CSS5330 Pattern Recognition and Computer Vision

Project 3: Real-time 2-D Object Recognition

Francis Jacob Kalliath

1. A short description of the overall project

In this project I learned about how 2-dimensional object detection works. Here the objects are taken from scratch then the image is thresholded and cleaned using dilation and erosion. The training objects are stored in a csv file so that the things from the testing data can be compared. This model can detect 3 items simultaneously and the smaller segments will be ignored. The model also labels the items during the detection. I did object detection in 10 classes which includes: mug, bottle, bowl, lotion, scissor, tiger, power bank, specs and clip.

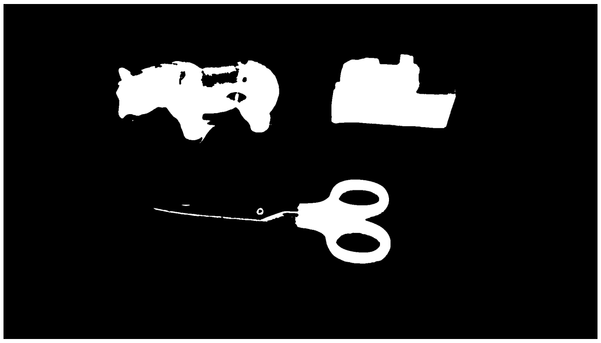
2. Images collected

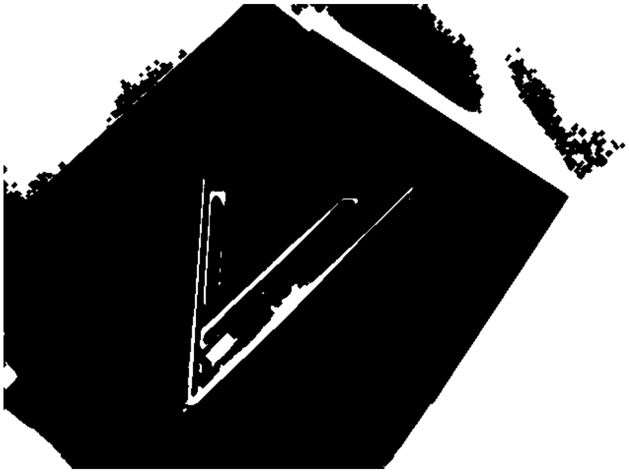
TASKS:

Task 1

Thresholded Image

Convert the captured image to grayscale then we are assigning the value 255 to our required image and 0 to the background.

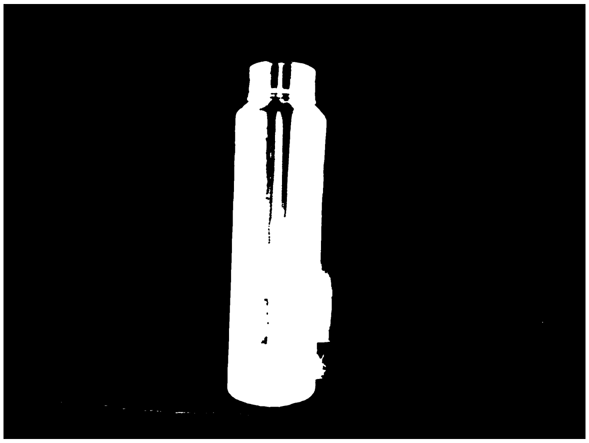


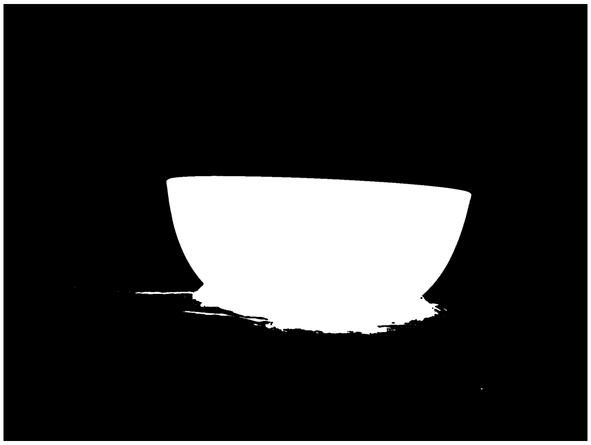


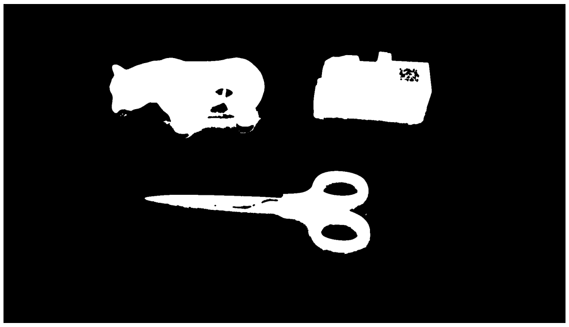
Task 2

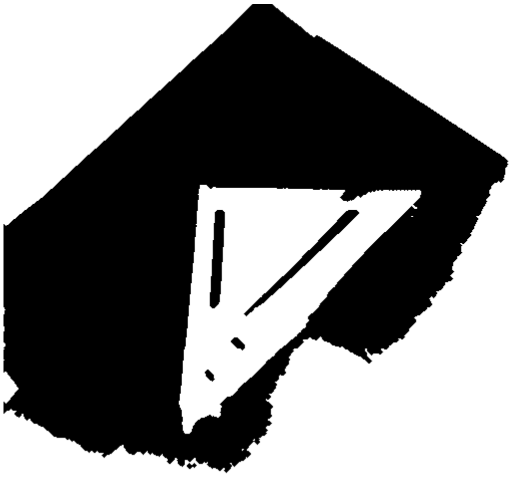
Cleaned Up Image

For this task I am doing two iterations of Dilation and two iterations of Erosion. I also have implemented OpenCV built-in morphologyEx() to obtain better performance for a wide variety of images





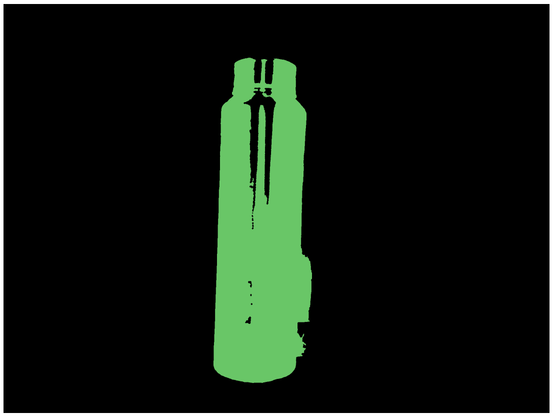
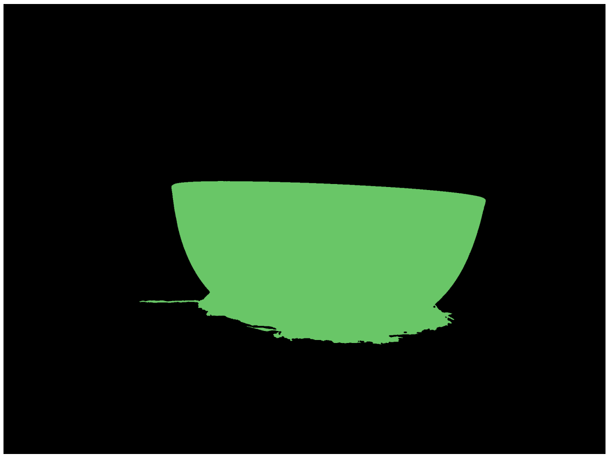




Task 3

Region Map

I am using connectedComponentsWithStats() inbuilt function to identify the regions in the image and then sort the image in accordance with the change in size in ascending order and also an additional check is implemented to make sure that regions that are too small are not considered.

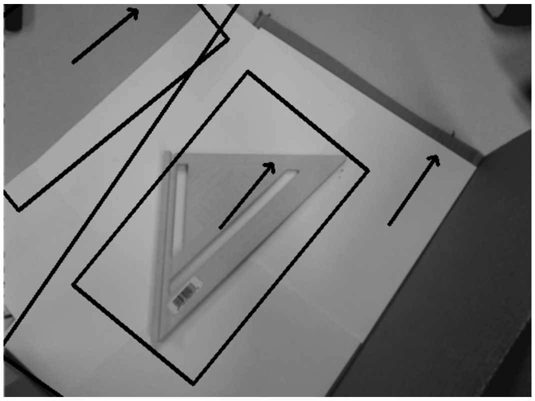



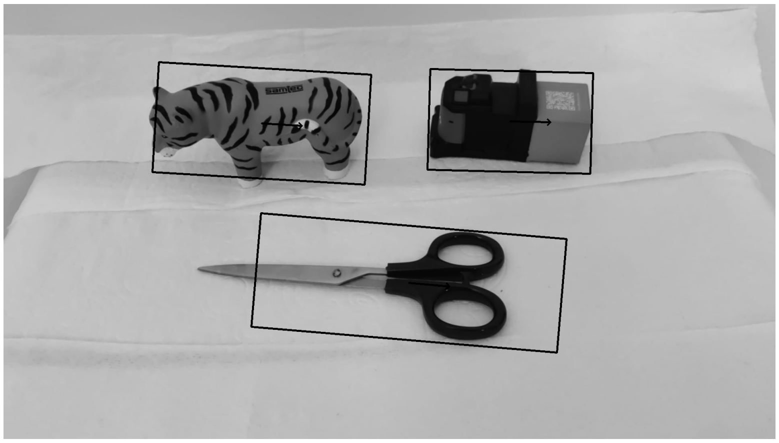
Task 4

Bounding box and axis

The following images show the objects with their bounding box and axis drawn on the image. The regions are calculated by using the openCV functions moments() and HuMoments().

moments() calculates a set of attributes of a region, for example, the centroid, size or area, moments, and central moments. However, none of the moments or central moments can satisfy the requirements of scale, translation, and rotation invariant. Therefore, we calculate HU moments based on the results of the central moments, and the final results are seven invariant moments.



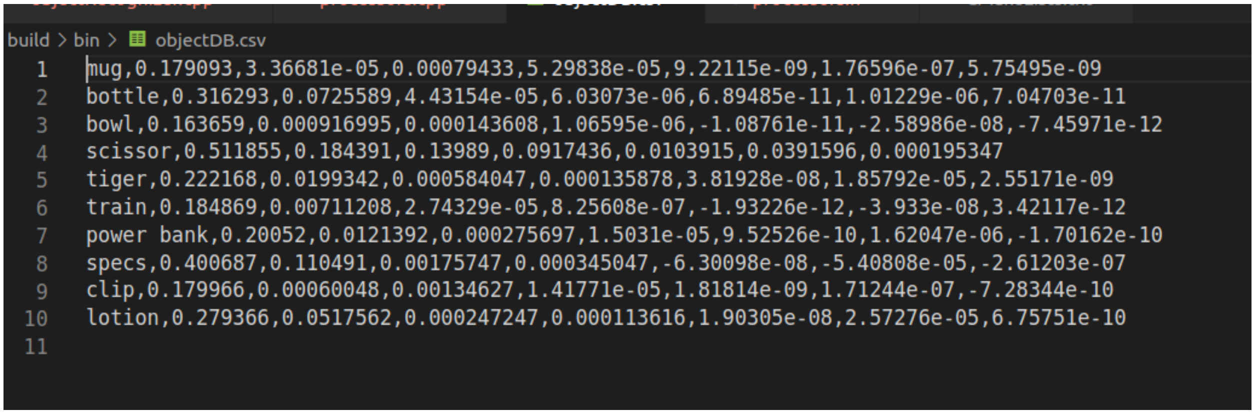


Task 5

Training System

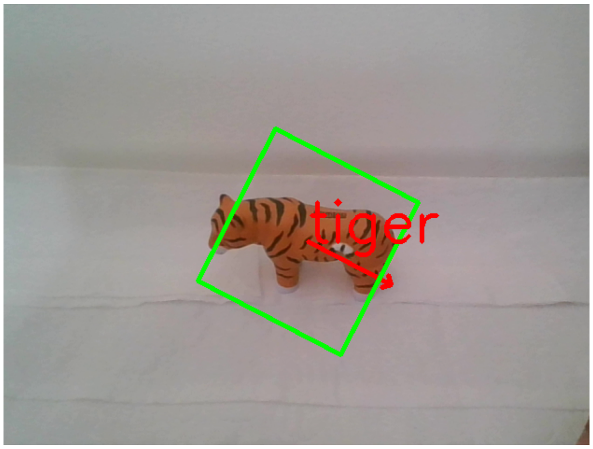
In the training mode we capture the image and store the features in the vector. The CSV file shows the number of classes of data stored in the model.

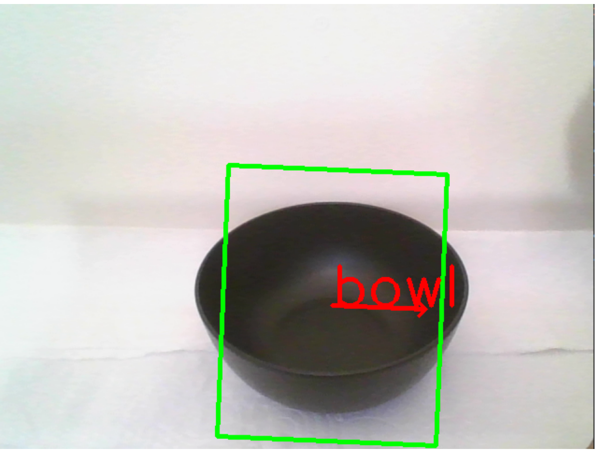
The model will be in training mode when we press ’t’. After the corresponding class is entered then the model automatically moves into prediction mode.

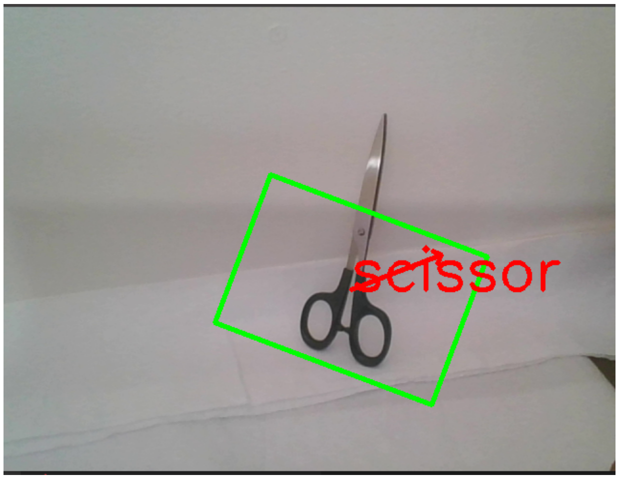


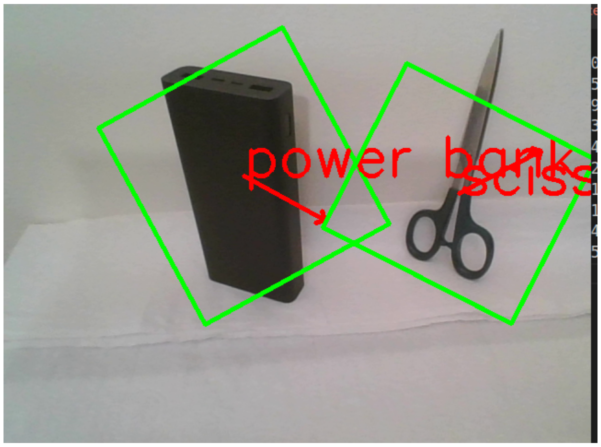
```
mug,0.179093,3.36681e-05,0.00079433,5.29838e-05,9.22115e-09,1.76596e-07,5.75495e-09
bottle,0.316293,0.0725589,4.43154e-05,6.03073e-06,6.89485e-11,1.01229e-06,7.04703e-11
bowl,0.163659,0.000916995,0.000143608,1.06595e-06,-1.08761e-11,-2.58986e-08,-7.45971e-12
scissor,0.511855,0.184391,0.13989,0.0917436,0.0103915,0.0391596,0.000195347
tiger,0.222168,0.0199342,0.000584047,0.000135878,3.81928e-08,1.85792e-05,2.55171e-09
train,0.184869,0.00711208,2.74329e-05,8.25608e-07,-1.93226e-12,-3.933e-08,3.42117e-12
power bank,0.20052,0.0121392,0.000275697,1.5031e-05,9.52526e-10,1.62047e-06,-1.70162e-10
specs,0.400687,0.110491,0.00175747,0.000345047,-6.30098e-08,-5.40808e-05,-2.61203e-07
clip,0.179966,0.00060048,0.00134627,1.41771e-05,1.81814e-09,1.71244e-07,-7.28344e-10
lotion,0.279366,0.0517562,0.000247247,0.000113616,1.90305e-08,2.57276e-05,6.75751e-10

```

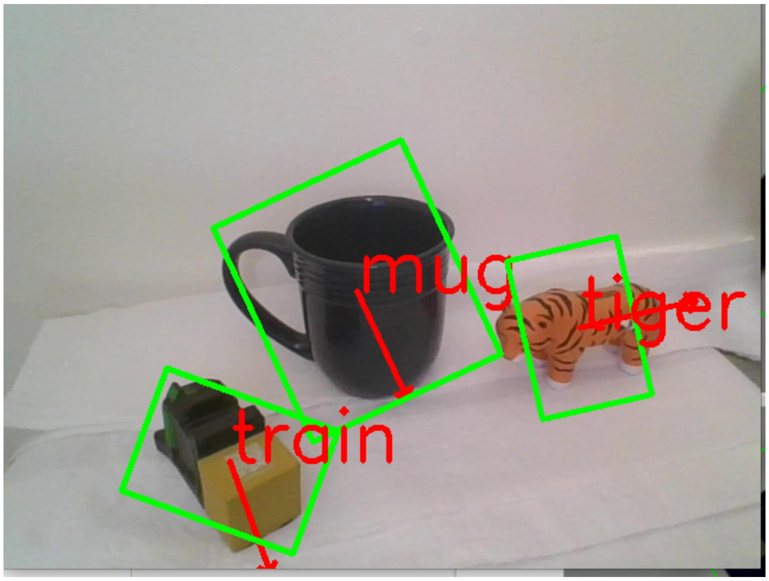
Task 6
Object with their Labels







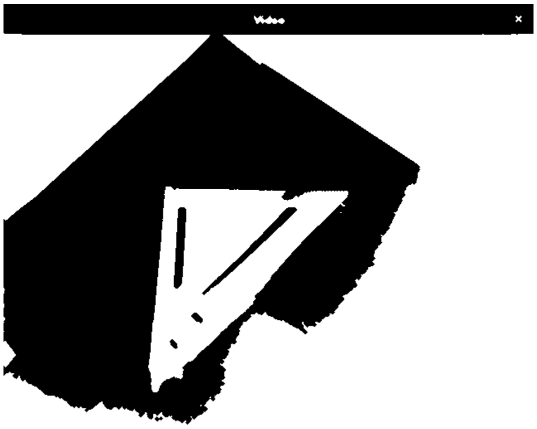


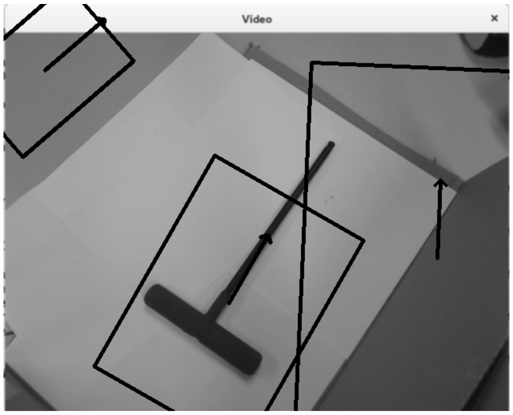


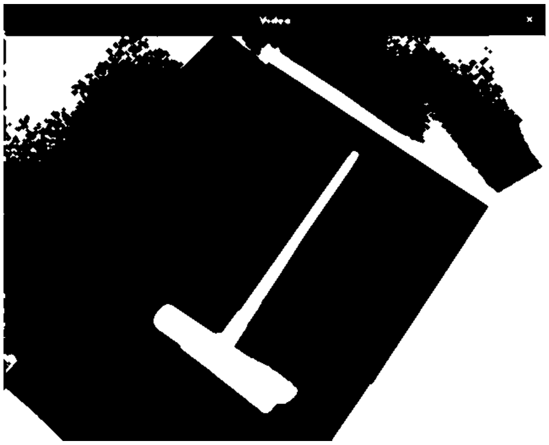
Task 7

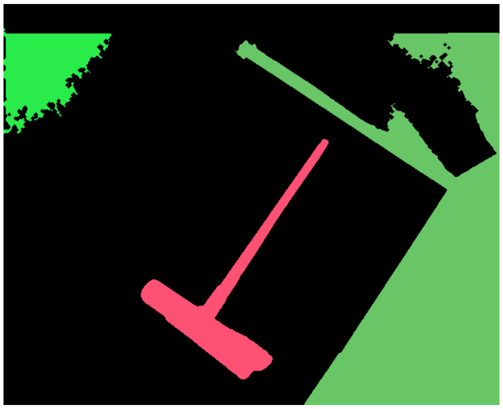
K Nearest Neighbor Classifier

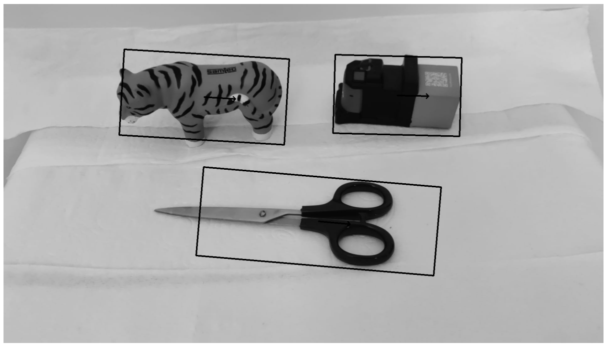
I am using the Euclidean Distance to compare this distance and then I am sorting using the bubble sort method

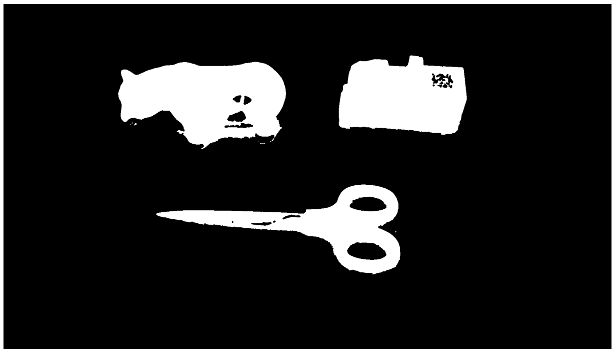


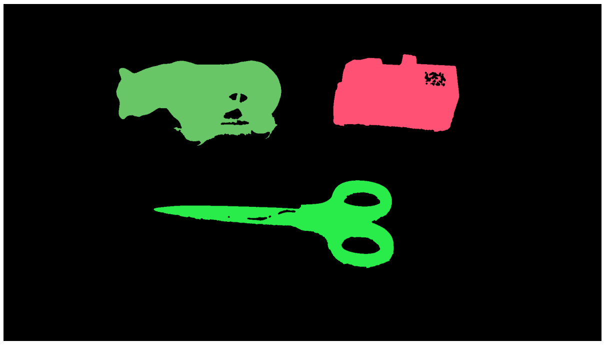

Video

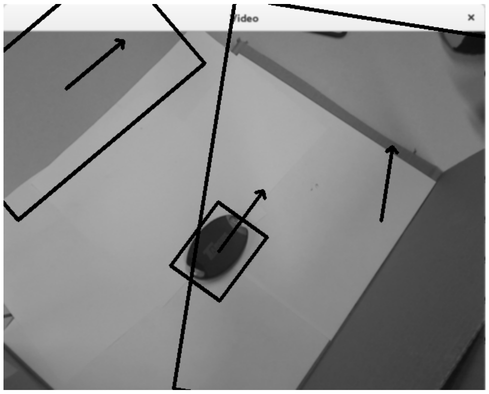

Video

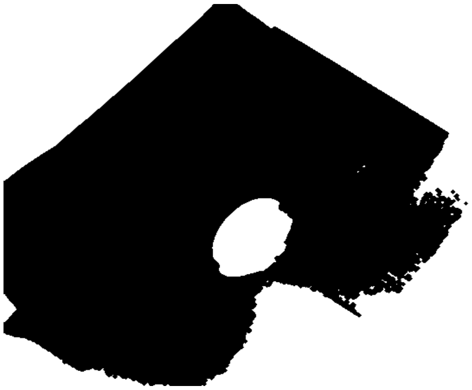


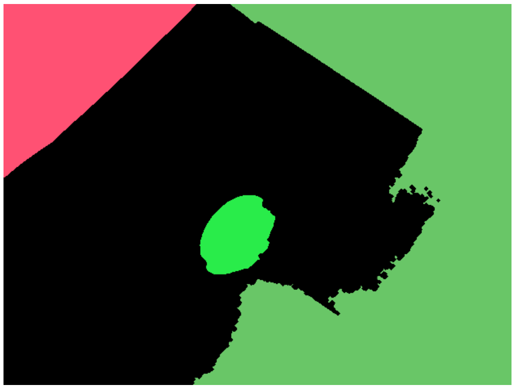












Task 8

Performance of the model

Though I have trained the model to detect 10 objects. For testing purposes I have only used 4 objects and its class for object training and testing of the performance of the model.

From the below confusion matrix it can be proved that the objects are being predicted with good accuracy. The error rate is a bit more between the predictions of the mug and power bank as both the objects are a little bit similar in their solid structure. BUt this can be corrected by adding more training data to the model.

	Mug	**Tiger**	**Power Bank**	**Scissor**
Mug	**10**		**2**	
Tiger	**1**	**7**		**1**
Power Bank	**2**		**9**	**1**
Scissor		**1**		**7**

Task 9

[Video1.mp4](https://northeastern-my.sharepoint.com/:v:/g/personal/jacobkalliath_f_northeastern_edu/EVJRKqOlf7JKgRmV-q13_KcB81ufncfR_gUhF8UwXe0QcQ?e=z3XOiP)

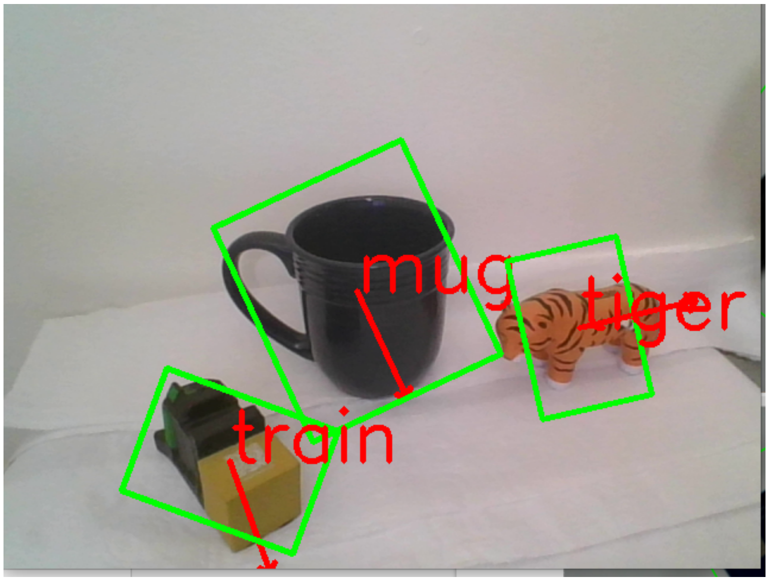
https://northeastern-my.sharepoint.com/:v:/g/personal/jacobkalliath_f_northeastern_edu/EVJRKqOlf7JKgRmV-q13_KcB81ufncfR_gUhF8UwXe0QcQ?e=z3XOiP

Extension 1

The dataset features that are previously started can be retrieved and in case the object is not previously saved only a bounding box is present abound the object and no class name.

Extension 2

The extension has the ability to recognize 3 objects simultaneously. The Image is displayed below



Extension 3

This extension has the capability to identify as many images as required just by updating the class map and each object can have many feature vectors stored in the CSV file. Hence the prediction capability of the model will increase giving more accurate predictions with good accuracy.

3. Reflection of the learnings

In this project I got hands-on exposure to the Objection Recognition algorithm. Doing computation by computing the pixels to vectors and them comparing give a ground to up perspective of what exactly is being done with the image and how the comparison works:
Some of the most important learning from this project are:

1. Converting a image into black and white and then cleaning up the image using dilation and erosion give us an understanding of how to manipulate an image for image comparison
2. Isolating each region to separate parts and then assigning a random color gives me an understanding of image segmentation to obtain objects from the image.
3. Converting the image into a vector and storing the data to a csv file helps me understand how images are converted to feature values and the features are the once that can be compared to find other similar objects.
4. Running the object detection on a live camera gives me the overall perspective of object detection models and how the captured image is compared with the training data to get a good prediction. This project also taught me about some of the major disadvantages and how to take corrective steps to increase the accuracy of models by giving more data to the model.

4. Acknowledgement of the material

- HackerRank
- OpenCV Documentation / Tutorials
- Stackoverflow
- GeeksforGeeks
- W3 School
- Git
- Quora
- Gormanalysis