**GIVEN TASK:**

* Given a set of Aruco markers, all in different orientation, the task was to place the Aruco markers on squares on an image consisting of various shapes.
* Each Aruco markers is referenced with an ID of its own. We were provided with the IDs of the Aruco markers corresponding to the colours of the various squares. Our task was to place the correct Aruco markers on the square with color corresponding to its ID. The ID of the Aruco Marker corresponding to the color of the squares was given to us.

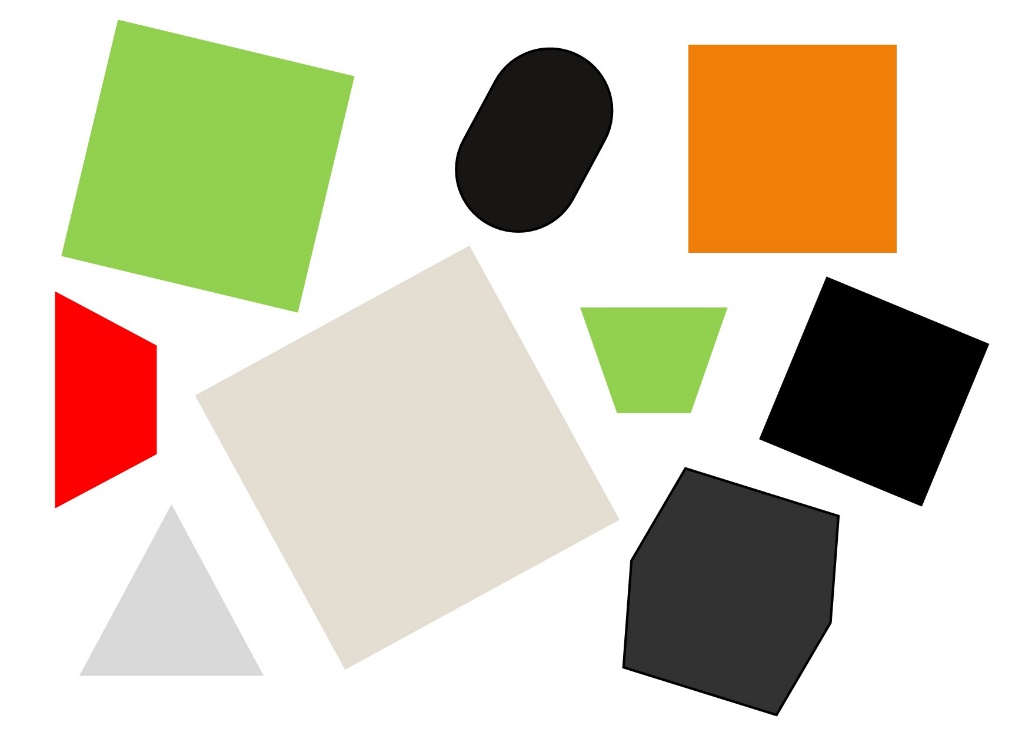
**ALGORITHM:**

* Our first task was to get the IDs of the Aruco Markers using the in-built Aruco sub-library of CV2 library.
* These IDs were stored along with the filenames in a dictionary having the IDs as the key and the corresponding filename as the value pairs.
* Then since the Aruco markers were all in different orientation, I rotated all of them to the straight orientation using cv2.getRotationMatrix() and cv2.WarpAffine() methods.
* The next step was to detect squares in the image given to us. For this a copy of the image was created which was then converted into threshold image. All the pixel values between 240 and 255 were converted to white, the remaining pixel values were converted to black.
* Using contour detection, the shapes in the image were identified.
* Then using the concept that the sides of a square are equal, the contours were filtered so as to detect only the squares.
* The next task was to place the Aruco markers with correct orientation on the squares.
* For this, each square was selected using for loop, then the whole image was rotated by the angle made by the square with the x-axis. The purpose of doing this was so that the image becomes oriented such that one square is always in its correct orientation so that the aruco markers can be easily placed on that.
* The Aruco markers, which were already modified to the straight orientation, were then resized according to the size of the square. This was achieved by locating the co-ordinates of the midpoint of the square and then using if and else to determine which Aruco would be placed according to the colour of the square.
* The Aruco marker was then placed on the correct square. The image was then rotated back to its correct orientation. This process was repeated until all the Aruco markers got placed on the squares.
* The image was little blurred due to rotation and cropping, so a kernel was used to sharpen the image.

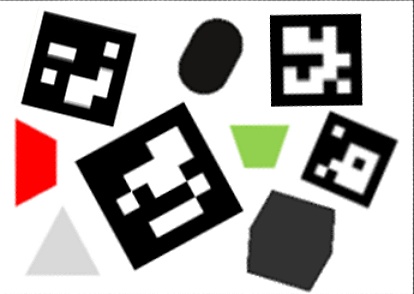
**Libraries Used:**

* cv2 – For image processing features.
* Numpy – For transforming pixel values.
* Os – For working with files.
* cv2.Aruco – For working with Aruco markers.
* Math – For functions like tan inverse and converting angle from radian to degrees.
* Imutils – For working with image rotation. Purpose of using this was that it does not crop the image while rotating it.

**Initial Image:**

****

**Final Image:**

****

To avoid edge detection doubling, I used thresholding instead of Canny Edge detection because Canny Edge detects twice: first the inner edges then the outer edges, instead thresholding detects once.

**SHREYANSH SHANDILYA**

**Indian Institute of Technology (ISM), Dhanbad**