

**SIR SYED UNIVERSITY OF ENGINEERING & TECHNOLOGY**

**Artificial Intelligence Project Report**

**Shape Detection On Image And GUI**

**Group Members:**

FARRUKH ASLAM

{2017-CE-149}

TAHA JAMAL

{2017-CE-132}

UMAIR SHAKEEL

{2017-CE-145}

USAMA AYAZ

{2017-CE-120}

**SECTION C**

**SUBMITTED TO : SIR UMAIR NAQVI**

**DEPARTMENT OF COMPUTER ENGINEERING**

[18-April-2020]

Libraries Used In Project

1.Open Cv

Open Cv  (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. **OpenCV** was built to provide a common infrastructure for computer vision applications to manipulate with images and images and to accelerate the use of machine perception in the commercial products.

2.Matplotlib.pyplot

Matplotlib.pyplot is the python library used to plot the images or we can say to show the image visually we are using in project to show the different shapes.

3.Numpy

Numpy is the python library adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays we are using in our project because images are also build on arrays.

Techniques to Detect The Shapes

Step 1

Canny Edge Detection

Edge Detection: Edge detection is an image processing technique for finding the boundaries of objects within images. It works by detecting discontinuities in brightness. Edge detection is used for image segmentation and data extraction in areas such as image processing, computer vision, and machine vision.

How It Works?



Firstly it converts a color image to grayscale image and convert more to smooth image so that noise cannot interfere then it perform edge detection. Below is the example of canny edge detection Syntax of Canny edge detection is cv2.Canny(img,threshold1,threshold2)

Step 2

Contour Detection of Images

Contours can be defined simply as a curve joining all the continuous points (along the boundary), having same color or intensity. You can use contour detection insetead of blob detection. Now in contour detection, the objects to be detected are in white and the background is black. Internally in blobdetection, findContour function is also used.

Syntax Of Cotour Detection is

contours, hierarchy = cv2.findContours(image, mode, method[, offset])

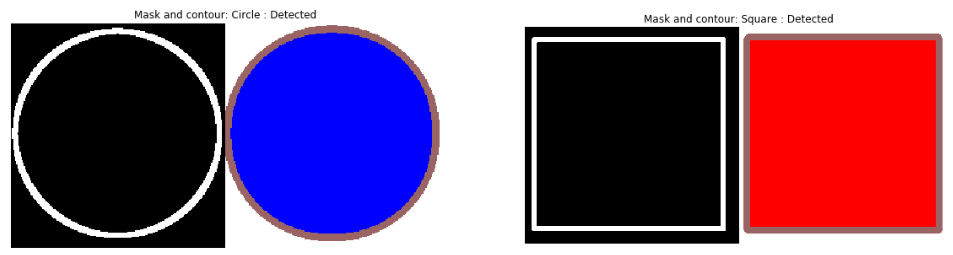
where images are not the exact color images it is the image which we get after edge detection or we have image on which edge detection already done

Here modes are the different Retrieval Modes

In Our Project We have Used:



Where mask we get from Canny Edge Detection where cv2.RETR\_EXTERNAL retrieves only the extreme outer contours and cv2.CHAIN\_APPROX\_SIMPLE: compresses horizontal, vertical, and diagonal segments and leaves only their end points.

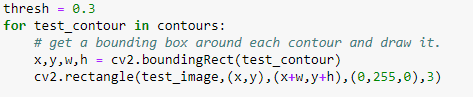


The above is the example from our project that through **canny edge detection** we found edges then these adges pass on to the **contour detection**  and we found contour from image.

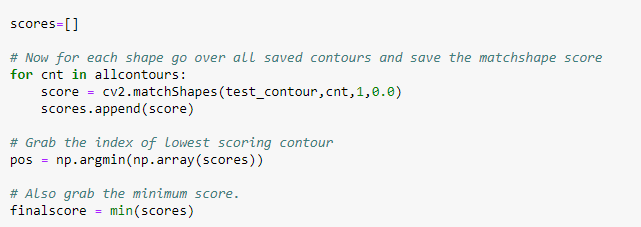
The above tecniques we have used in our project to detect shapes.



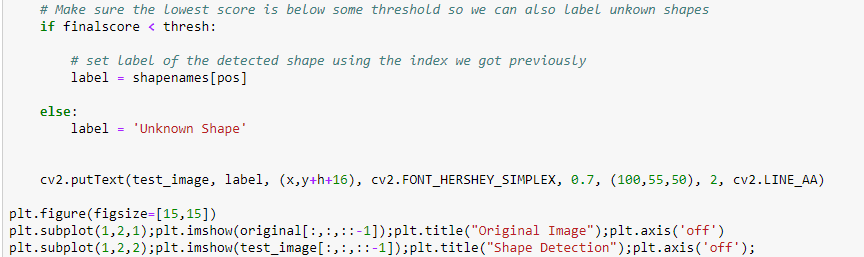
In this part of our Project we have taken image file and perform both Canny Edge Detection and Contour Detection on image we have choosen once we have found the contours we have wet thresh and draw a rectangle on a detected image which is as follows



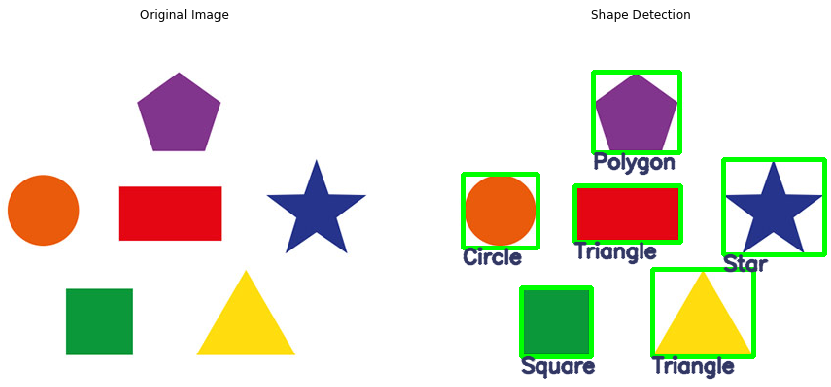
After that we have created an empty list of score to compare the scores of shape with contour and save the match shape score along with the index which is as follows



As above we also finding minimum scores in final scores it is bescause the minimum value a great detection or accurate contour after we put up a condition if final score is less than threshold whose value is 0.3 then draw labels and name the shapes otherwise Unknown shape and in the end just putting a text on image and plotting it as shown below



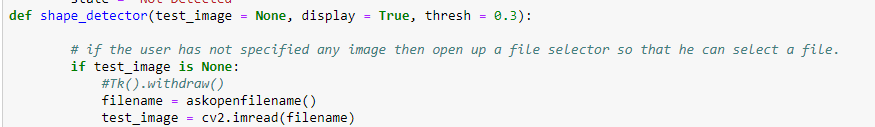
**OUTPUT**





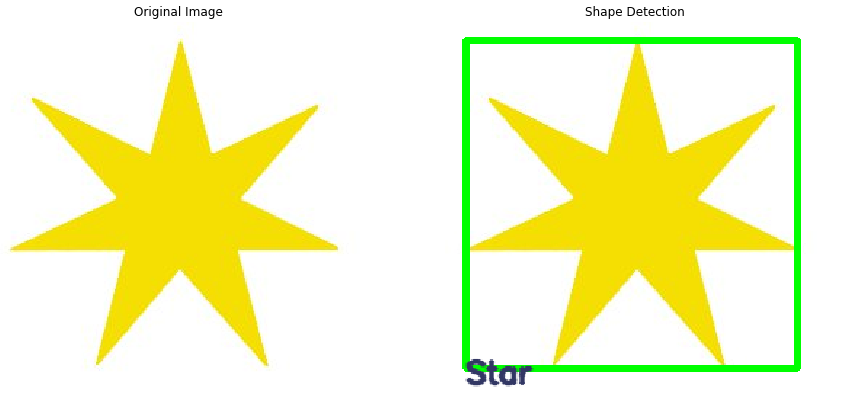
In this part of our project user can select shape image of his own choice and detection will be performed on that the steps are all same as above in part one of project two different things we have done are as follows

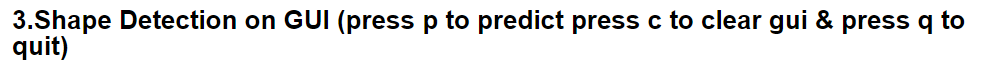




Firstly we have taken library of tkinter which allows us to select image of our choice other thing is that we have a function of shape detector as you can see above in code. Apart from that every step is same as that on **page 4** of this report.

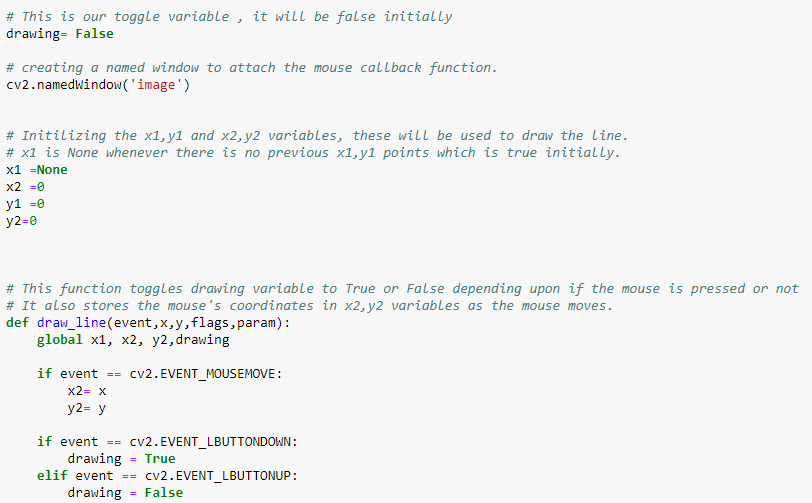
OUTPUT



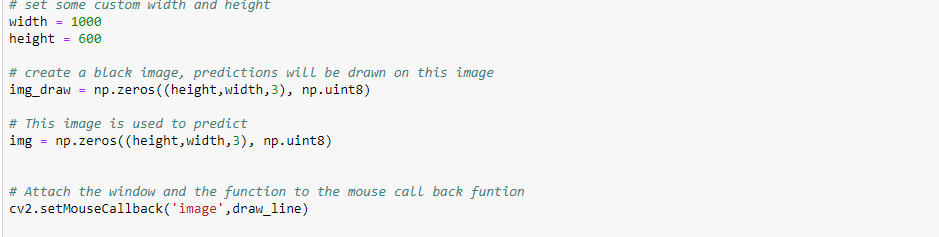


In this part of our project user can select shape image of his own choice and detection will be performed on that the steps are all same as above in part one and two on **page 4 & 5** of project further thing we have added is the code to created GUI.

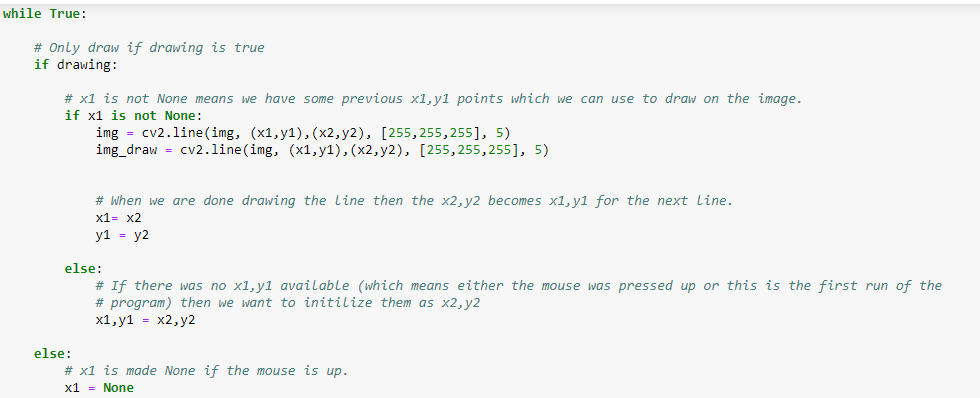
GUI Creation



In the above few lines of creating GUI what we doing is that first of all we taking drawing = false that means if we not clicking a mouse no draw anything then we are creating a window named image further we are initializing **x1,x2,y1,y2** that are used to draw lines **x1=none** because its starting point then further we have used function **draw\_line** where events are there as when we move mouse the x2 becomes x again and y2 becomes y again then if we click mose draw otherwise no.

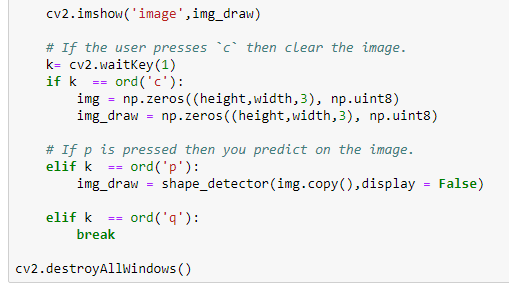


Then we aere setting width and height and creating a black screen where we will draw and at the end adding mousecallback function to window



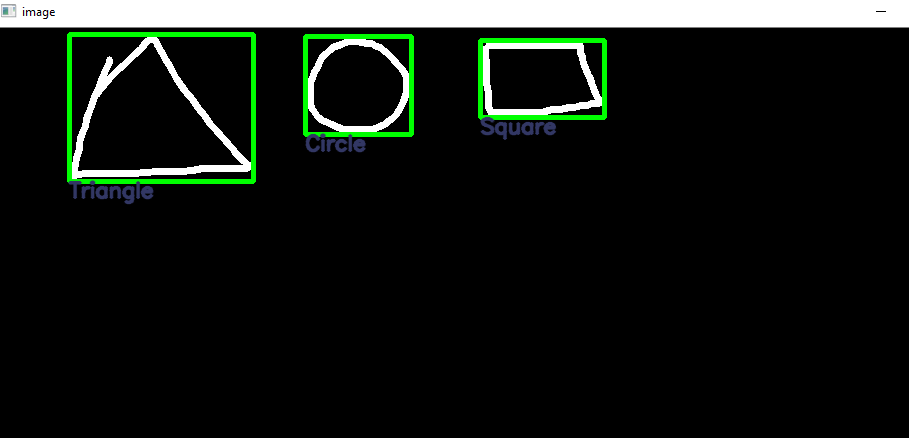
Then Further we have included While loop where condition is that if x1 is not

none x1,y1=x2,y2 if x1 is known keep drawing.



In the end we showing image that is the GUI window where if we press c it will be clear if we press p it predicts what shape it is and if we press q we will quit from GUI.

OUTPUT



THE END