CS5330 - Pattern Recognition and Computer Vision
(sec-01 spring 2023)

Northeastern University

Project - 3

Sreehari Premkumar

Description of the Project

Project 1 of Pattern Recognition and Computer Vision, has the usage of OpenCV along with C++ to implement an object detector. The process involves thresholding the image to extract the object of interest, followed by applying morphological filters such as grassfire to remove small noise and enhance edges. Moments of the image are then computed to extract features and classify the object. The project is implemented from scratch using C++ without relying on external libraries(except connected Component with stats) for object detection. This approach provides a deeper understanding of image processing techniques and helps in creating customized algorithms for specific requirements.

The project can be extended to include more advanced techniques like machine learning and deep learning to improve the detection accuracy. Overall, the project provides an excellent starting point for understanding object detection and computer vision algorithms.

TASK - 1 - Thresholding

Input: The Input was a video streamed from an android phone using DroidCam app, using the ip address to send the video over wifi network, although this has a lag.


Mug


Pen


Bottle

Darkened using HSV format:

The image was darkened to highlight the object even more :


Mug


Pen


Bottle

The result is more prominent in the red mug, which was darkened.

Threshold:

Applied Thresholding to separate foreground from background :

Mug **Pen** **Bottle**

Task 2 - Clean Up

This was achieved using a grassfire transform to dilate and erode the foreground, so that the image is much more clear, it also removed the watermark of the droidcam along with the timestamp.

Applied Grassfire Transform :

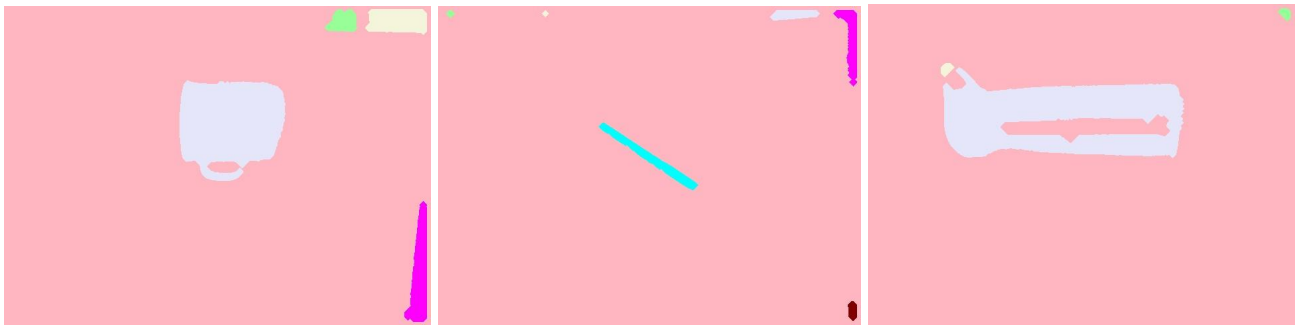
Mug **Pen** **Bottle**

We can see that the pen image was cleaned up and we got the basic shape of the pen and cleaned up streaks around the background.

Task 3 - Segment

The image was run through the function connected component with stats provided by openCV, to get the region map, and then the region map was given different colors according to the id and was displayed.

Applied Segmentation:



Mug **Pen** **Bottle**

Then the region with the biggest area was taken for the next steps.
We could also take multiple regions for multiple object detection in the same image, but it will slow down the process, so I didn't take it for the project.

Task 4 - Computing Features

The Features computed and used as features were invariant to rotation, translation and zoom. First calculated the Oriented Central Moments, then found the bounding box along the longest axis, then next was ratio of height to width of the bounding box, and finally the percentage of the object filling the bounding box.

Output Axis + Bounding box:

Mug **Pen** **Bottle**

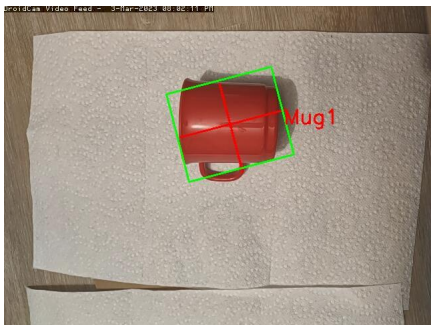
Task 5 - Collect Training Data

On Pressing ‘t’ on the keyboard, it will ask if the current object in the frame needs to be stored into the database along with its computed features. To accept enter ‘y’, to reject enter ‘n’. If it was accepted it will ask a label for the object, once entered it will store and start evaluating with the new changes.
This is showcased in the video as well.

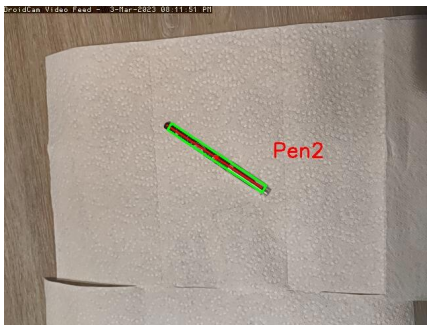
Task 6 - Classify

The classification was done using sum of squared errors (SSE), normalized and scaled features. The lower the SSE the closer the object is to the stored features.

Classified :



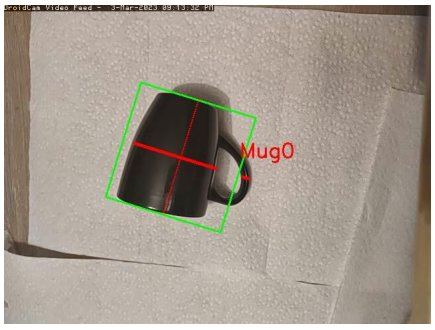
Mug



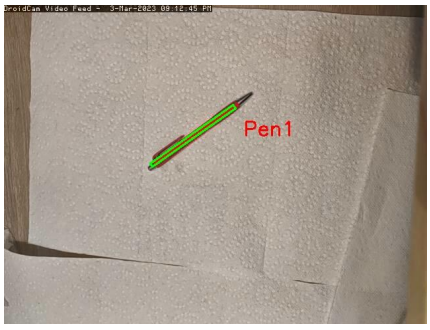
Pen

Bottle

Using a different type & color of same object :



Mug



Pen

Bottle

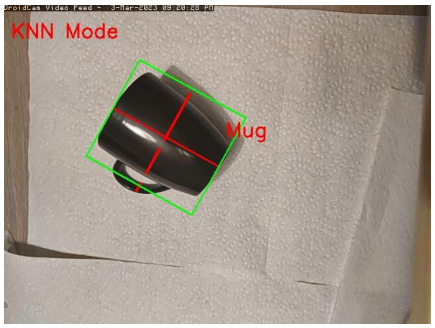
The names of the labels have 0, 1 , 2 at the end as 3 different data of the same object in different angles and lighting was stored for KNN in the upcoming task...

Task 7 - Classification using KNN

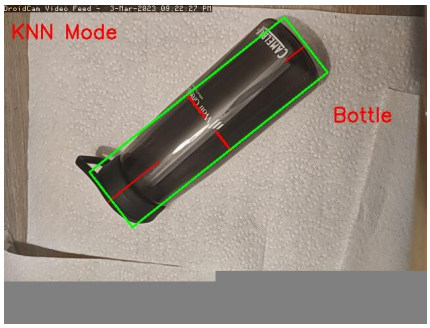
Just Press ‘k’ and Knn Classification will be activated

Knn with K = 3;

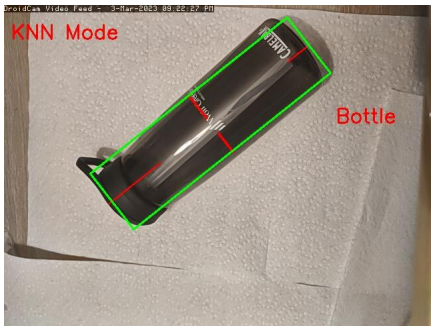
Classified :



Mug



Pen



Bottle

Task 8 - Evaluate Performance

Confusion Matrix :

	Mug (True)	Pen(True)	Bottle(True)
Mug (Classified)	**10**	**0**	**1**
Pen (Classified)	**0**	**8**	**8**
Bottle (Classified)	**0**	**2**	**1**

I would say that the classifier was pretty accurate, it might be due to the fact that there was multiple data of the same object and KNN and also not a lot of different kinds of objects are stored as of now.

The explanation for the Bottle being classified incorrectly is due to the fact that the bottle had a shiny and reflective surface, hence when the lights reflected directly at camera, the part of the bottle was considered to be background reducing it to a mug size, also at different angle the feature distort to that of a pen.

The explanation for the pen being incorrectly classified is that the size of the pen is very small and also a bit reflective. And sometimes after grassfire transform(6 dilation, 12 erosion,6 dilation) the pen gets separated into two smaller objects which might be closer to having features of the bottle.

Video Link :

https://drive.google.com/file/d/1hbfgaYsAeXN8xkHKjLgjbCVslJEGgGFV/view?usp=sharing

Reflection

I have Learned how to use the OpenCV library to read and manipulate images. Also dealing with video as a sequence of images. It was a tremendous experience coding, I had quite the fun. I learned a lot of basic image processing concepts and how to apply them like the filters and different types of pre-processing available to the images. Separation of background and foreground using thresholding, grassfire transform to fill in holes and remove inconsistency.

Then Segmentation into regions and finding features that are invariable with respect to change in angle, location and size. I never had thought that Object detection could be achieved by doing all this. I had previously thought that this would be only possible using neural nets. But thanks to the Professor I feel accomplished.

Acknowledgement

All the Images used were taken by me.

Referenced Stack overflow to help me with debugging the code.

Connected Components with stats function was directly used from their library.
The Remainder was coded on my own.

Referred OpenCV official website to write the code.