

# COL780 Assignment 3

Samarth Bhatia  
2019CH10124

## Part 1: Intrinsic Camera Parameter Calculation

### Details/Formulation

I will be using Zhang's method to perform calibration (using a checkerboard pattern) to calculate intrinsic parameters for a camera projecting 3d world coordinates to 2d image coordinates.

Note that I have avoided using `cv2` functions unless necessary: I don't use even the `findChessboardCorners` function and use a non-standard checkerboard pattern.

Let  $[x, y, z]$  be a point in the 3d space, and its corresponding point in the image be  $[u, v]$ . Then, we write:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (1)$$
$$P = MX$$

However, this does not separate the intrinsic parameters (that map the camera 3d coordinates to the 2d image plane) from the extrinsic parameters (that map 3d world coordinates to 3d camera coordinates).

We can write this matrix as:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f \cdot m_x & \gamma & v_0 \\ 0 & f \cdot m_y & u_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (2)$$
$$P = K[R|t]X$$

Here  $K$  is the intrinsic parameter matrix.  $f$  is the focal length,  $m_x, m_y$  are pixel densities of the sensor in x and y direction,  $\gamma$  is the skew in x-direction, and  $(u_0, v_0)$  is the image of the principal point.

If we take a flat checkerboard pattern, we can set the 3d coordinate system to be at one of the corners in the plane of the pattern. This gives us that  $z = 0$  for all points on the checkerboard, and their  $x, y$  coordinates can be easily calculated.

So, putting  $z = 0$  in 2, we can ignore the 3<sup>rd</sup> column of  $[R|t]$ , the extrinsic parameter matrix. So, our eqn becomes:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f \cdot m_x & \gamma & v_0 \\ 0 & f \cdot m_y & u_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3)$$

or,

$$P = HX$$

So, we find multiple correspondences on the checkerboard in the 2d and 3d coordinates  $(x_i, y_i, 0) \rightarrow (u_i, v_i)$ , and use them to find a  $3 \times 3$  homography instead of a  $3 \times 4$  projection. We need 4 points to determine H.

**Note that intrinsic parameters are same for all checkerboard points in all images but extrinsic parameters are only same for a single frame.**

$$h = [H_{.1} \quad H_{.3} \quad H_{.2}]^T$$

$$a_{u,i}^T = [-x_i \quad -y_i \quad -1 \quad 0 \quad 0 \quad 0 \quad u_i x_i \quad u_i y_i \quad u_i] \quad (4)$$

$$a_{v,i}^T = [0 \quad 0 \quad 0 \quad -x_i \quad -y_i \quad -1 \quad v_i x_i \quad v_i y_i \quad v_i]$$

Since  $H$  is not a product of an upper triangular matrix and a rectangular matrix anymore (as in the Direct Linear Transform method), we can't use QR factorization to get  $K$  and  $[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}]$ . So, we make use of the fact that:

1.  $K$  is invertible (as it is an upper-triangular matrix and has non-zero diagonal elements)  
 $\mathbf{r}_1 = K^{-1} \mathbf{h}_1$  and  $\mathbf{r}_2 = K^{-1} \mathbf{h}_2$
2.  $\mathbf{r}_1$  and  $\mathbf{r}_2$  are orthogonal to each other (as they are part of a rotation matrix),  $\mathbf{r}_1^T \mathbf{r}_2 = 0$
3.  $\|\mathbf{r}_1\| = \|\mathbf{r}_2\| = 1$

These give us:

$$\mathbf{h}_1^T B \mathbf{h}_2 = 0$$

$$\mathbf{h}_1^T B \mathbf{h}_1 - \mathbf{h}_2^T B \mathbf{h}_2 = 0 \quad (5)$$

where,  $B = (K^{-1})^T K^{-1}$  is positive definite

Since  $B$  is symmetric, we only have 6 unknowns (consider the upper triangle),

$\mathbf{b} = [b_{11}, b_{12}, b_{13}, b_{22}, b_{23}, b_{33}]$ .

We can write 5 in a linear form:

$$\mathbf{v}_{12}^T \mathbf{b} = 0 \text{ and } (\mathbf{v}_{11}^T - \mathbf{v}_{22}^T) \mathbf{b} = 0$$

where,

$$\mathbf{v}_{ij} = \begin{bmatrix} h_{1i} h_{1j} \\ h_{1i} h_{2j} + h_{2i} h_{1j} \\ h_{1i} h_{3j} + h_{3i} h_{1j} \\ h_{2i} h_{2j} \\ h_{2i} h_{3j} + h_{3i} h_{2j} \\ h_{3i} h_{3j} \end{bmatrix} \quad (6)$$

or,

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ \mathbf{v}_{11}^T - \mathbf{v}_{22}^T \end{bmatrix} \mathbf{b} = 0$$

This only gives us 2 equations, so we do the same procedure for multiple images, say  $n \geq 3$ , and stack them together - like we stack multiple points together to find transformations. Also, we add a constraint  $\|\mathbf{b}\| = 1$  to remove the trivial solution  $\mathbf{b} = \mathbf{0}$ . This gives us:

$$\begin{bmatrix} \mathbf{v}1_{12}^T \\ \mathbf{v}1_{11}^T - \mathbf{v}1_{22}^T \\ \vdots \\ \mathbf{v}n_{12}^T \\ \mathbf{v}n_{11}^T - \mathbf{v}n_{22}^T \end{bmatrix} \mathbf{b} = \mathbf{0} \quad (7)$$

$$V\mathbf{b} = \mathbf{0}$$

or

$$\min_{\mathbf{b}} \|V\mathbf{b}\|$$

$$s. t. \|\mathbf{b}\| = 1$$

So, to minimize this (or find a vector in the null-space of  $V$ ,  $\mathcal{N}(V)$ ), we find the SVD of  $V$ . Let  $V = MSN^T$  be the SVD where  $M$  is  $2n \times 6$ , and  $S$  and  $N$  are  $6 \times 6$ .

Then, we choose  $\mathbf{b} = \mathbf{n}_6$  (the singular vector belonging to smallest singular value of  $V$ ). Rearrange this to find  $B$ .

Now, if we find the cholesky decomposition of  $B$ ,

$$\text{chol}(B) = AA^T. \quad (8)$$

Clearly, from 5,  $A = (K^{-1})^T$

Now we have  $K$ , the required intrinsic camera parameters.

## Results

The camera matrix I obtained is (using 3 images):

$$\begin{bmatrix} 0.5406 & 0.3345 & 765.37 \\ 0 & 0.6653 & 214.96 \\ 0 & 0 & 1 \end{bmatrix} \quad (9)$$

**Note that the values are different in magnitude because I have chosen the points in cm.**

Comparing to the original focal length of my phone's camera <sup>1, 2</sup>, we are not far off from the actual value ( $\approx 0.39$ ). And the center of the image is also correct (image dimensions were  $2016 \times 930$ ). Note that using more than 3 images, our objective will try to minimize  $\|V\mathbf{b}\|$  and so the translation column will be wrong (i.e. the last singular value is not the right solution anymore).

However, we are getting a *skew* parameter of 0.3345, which is not ideal, and we are limited by the SVD and the Cholesky decomposition to be able to choose a more accurate mathematical model of the  $K$  matrix.

## Part 2: Placing 3D objects in the image (AR)

---

For this part, there is not much math involved. However, we do need to find an estimate for the projection matrix from the camera parameters and the homographies.

$$\begin{aligned} KH &= [R \mid t] \\ R &= [R_1 \ R_2] \end{aligned} \quad (10)$$

Now, we might not find  $R_1$  and  $R_2$  that are orthogonal, and we need to find  $R_1, R_2, R_3$  s.t. they are orthonormal. We can simply:

$$\begin{aligned} R'_1 &= R_1 + R_2 \\ R'_2 &= R_1 \times R_2 \\ \implies R'_1 \times R'_2 &= 0 \text{ [orthogonal]} \\ \text{Take } R'_3 &= R'_1 \times R'_2 \implies \text{orthonormal vectors} \\ P &= [R'_1 \ R'_2 \ R'_3 \ t] \end{aligned} \quad (11)$$

Now, we can easily project points from 3D to 2D by

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underset{3 \times 3}{K} \underset{3 \times 4}{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (12)$$

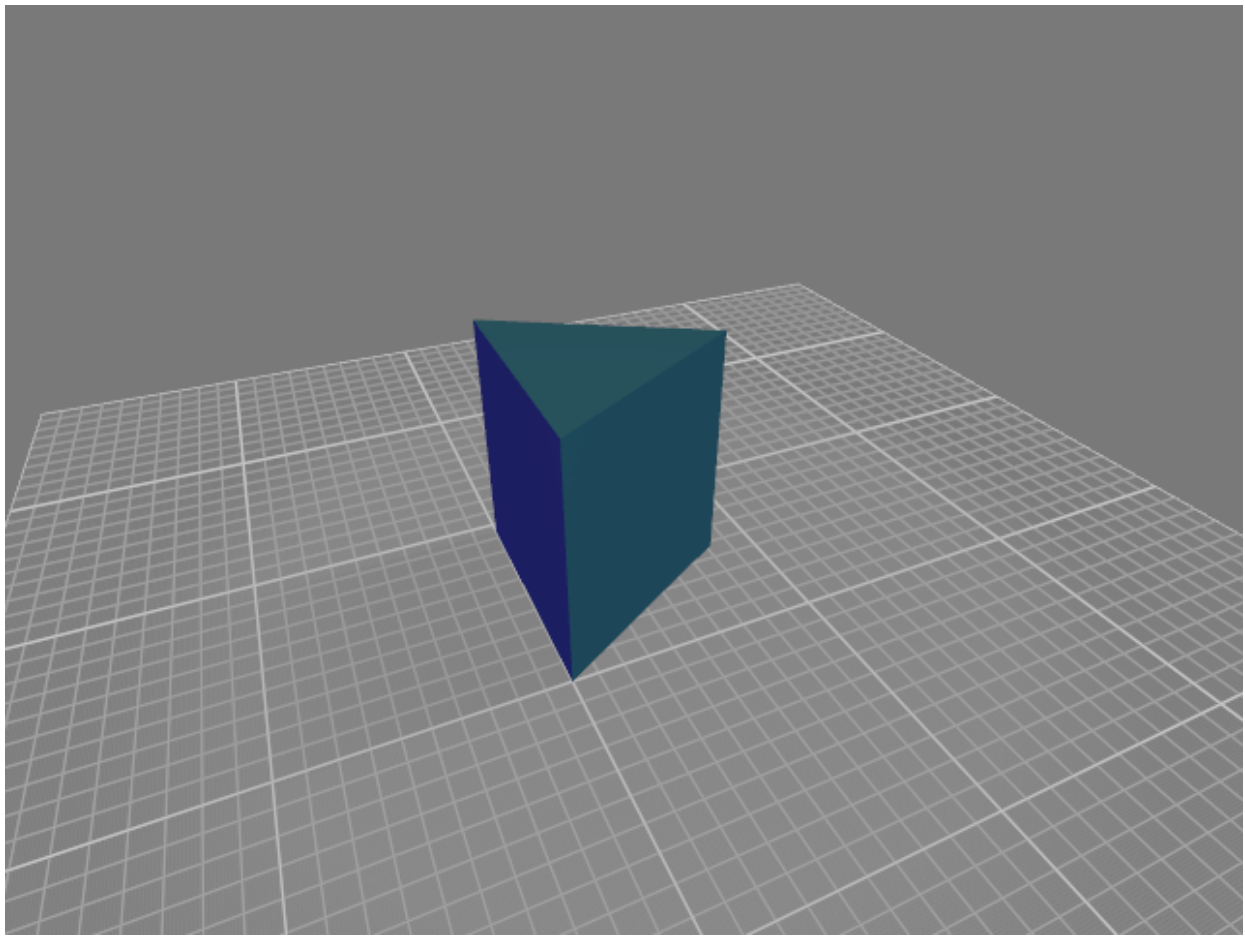
Now that we have the pixel coordinates, we can simply fill the pixels with the correct colors and we have ourselves an AR image!

## Results

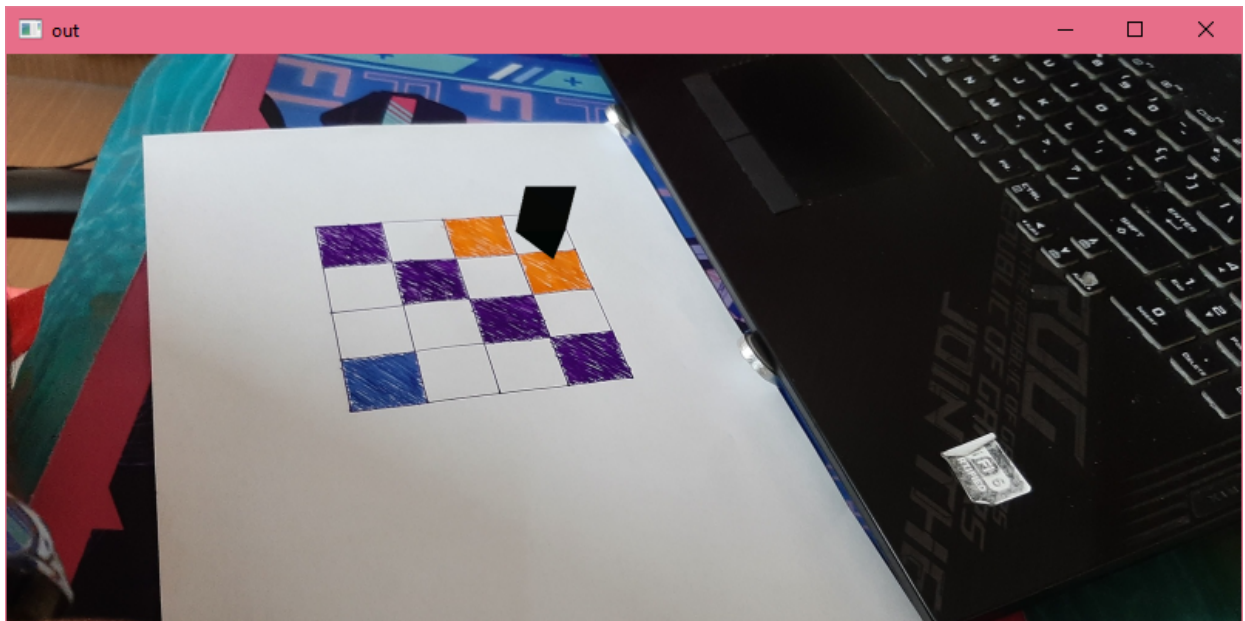
*I was not able to get loading of the texture file working, so the color functionality does not work. However, there is enough detail in the images to tell that the code is working and the objects are situated on the plane.*

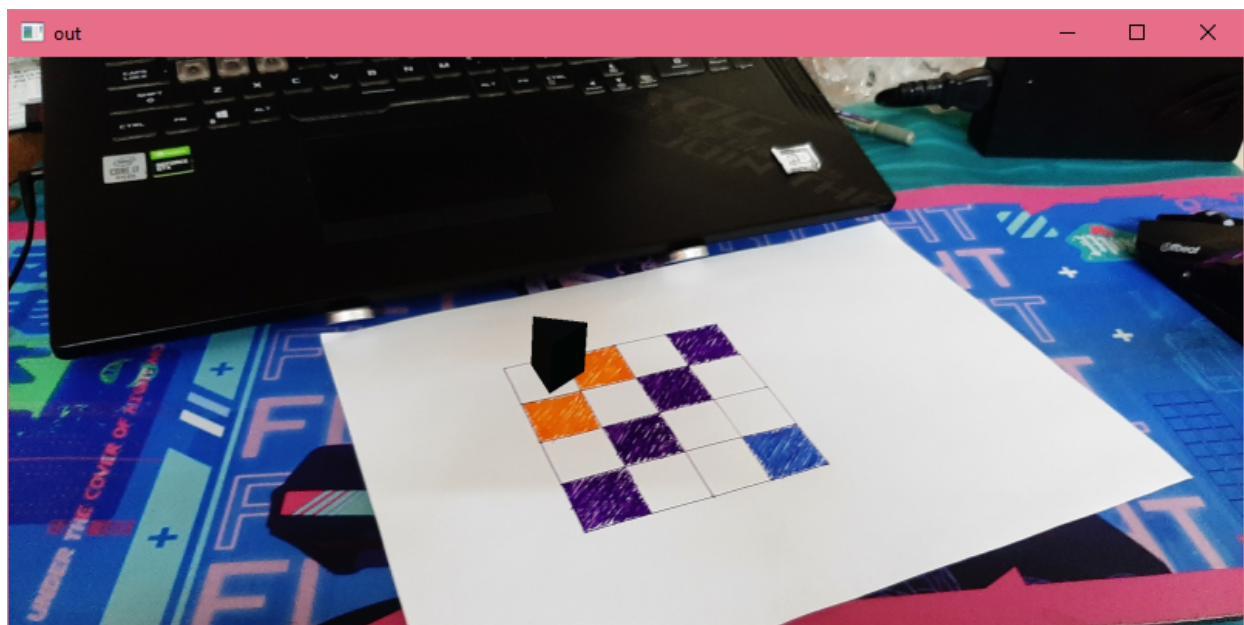
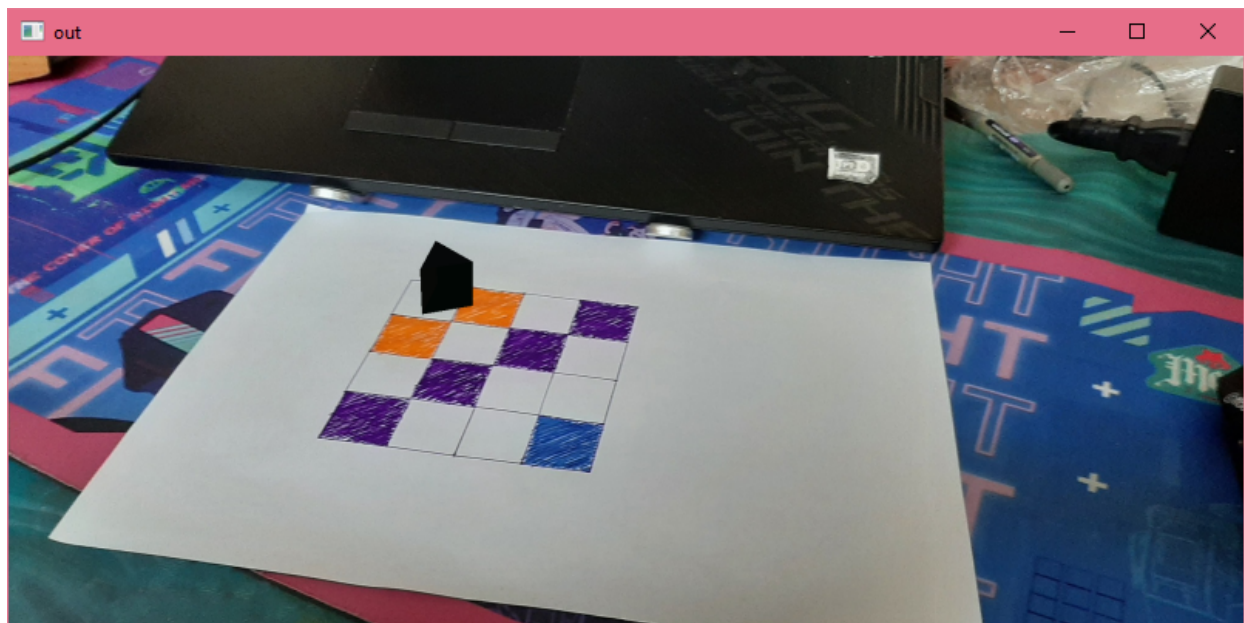
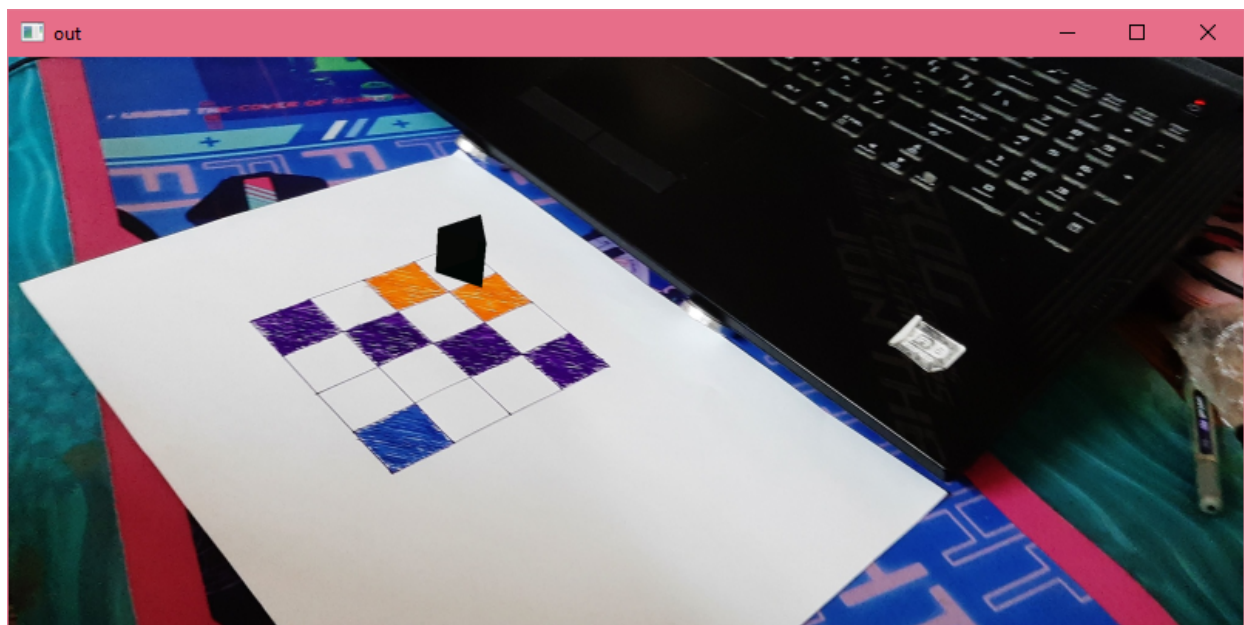
### Simple Prism

3D Model:



Projected into images:



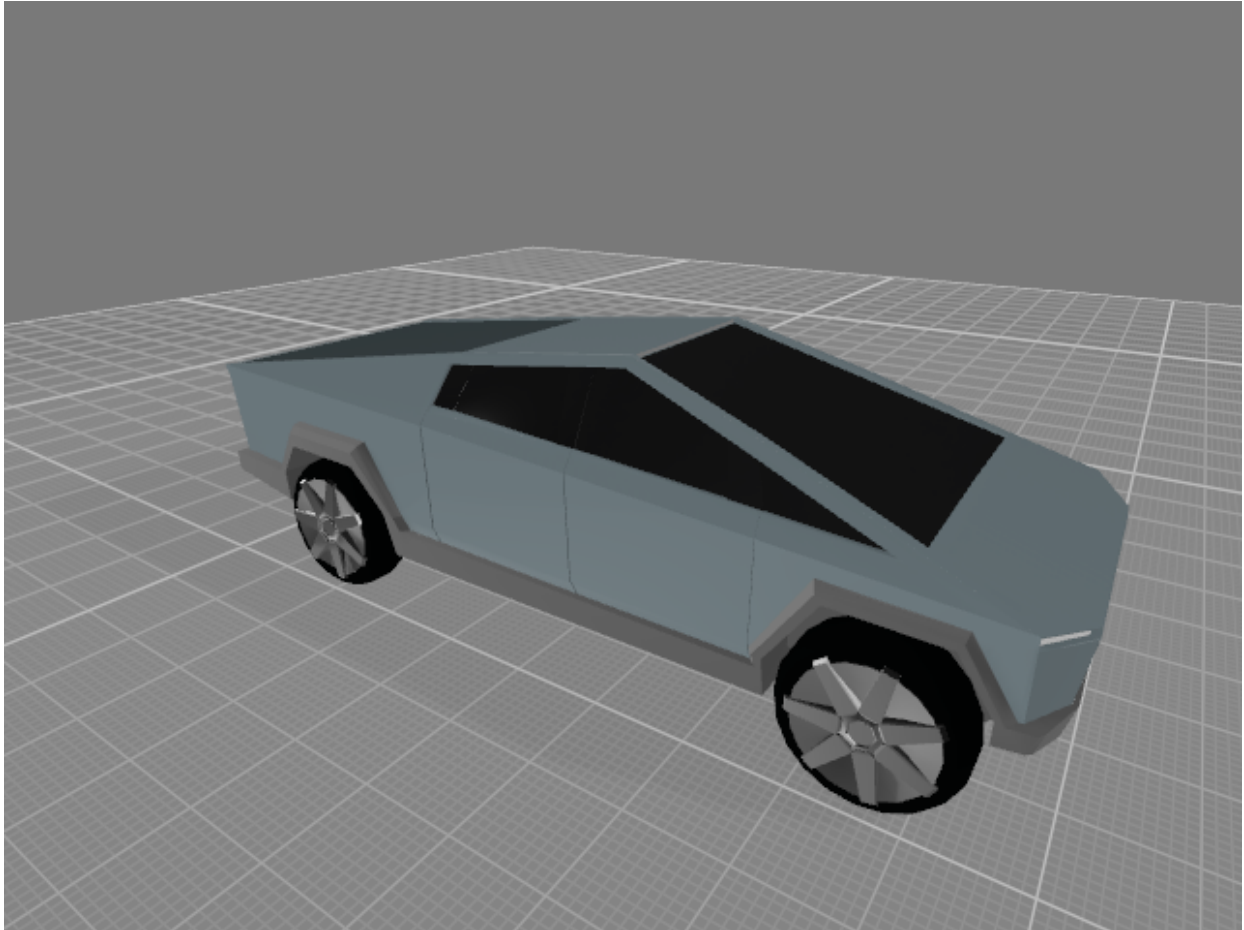




*Even with a change in angle, the projected prism has one face with the diagonal of the checkerboard as a normal, just like in previous images.*

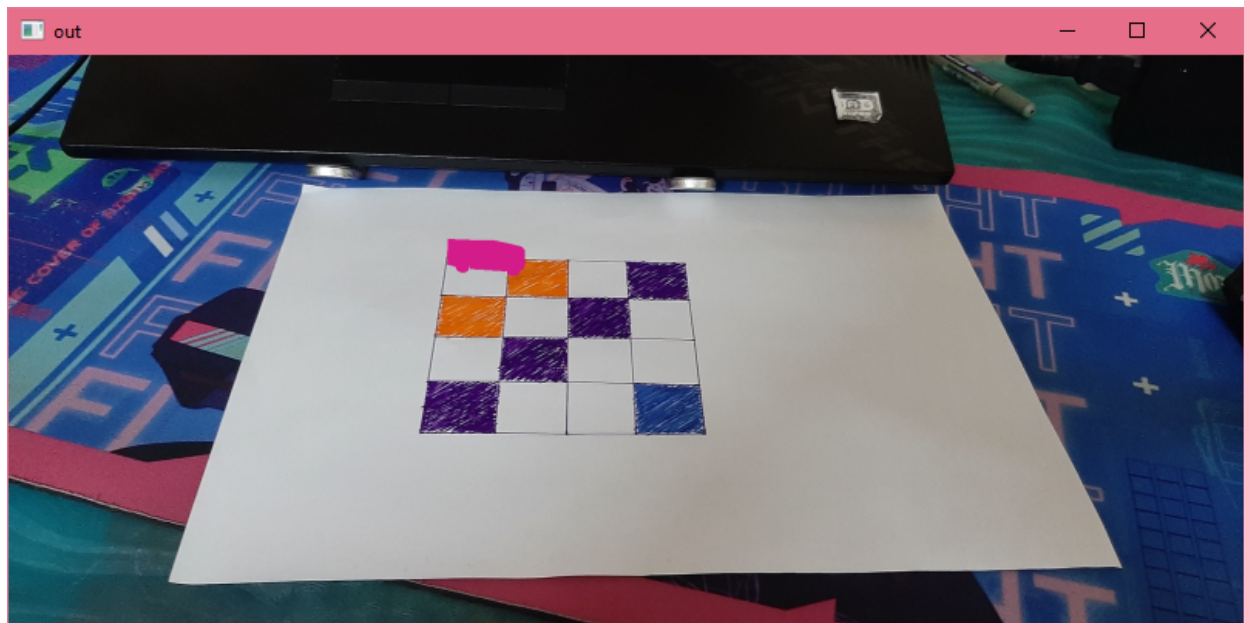
### **With a car**

3D Model:



Projected into images:





## Usage

```
1 usage: main.py [-h] [--color] [--model MODEL]
2
3 optional arguments:
4   -h, --help      show this help message and exit
5   --color          To give colors to faces
6   --model MODEL    which model to show: -1=None 0=Prism 1=Cybertruck 2=Building
                    3=Katana
```


`main.py` contains the driver code to run everything.

`find_intrinsic_params.py` contains the code for Part 1.

`overlay_object.py` contains the code for Part 2.

## References



- 
1. [https://www.camerafv5.com/devices/manufacturers/samsung/sm-a505g\\_a50\\_0/](https://www.camerafv5.com/devices/manufacturers/samsung/sm-a505g_a50_0/) 
  2. <https://learnopencv.com/approximate-focal-length-for-webcams-and-cell-phone-cameras/> 