**Machine Vision System Practical 6**

**Color and Shape Identification using Machine Vision in OpenCV**

In this Practical, we will learn

1. **How to Identify Colors and Shapes from a given Image.**

**1] Shape and Color Detection:**

* **Importing OpenCV & Numpy libraries:**

The code starts by importing the OpenCV library i.e. cv2 as cv, which is used for computer vision tasks, including reading and processing images. We also import numpy library as np for matrix calculations for transformation.

* **Reading the Original Image:**

Here we read our image with the help of cv.imread() function by giving it the proper path where the image is stored. And then we display the image with any name suitable using cv.imshow() function.

* **Shape Identification:**

**Code:**

import cv2 as cv

import numpy as np

img = cv.imread('Photos/shapes.jpg')

imgGrey = cv.cvtColor(img, cv.COLOR\_BGR2GRAY)

\_, thrash = cv.threshold(imgGrey, 240, 255, cv.THRESH\_BINARY)

contours, \_ = cv.findContours(thrash, cv.RETR\_TREE, cv.CHAIN\_APPROX\_NONE)

cv.imshow("img", img)

for contour in contours:

    approx = cv.approxPolyDP(contour, 0.01\* cv.arcLength(contour, True), True)

    cv.drawContours(img, [approx], 0, (0, 0, 0), 5)

    x = approx.ravel()[0]

    y = approx.ravel()[1] - 5

    if len(approx) == 3:

        cv.putText(img, "Triangle", (x, y), cv.FONT\_HERSHEY\_COMPLEX, 0.5, (0, 0, 0))

    elif len(approx) == 4:

        x1 ,y1, w, h = cv.boundingRect(approx)

        aspectRatio = float(w)/h

        print(aspectRatio)

        if aspectRatio >= 0.95 and aspectRatio <= 1.05:

          cv.putText(img, "square", (x, y), cv.FONT\_HERSHEY\_COMPLEX, 0.5, (0, 0, 0))

        else:

          cv.putText(img, "rectangle", (x, y), cv.FONT\_HERSHEY\_COMPLEX, 0.5, (0, 0, 0))

    elif len(approx) == 5:

        cv.putText(img, "Pentagon", (x, y), cv.FONT\_HERSHEY\_COMPLEX, 0.5, (0, 0, 0))

    elif len(approx) == 10:

        cv.putText(img, "Star", (x, y), cv.FONT\_HERSHEY\_COMPLEX, 0.5, (0, 0, 0))

    else:

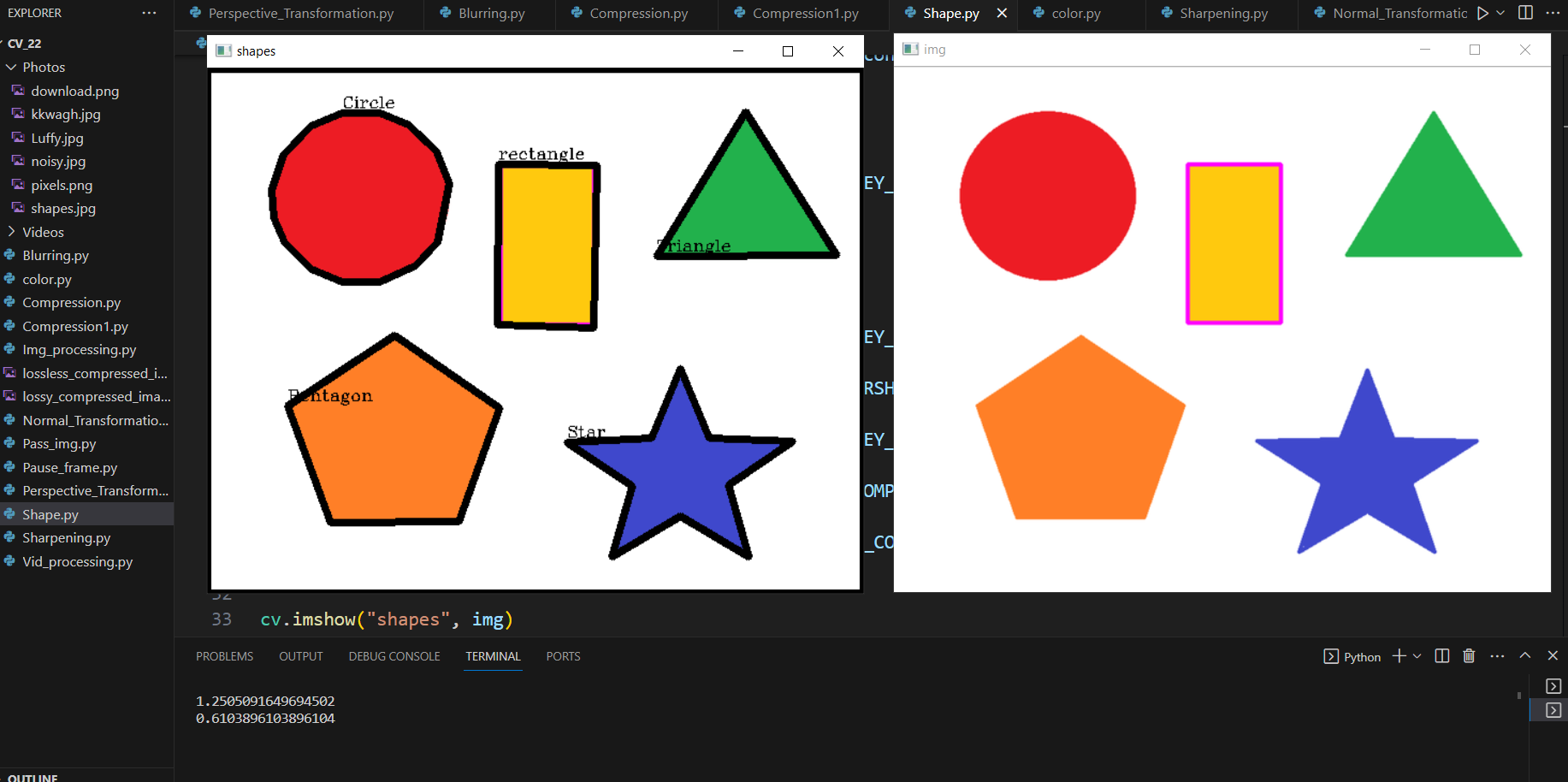
        cv.putText(img, "Circle", (x, y), cv.FONT\_HERSHEY\_COMPLEX, 0.5, (0, 0, 0))

cv.imshow("shapes", img)

cv.waitKey(0)

cv.destroyAllWindows()

**Output:**

****

* **Color Identification:**

**Code:**

import cv2 as cv

import numpy as np

# Read the image

image = cv.imread('Photos/download.png')

# Convert the image to HSV color space

hsv = cv.cvtColor(image, cv.COLOR\_BGR2HSV)

# Define color ranges for the colors you want to identify (e.g., blue)

lower\_blue = np.array([90, 50, 50])  # Lower HSV value for blue

upper\_blue = np.array([130, 255, 255])  # Upper HSV value for blue

# Create a mask to isolate the desired color

mask = cv.inRange(hsv, lower\_blue, upper\_blue)

# Find contours in the mask

contours, \_ = cv.findContours(mask, cv.RETR\_EXTERNAL, cv.CHAIN\_APPROX\_SIMPLE)

# Iterate through the detected contours and identify color

for contour in contours:

    area = cv.contourArea(contour)

    if area > 100:  # Filter out small contours

        # Draw the contour on the original image

        cv.drawContours(image,[contour], -1, (0, 255, 0), 2)

        # Get the centroid of the contour

        M = cv.moments(contour)

        if M["m00"] != 0:

            cx = int(M["m10"] / M["m00"])

            cy = int(M["m01"] / M["m00"])

            # Identify and label the color

            color = "Blue"  # You can add more color ranges and labels as needed

            cv.putText(image, color, (cx - 20, cy - 20), cv.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 255, 0), 2)

# Display the result

cv.imshow('Color Identification', image)

cv.waitKey(0)

cv.destroyAllWindows()

**Output:**

****