp=sharing

Detect robust feature: explain how you might be able to use those feature points as the basis for putting augmented reality into the image.?

So its similar to doing getcheckerboardCorners instead here the goodFeaturesToTrack gives me the corners and then based on my shape in the real world I can pick a unit of measure and define world 3d coordinates of those corners. i.e based on my choice of origin and unit I can decide 3d coordinates in the plane of the image and then like previous tasks I can call caliberateCamera with these corners and actual world coordinates and proceed

Ishan Tripathi
Naveen Verma

Extension:

The overall idea of the extension is to build a **Lego like augmented reality framework**. In play mode of the program:

- The user can create multiple 3D objects of different shapes.
- Increase/Decrease the size of the shape.
- Move the 3D objects around to build anything from those shapes.

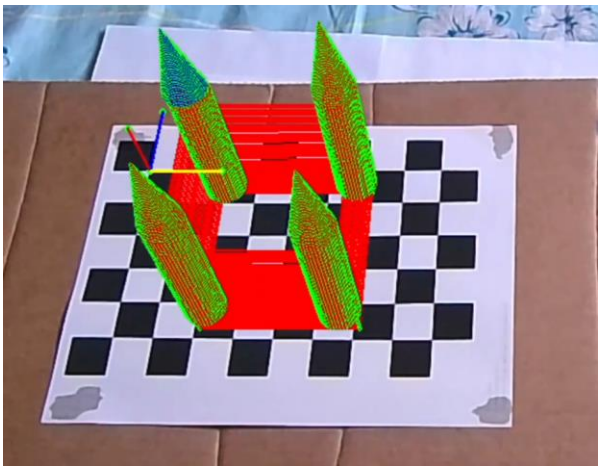
Controls to play around: Following keys needs to be used to perform the operation.

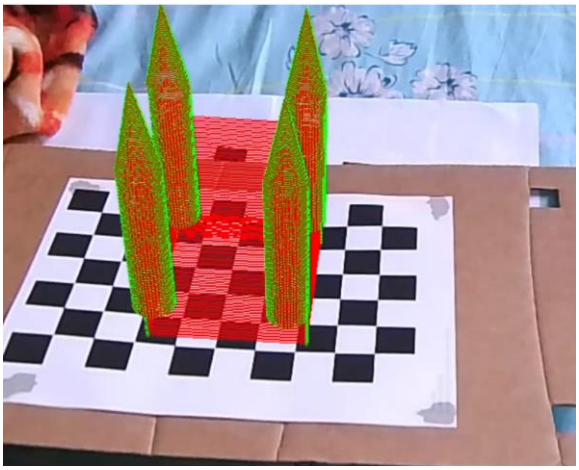
- Object creation control:
  - **Key 's' will create a sphere.**
  - **Key 'c' will create a cylinder.**
  - **Key 'o' will create a Cone.**
  - **Key 'u' will create a Cube.**
  - **Key 'v' will display the 3D axes on the origin for help and pressing v again removes it.**
- Object selection control:
  - **Key 't/T' will switch between virtual 3D objects.**
  - **The 'blue colored' object will be the one that is currently selected.**
  - **We can change the size or move around the currently selected object.**
- Object size control: (Blue colored object is the selected object)
  - **Key 'R' will increase the radius of object.**
  - **Key 'r' will decrease the radius of object.**
  - **Key 'H' will increase the height of object.**
  - **Key 'h' will decrease the height of object.**
- Object Movement control: (Blue colored object is the selected object)
  - **Key 'X' will move the object in positive x direction.**
  - **Key 'x' will move the object in negative x direction.**
  - **Key 'Y' will move the object in positive y direction.**
  - **Key 'y' will move the object in negative y direction.**
  - **Key 'Z' will move the object in positive z direction.**
  - **Key 'z' will move the object in negative z direction.**
- Change the central axis of the selected object between x, y and z axis:
  - **Key 'p' will shift between the planes X, Y and Z axis.**
- Object Deletion control:
  - **Key 'd' will delete the currently selected object (blue colored object).**
- Save the current frame:
  - **Key 'S' will save the current frame as 'saved.jpg'**

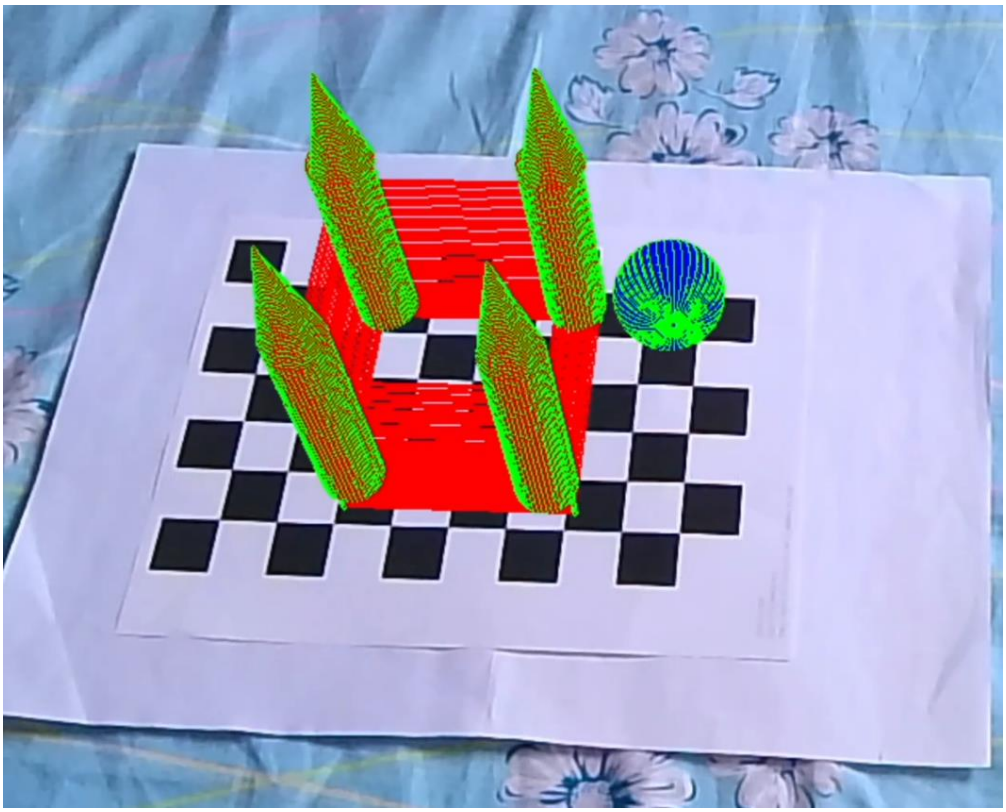
Ishan Tripathi
Naveen Verma

Extension Images:

Making fort using cube, cone, and cylinder:







Extension Videos:

Fort making time lapse video: https://drive.google.com/file/d/1KeqKe2gAFWm8F7oTzuU0GCYGkvAIxdIs/view?usp=sharing

Simple object moving video: https://drive.google.com/file/d/1Jx5GnHg7VvaPSGavctLTe2HvgDF8W9Rl/view?usp=sharing

Ishan Tripathi
Naveen Verma

Reflection on learning through this project:

The project helped us learn two main things:

- How a camera calibration works: What intrinsic and extrinsic values are and how they can be extracted using OpenCV and be used to map the actual world 3D coordinates into camera 2D coordinates.
- How camera calibration can further be used for creating augmented reality i.e. virtual objects in the scene: We can construct virtual 3D objects by constructing their real world 3D points in the code and make it appear on the camera frame which will calculate the 2D coordinates of those points in the camera Euclidian. And since their 3D world points are fixed w.r.t. world coordinates, hence they will orient itself correctly relative to the target that is defining the world 3D points even when we move the camera or the target.

Acknowledgement:

OpenCV documentation.

Lecture Recordings.