**PART-1: Camera Calibration (K)**

Steps for obtaining camera calibration

1. Click 6 images of a checkerboard using the mobile camera.
2. Pick 4 corresponding points on each image such that they should be far from each other and form a rectangle in the original checkerboard. Let it be p1, p2, p3, p4.
3. Now get 2 vanishing points for each image.
   i.e Let l1 be the line obtained from (p1,p2) and l2 be the line obtained from (p3,p4) now the intersection of l1 and l2 gives a vanishing point v1.
   Similarly Let l3 be the line obtained from (p1,p3) and l4 be the line obtained from (p2,p4) now the intersection of l3 and l4 gives a vanishing point v2.
4. So we are obtained with a total of 12 points i.e (2 points from each image and 6 such images).
5. Now calculate K using the DLT method.

**get_line():** Output line coordinates given two points as input
**get_point():** Output point coordinates given two lines as input
**get_matrix():** Used for solving K i.e by giving input as vanishing points generated a 12x5 matrix.

Below is the derivation for getting K. i.e DLT and SVD are used for calculating W.
i.e $W = (K^{-T})(K^{-1})$

$$K = \begin{bmatrix} f_x & 0 & u_x \\ 0 & f_y & u_y \\ 0 & 0 & 1 \end{bmatrix} \qquad (K^{-1})^T = \frac{1}{f_x f_y} \begin{bmatrix} f_y & 0 & 0 \\ 0 & f_x & 0 \\ -u_x f_y & -f_x u_y & f_x f_y \end{bmatrix}$$

$$K^{-1} = \frac{1}{f_x f_y} \begin{bmatrix} f_y & 0 & -u_x f_y \\ 0 & f_x & -u_y f_x \\ 0 & 0 & f_x f_y \end{bmatrix}$$

$$w = (K^{-1})^T K^{-1} = \frac{1}{(f_x f_y)^2} \begin{bmatrix} f_y^2 & 0 & -u_x f_y^2 \\ 0 & f_x^2 & -u_y f_x^2 \\ -u_x f_y^2 & -u_y f_x^2 & u_x^2 f_y^2 + u_y^2 f_x^2 + f_x^2 f_y^2 \end{bmatrix}$$

$$w = \begin{bmatrix} 1/f_x^2 & 0 & \frac{-u_x}{f_x^2} \\ 0 & 1/f_y^2 & \frac{-u_y}{f_y^2} \\ \frac{-u_x}{f_x^2} & \frac{-u_x}{f_y^2} & \frac{u_x^2}{f_x^2} + \frac{u_y^2}{f_y^2} + 1 \end{bmatrix} = \begin{bmatrix} w_{11} & 0 & w_{13} \\ 0 & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

$$v_2^T w v_1 = 0 \Rightarrow \begin{bmatrix} x_2 & y_2 & z_2 \end{bmatrix} \begin{bmatrix} w_{11} & 0 & w_{13} \\ 0 & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = 0$$

$$\begin{bmatrix} (x_2 w_{11} + z_2 w_{31}) & (y_2 w_{22} + z_2 w_{32}) & (w_{13} x_2 + w_{23} y_2 + z_2 w_{33}) \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = 0$$

$$\Rightarrow x_1 x_2 w_{11} + x_1 z_2 w_{31} + y_1 y_2 w_{22} + w_{13} x_2 z_1 + w_{23} y_2 z_1 + z_1 z_2 w_{33} = 0$$

$$\begin{bmatrix} (x_1 x_2) & (x_1 z_2 + x_2 z_1) & (y_1 y_2) & (y_2 z_1 + y_1 z_2) & (z_1 z_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{8\times 5} \begin{bmatrix} w_{11} \\ w_{13} \\ w_{22} \\ w_{23} \\ w_{33} \end{bmatrix}_{5\times 1} = 0$$

**PART-2: Getting [R|t] for each image and getting image points for the object projection.**

The image was clicked at a scale of particular height(here 9). World points and corresponding image points are taken i.e 5 pairs. Now below derivation is applied for getting [R|t] then cube and pyramids are plotted. Then DLT is applied for getting [R|t].

**pyramid()** and **cube()** are functions used for plotting the world points on image.

The below derivation is applied.

$$PX = x$$

$$K[R|t]\,X = x$$

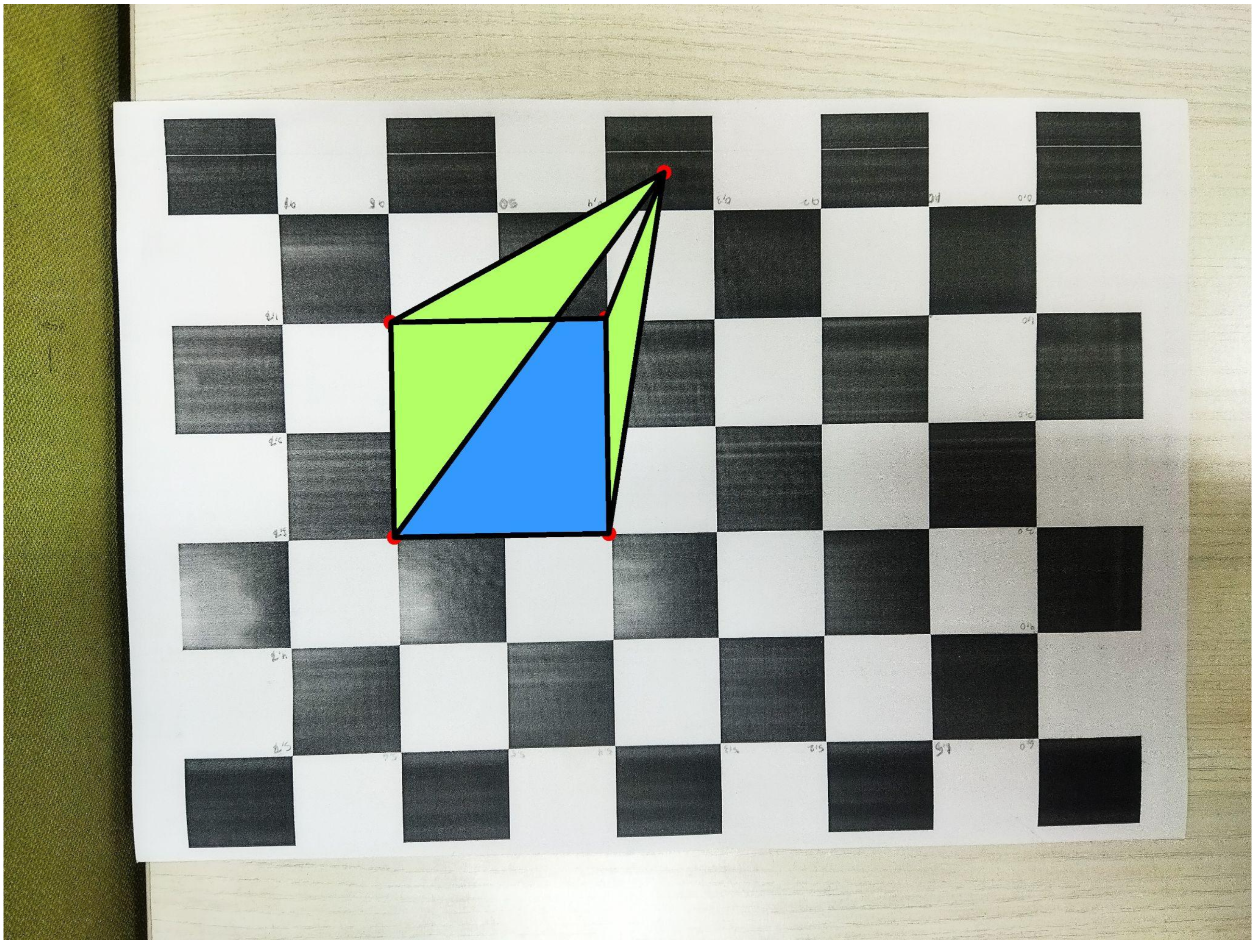
$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

i.e

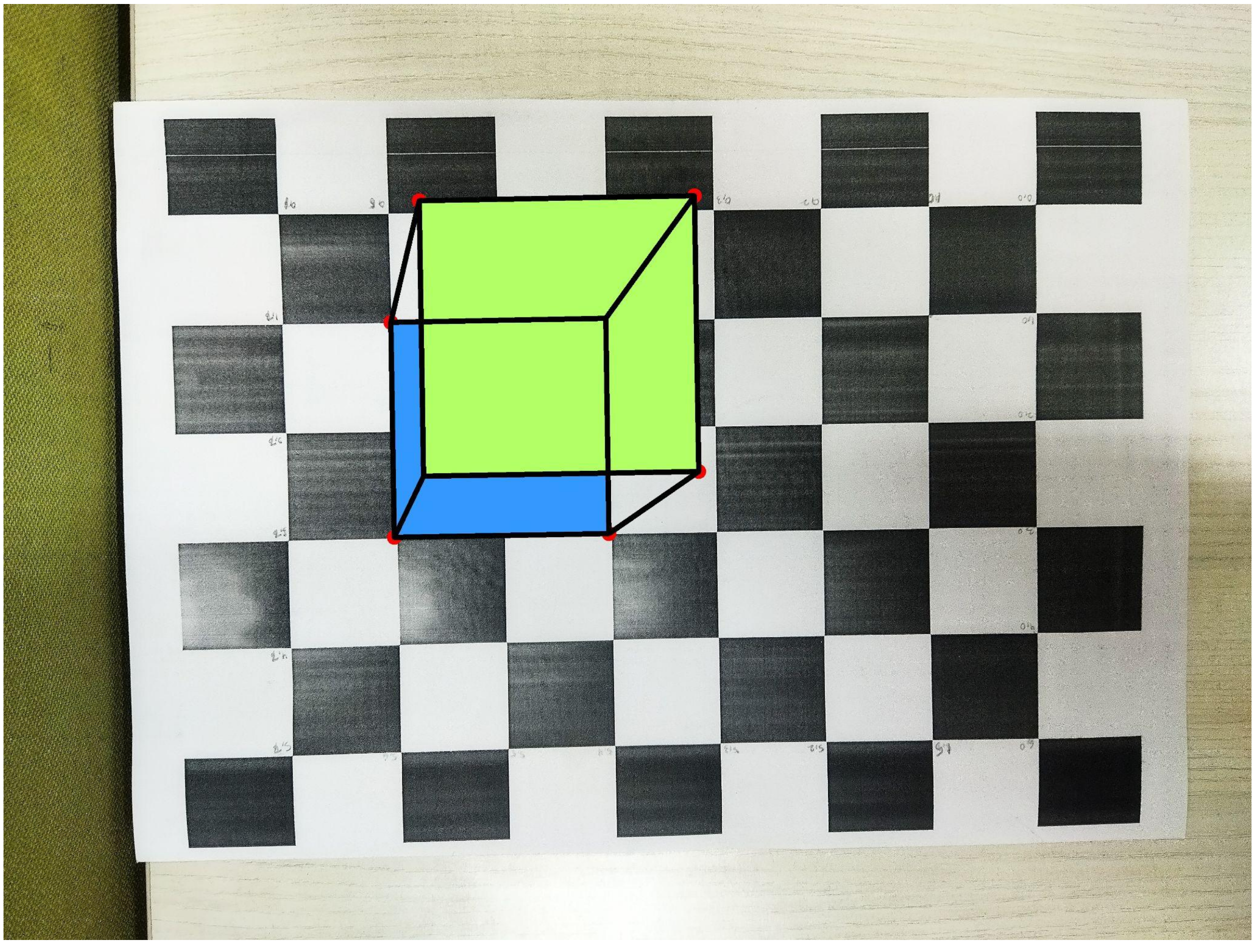
$$\underbrace{\begin{bmatrix} X_1 & Y_1 & Z_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 0 & 0 & 1 \\ X_2 & Y_2 & Z_2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 0 & 0 & 1 \\ X_3 & Y_3 & Z_3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & X_3 & Y_3 & Z_3 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X_3 & Y_3 & Z_3 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & & & & & \vdots & & & & \vdots \end{bmatrix}}_{3n \times 12} \underbrace{\begin{bmatrix} r_{11} \\ r_{12} \\ r_{13} \\ r_{21} \\ r_{22} \\ r_{23} \\ r_{31} \\ r_{32} \\ r_{33} \\ t_1 \\ t_2 \\ t_3 \end{bmatrix}}_{12 \times 1} = \underbrace{K^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{3n \times 1}$$

n → no of points

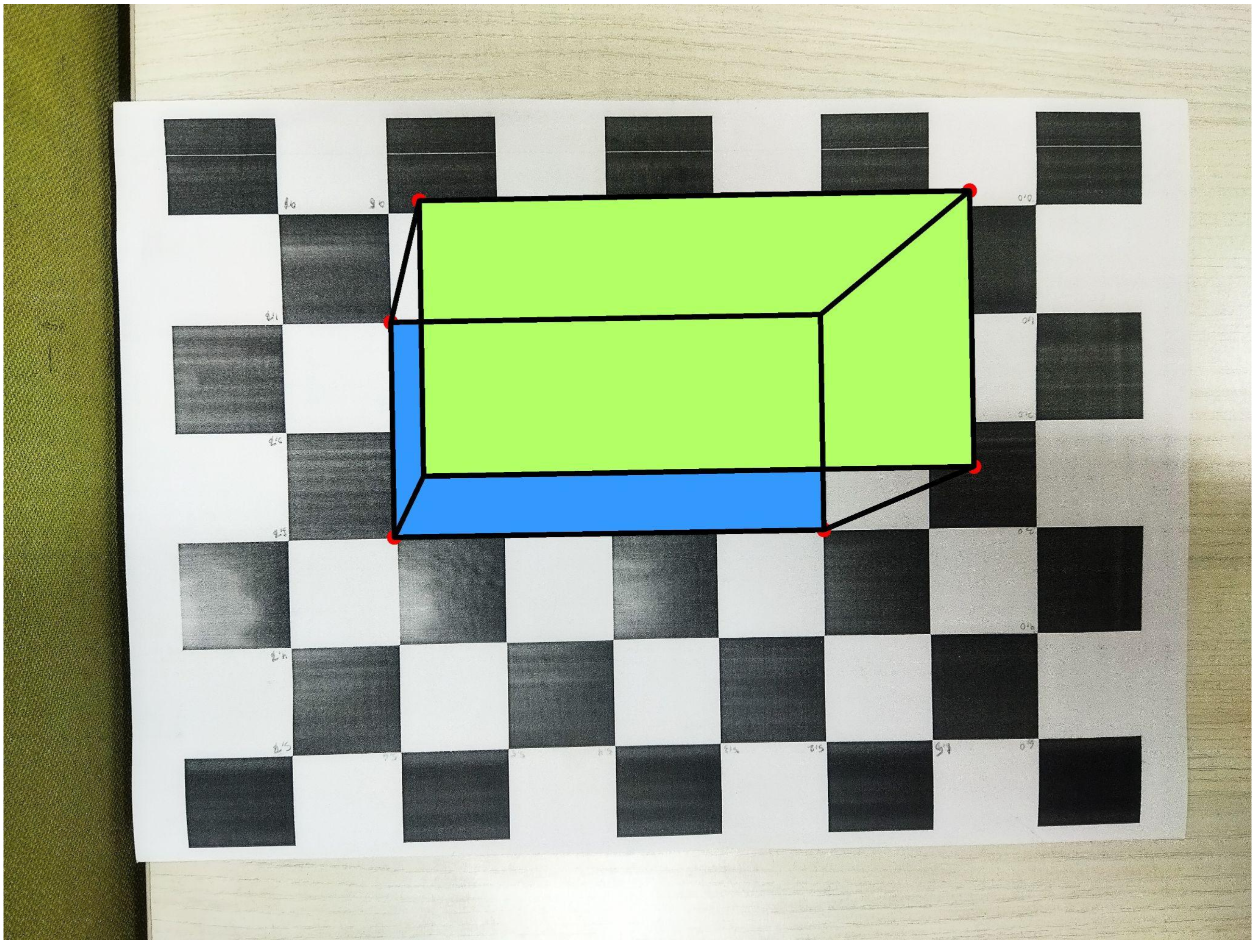
# Pyramid



# Cube



# Cuboid



# CuboidPyramid