COMPUTER VISION

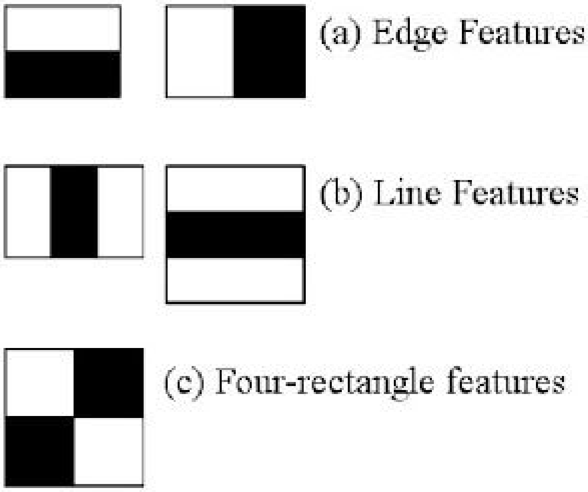
PROBLEM STATEMENT : To verify object detection by various methods and classify and segregate multiple objects by applying different algorithms of image processing.

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Well-researched domains of object detection include face detection and pedestrian detection. Object detection has applications in many areas of computer vision.It is widely used in computer vision tasks such as face detection, face recognition, video object co-segmentation. It is also used in tracking objects, for example tracking a ball during a football match, tracking movement of a cricket bat, tracking a person in a video.

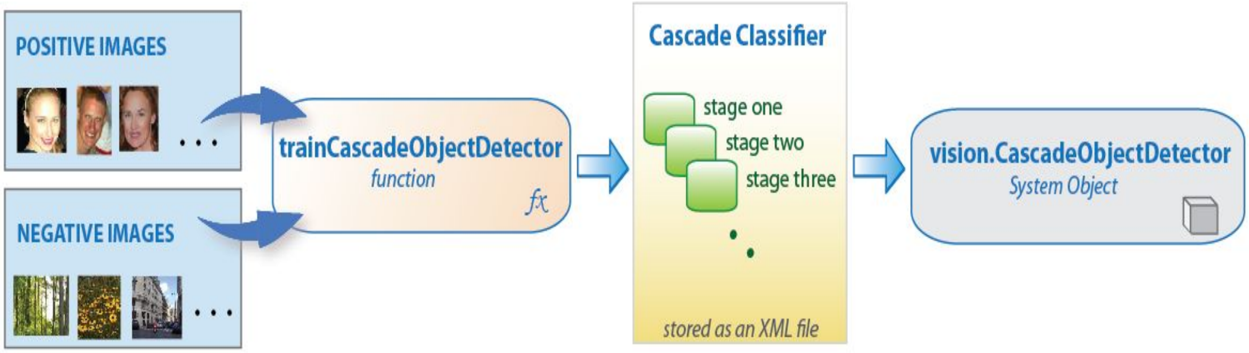
DIFFERENT METHODS OF OBJECT DETECTION :

1) HAAR CASCADE CLASSIFIER :

It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.
Initially, the algorithm needs a lot of positive images (images containing the object to be detected) and negative images (images without the object to be detected) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.



Image



For making classifier ,
1)Cascade Trainer GUI- We need to create a folder of negative images and positive images, and then feed them to this gui. We can set the number of stages as per our requirement.

2) Using command Line- We train the classifier in command prompt .

2) YOLO (You Look Only Once) METHOD :

You only look once (YOLO) is an object detection system targeted for real-time processing.

First, it divides the image into a 13×13 grid of cells. The size of these 169 cells vary depending on the size of the input. For a 416×416 input size that we used in our experiments, the cell size was 32×32. Each cell is then responsible for predicting a number of boxes in the image.

For each bounding box, the network also predicts the confidence that the bounding box actually encloses an object, and the probability of the enclosed object being a particular class.

Most of these bounding boxes are eliminated because their confidence is low or because they are enclosing the same object as another bounding box with very high confidence score. This technique is called non-maximum suppression.

BENEFITS OF YOLO :

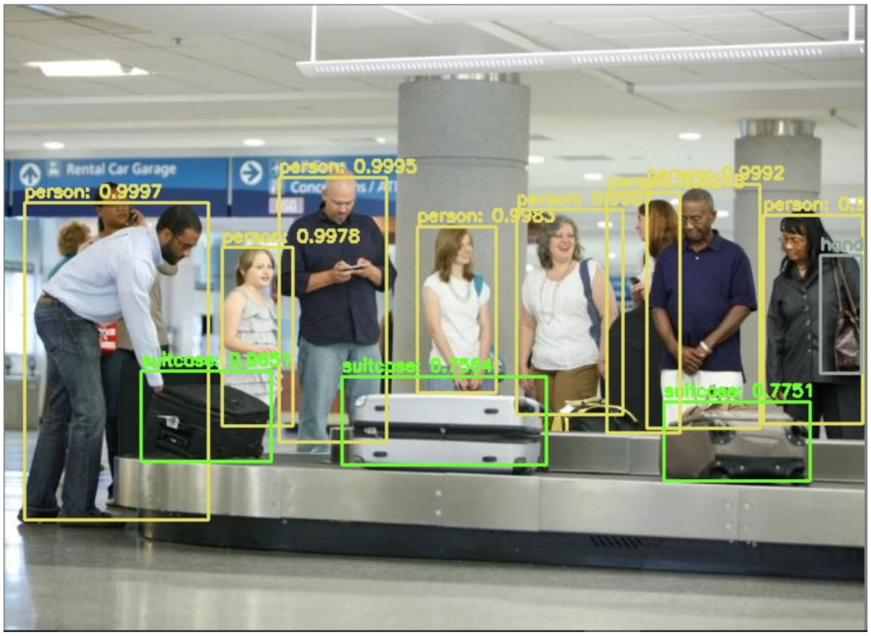
1)Fast. Good for real-time processing.

2) Predictions (object locations and classes) are made from one single network. Can be trained end-to-end to improve accuracy.

3) YOLO is more generalized. It outperforms other methods when generalizing from natural images to other domains like artwork.

4) Region proposal methods limit the classifier to the specific region. YOLO accesses to the whole image in predicting boundaries. With the additional context, YOLO demonstrates fewer false positives in background areas.

5) YOLO detects one object per grid cell. It enforces spatial diversity in making predictions.



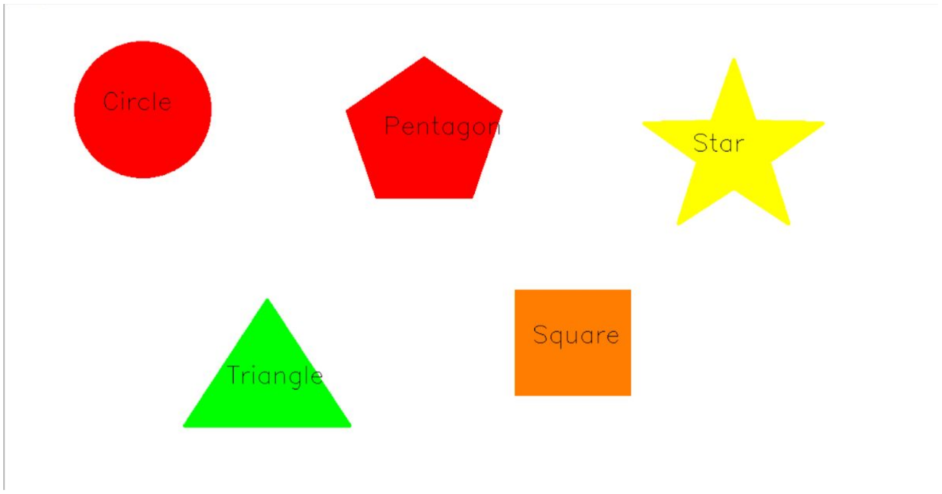
In addition to this , we implemented shape detection and colour detection by HSV method.

1) SHAPE DETECTION

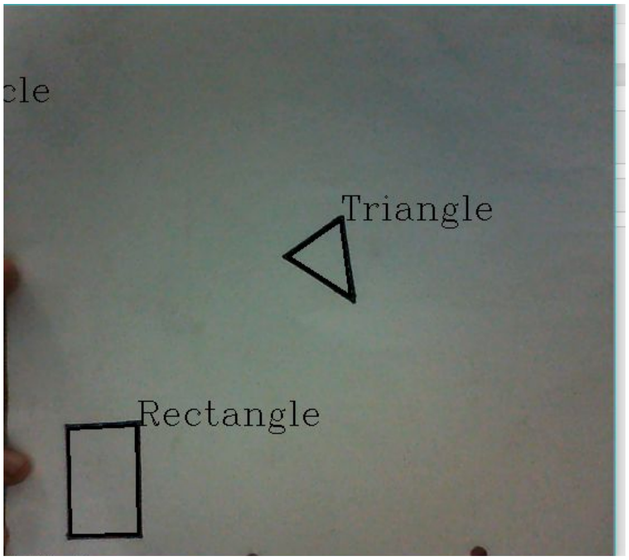
We detected various polygons like triangle, square, rectangle, pentagon, star and circle.

Basically,we identified the number of contours, and calculated the number of edges and detected the shape accordingly. We also displayed the names of the respective polygons at the center of it, by locating its centroid .

SHAPE DETECTION BY TEST IMAGE



SHAPE DETECTION USING LIVE FEED



2) COLOUR DETECTION USING HSV METHOD :

The HSV color wheel sometimes appears as a cone or cylinder, but always with these three components:

HUE

Hue is the color portion of the model, expressed as a number from 0 to 360 degrees:

Red falls between 0 and 60 degrees.
Yellow falls between 61 and 120 degrees.
Green falls between 121-180 degrees.
Cyan falls between 181-240 degrees.
Blue falls between 241-300 degrees.
Magenta falls between 301-360 degrees.

SATURATION

Saturation describes the amount of gray in a particular color, from 0 to 100 percent. Reducing this component toward zero introduces more gray and produces a faded effect. Sometimes, saturation appears as a range from just 0-1, where 0 is gray, and 1 is a primary color.

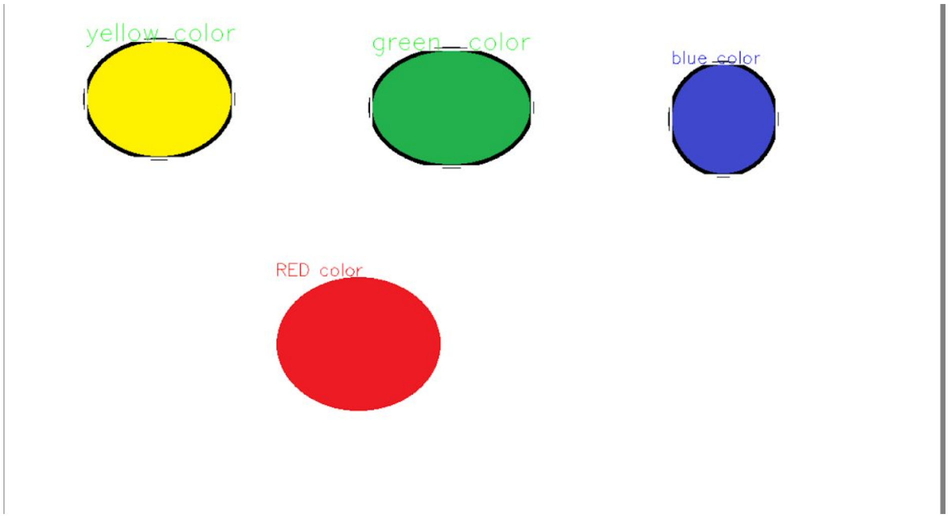
VALUE (OR BRIGHTNESS)

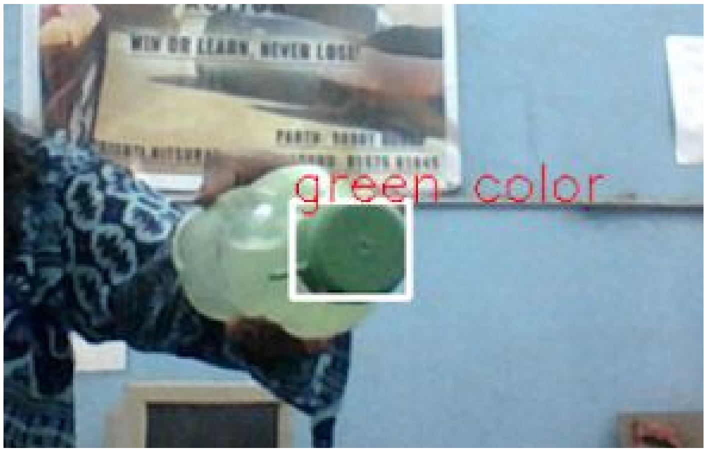
Value works in conjunction with saturation and describes the brightness or intensity of the color, from 0-100 percent, where 0 is completely black, and 100 is the brightest and reveals the most color.

USES OF HSV

Designers use the HSV color model when selecting colors for paint or ink because HSV better represents how people relate to colors than the RGB color model does. The HSV color wheel also contributes to high-quality graphics. Although less well known than its RGB and CMYK cousins, the HSV approach is available in many high-end image editing software programs.

We used the HSV values of the colours to identify them.





3). **HANDWRITTEN DIGIT RECOGNITION**

Basic steps followed are:-

1. Create a database of handwritten digits.
2. For each handwritten digit in the database, extract HOG features and train a Linear SVM.
3. Use the classifier trained in step 2 to predict digits.

MNIST DATABASE

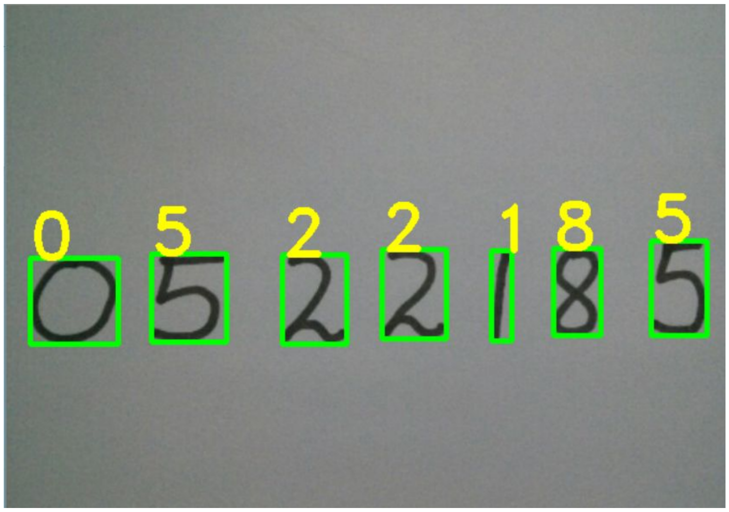
The MNIST database is a set of 70000 samples of handwritten digits where each sample consists of a grayscale image of size 28×28. There are a total of 70,000 samples. It is downloaded from http://yann.lecun.com/exdb/mnist/.The training files are extracted and then given as input.

CLASSIFIER

We will use the sklearn.externals.joblib package to save the classifier in a file so that we can use the classifier again without performing training each time. We use skimage.feature.hog class to calculate the HOG features and sklearn.svm.LinearSVCclass to perform prediction after training the classifier. We store our HOG features and labels in numpy arrays.

PREDICTION

We predict the digits by loading the classifier created in our code.


0
5
2
2
1
8
5