CS 5330

Pattern Recognition and Computer Vision

Project 3: Real-time 2-D Object Recognition

Team Members:
Member 1: Vishnu Vardhan Kolla
NUID: 002752854

Member 2: Vidya Ganesh
NUID: 002766414

Overview of the overall project:

We are developing a real-time object recognition system based on the moment features of the object for this project. We make use of a database of object features to perform object recognition. To discriminate between the background and foreground, the colored image from the training set is first converted to a binary image using thresholding. Image segmentation is then used to identify the object's region. The object region's moment features are then calculated.

While classifying an item, various distance matrices and classifiers are used to compare the attributes of the target object with those of the training. This is then used to determine which object is the most similar, which is then used as the label for the target. 15 photos of each of the 10 different things in our dataset are utilized as the training set. For each object, 4-6 photos are utilized to assess the system's performance during testing. Some of the classes and images are as follows:

NOTE: As a result of our **laptop's limitations** and the inability to connect our phones to the computer, we set up our system to accept a directory of photographs rather than a live video stream.

As part of **extensions**:

1. We first added more than the required ten objects to the DB so our system can recognize more objects.
2. Enabled recognition of multiple objects in segmentation.
3. We then experimented with more classifiers and/or distance metrics for comparing feature vectors.
4. We enabled our system to learn new objects automatically by first detecting whether an object is known or unknown, and then collecting statistics for it if the object is unknown and demonstrated how we can quickly create a database for the same.
5. Finally, we explored some of the object recognition tools in OpenCV.

Required Image 1: *Include 2-3 examples of the thresholded objects in your report*.

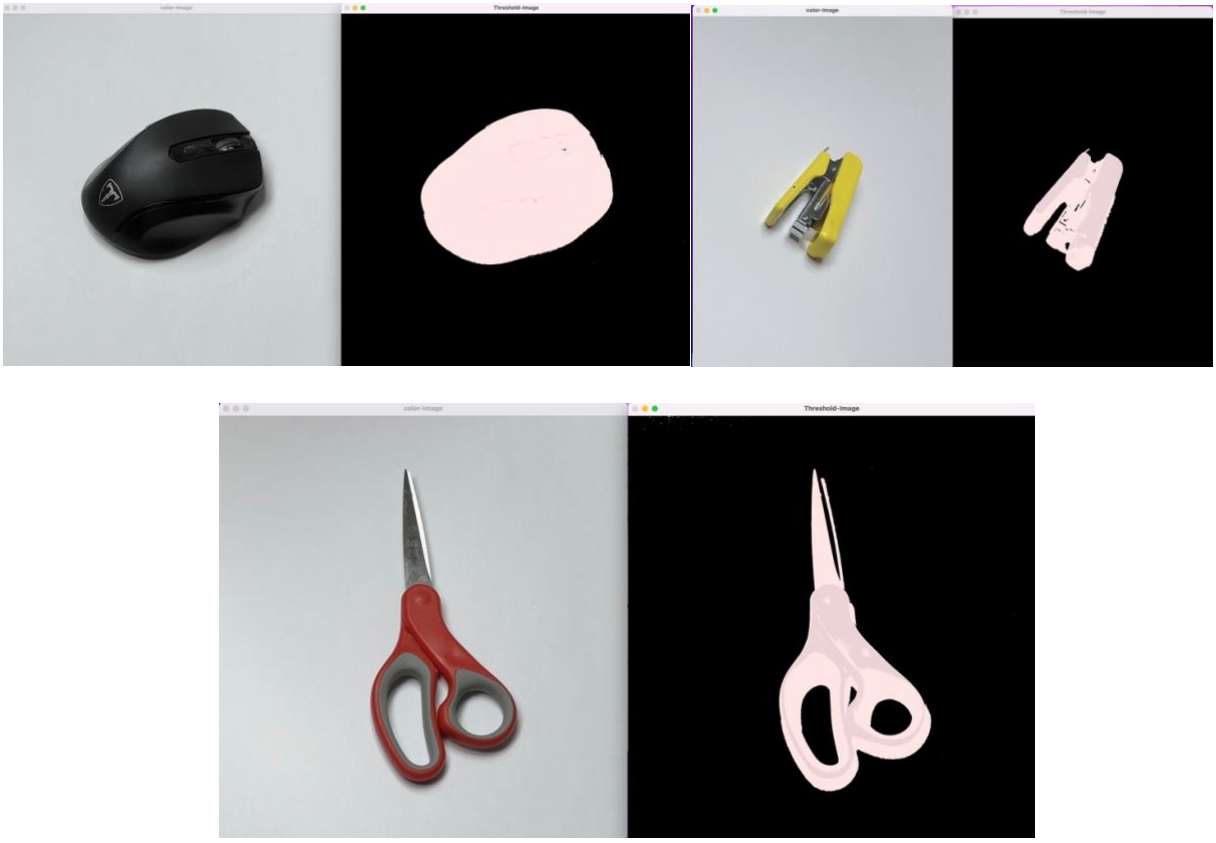

Fig Set 1: From the output it can be observed that after thresholding the original images the thresholded images look like the above ones. Thresholding is usually used to change the pixels of background to black and foreground to white to make the image easier to analyze

Required image 2: Clean up the binary image: *Include 2-3 examples of cleaned up images in your report. Use the same images as displayed for task 1.*

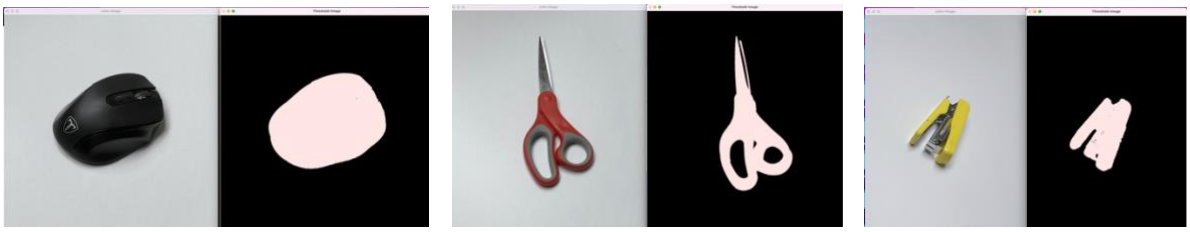

Fig Set 2 : *Examples of cleaned up images from the dataset are shown above.*

Required Image 3: Segment the image into regions: *Include 2-3 examples of region maps, ideally with each region shown in a different color. Use the same images as for the prior tasks.*

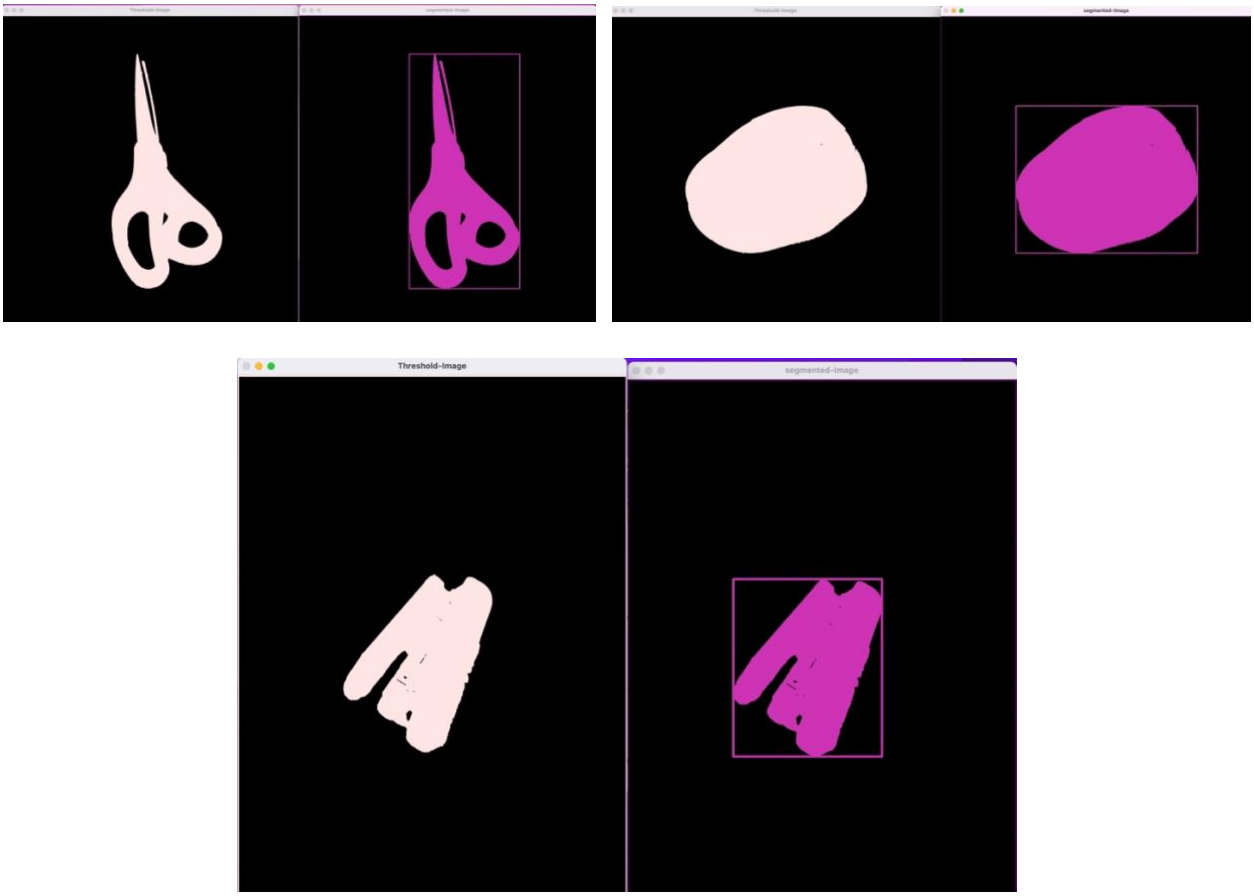

Fig Set 3: Above are the examples of images segmented into regions.

Required Image 4: Compute features for each major region: *Include 2-3 examples of regions showing the axis of least central moment and the oriented bounding box. use the same images as for the prior tasks.*

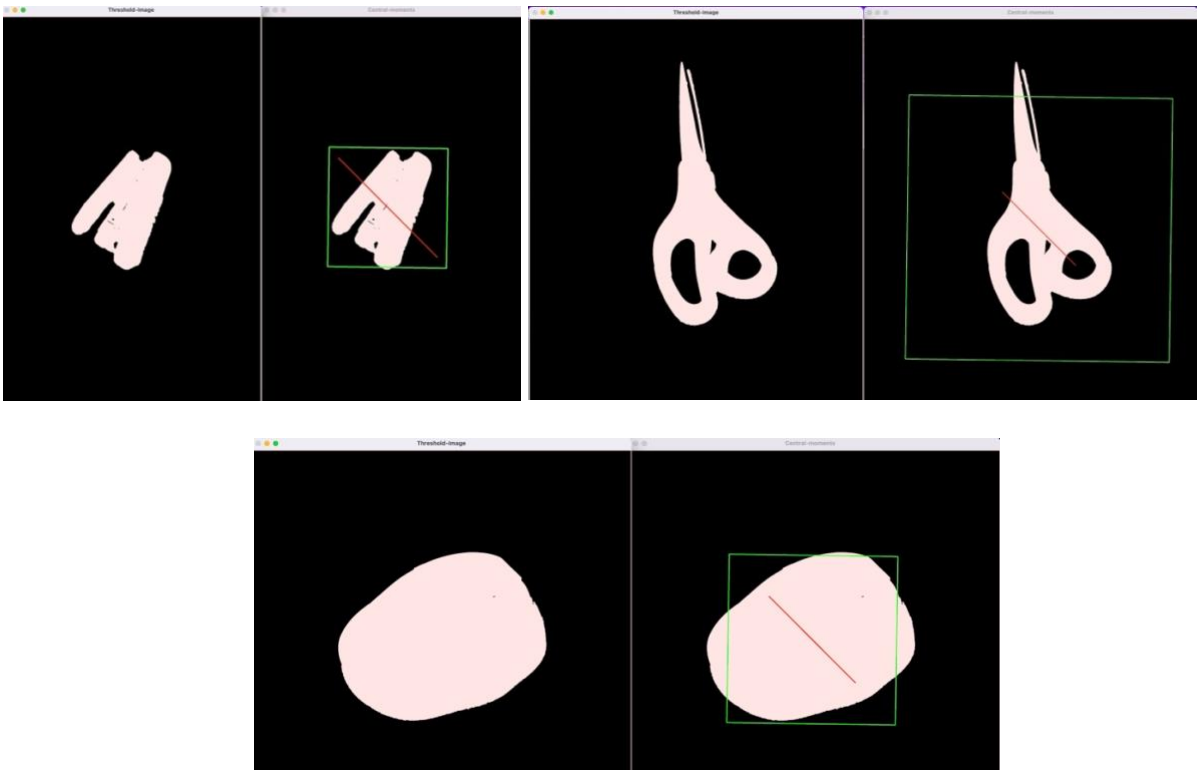

Fig Set 4: *Images above show their least central moment and their corresponding oriented bounding box.*

Task 5: Collect training data : *Explain in your report how your training systems works.*

1. We created a new file training.cpp which takes in a directory of images as an input and scans through all the classes and their corresponding images.
2. The feature vectors for each of them is then computed using different translation, scale and rotation invariant metrics and store them in the csv which is then used as the feature database.
3. The database stores the features of images belonging to 10 different categories.

The 10 classes we used are as follows:

1. Mouse (Computer Mouse)
2. Stapler (Paper Stapler)
3. Headphone
4. Clip
5. Phone

6. Scissors
7. Eyeliner
8. Comb
9. Cup
10. Sunscreen

Required Image 6: Collect training data : *include the results from at least one image of each object in your report with the assigned label clearly shown.*

Correct Classifications v/s incorrect

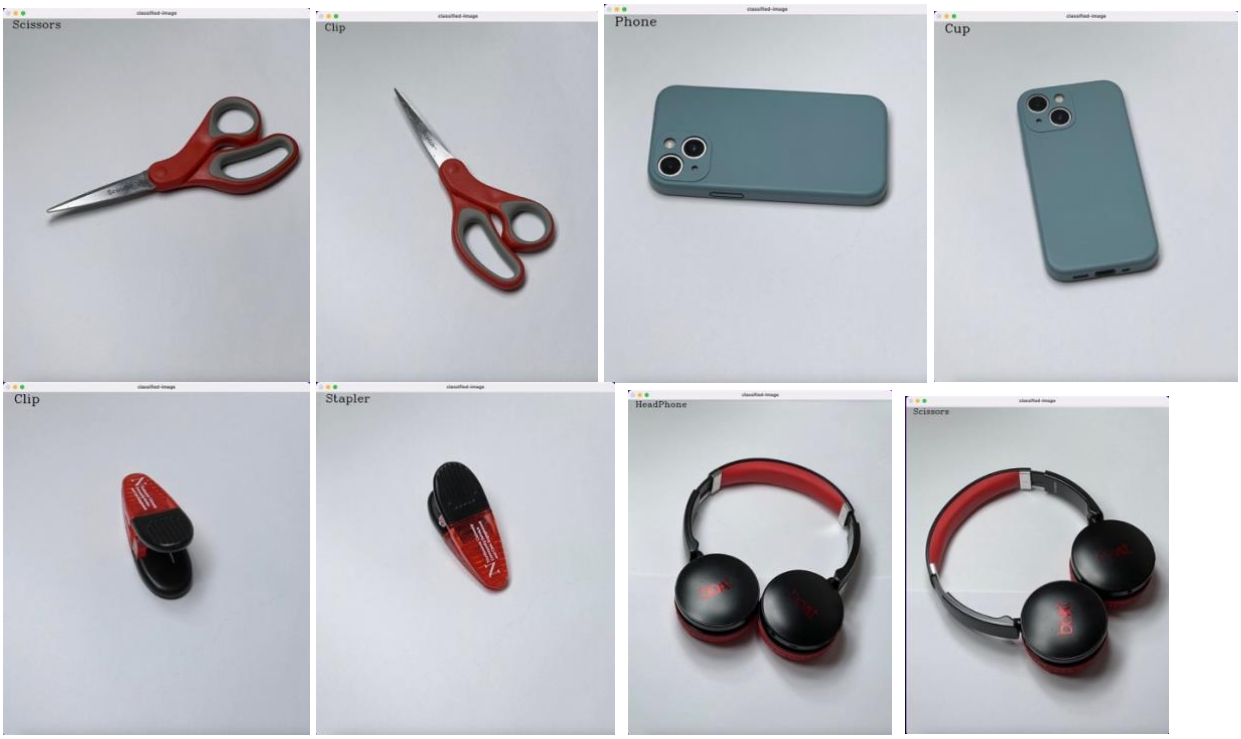

Fig Set 6: Correct Classifications v/s incorrect classifications of images in the database. At the left corner you can find the class label result of the classification for every image.

Task 8: Evaluate the performance of your system

From the test data folder with 5 images per object, we evaluated performance. The accuracy was computed with scaled Euclidean distance. Below is a confusion matrix created using the evaluation results from the experiment.

	Headphone	SunScreen	Cup	Stapler	Comb	Mouse	Phone	Scissors	EyeLiner	Clip
Headphone	2		1		1			1		
SunScreen	1	1							1	2
Cup			1		1		1	1	1	
Stapler		1	1	1		1		1		
Comb	1						4			
Mouse				1		1		1	2	
Phone	1			2			1		1	
Scissors			1		2			1	1	
EyeLiner	1	1		1					2	
Clip	1		1			1	1		1	

Task 9: Capture a demo of your system working

Video link demonstrating image processing steps:

https://drive.google.com/file/d/1_jimWT6xyBzeY0CtwC0taBEFOOGhMKcx/view?usp=sharing

Video link demonstrating image classifications:
https://drive.google.com/file/d/1BG2JacTbi4zyIGuoSN88zbxW7lbnVubP/view?usp=sharing

Extensions

Extension 1: We first added more than the required ten objects to the DB so our system can recognize more objects.

Fig Set Ext 1: Above are the other ten objects used for recognition.

	HP	SS	Cup	S	Co	Mo	Ph	Sc	EL	Clip	OSc	Oil	GP	AP	P	C	Ch	W	D	B
Headp	1				1			1		1				1						
SunScr	1			1					1			1			1					
Cup			1		1		1							1			1			
Stapler		1		1							1			2						
Comb	1			1			1			1	1									
Mouse						1							1			1		1	1	
Phone	1					2	1													1
Scissors			1		1				1		1								1	
EyeLin	1			1										1	1		1			
Clip	1								1			1				1			1	
O-Scis				1	1			1		1				1						
Oil		1			1							2						1		
GreenP			1				1			1	1					1				
AirPods	1							1					1	1						1
Pen		1		1				1									1			
Cap			1		1						1	1				1				
Chapst			1				1			1					1				1	
Wallet				1					1			1			1		1			
Deodrant		1				1					1			1						1

BlueCap			1							1				1			1		1	

Extension 2: Enabled recognition of multiple objects using segmentation.

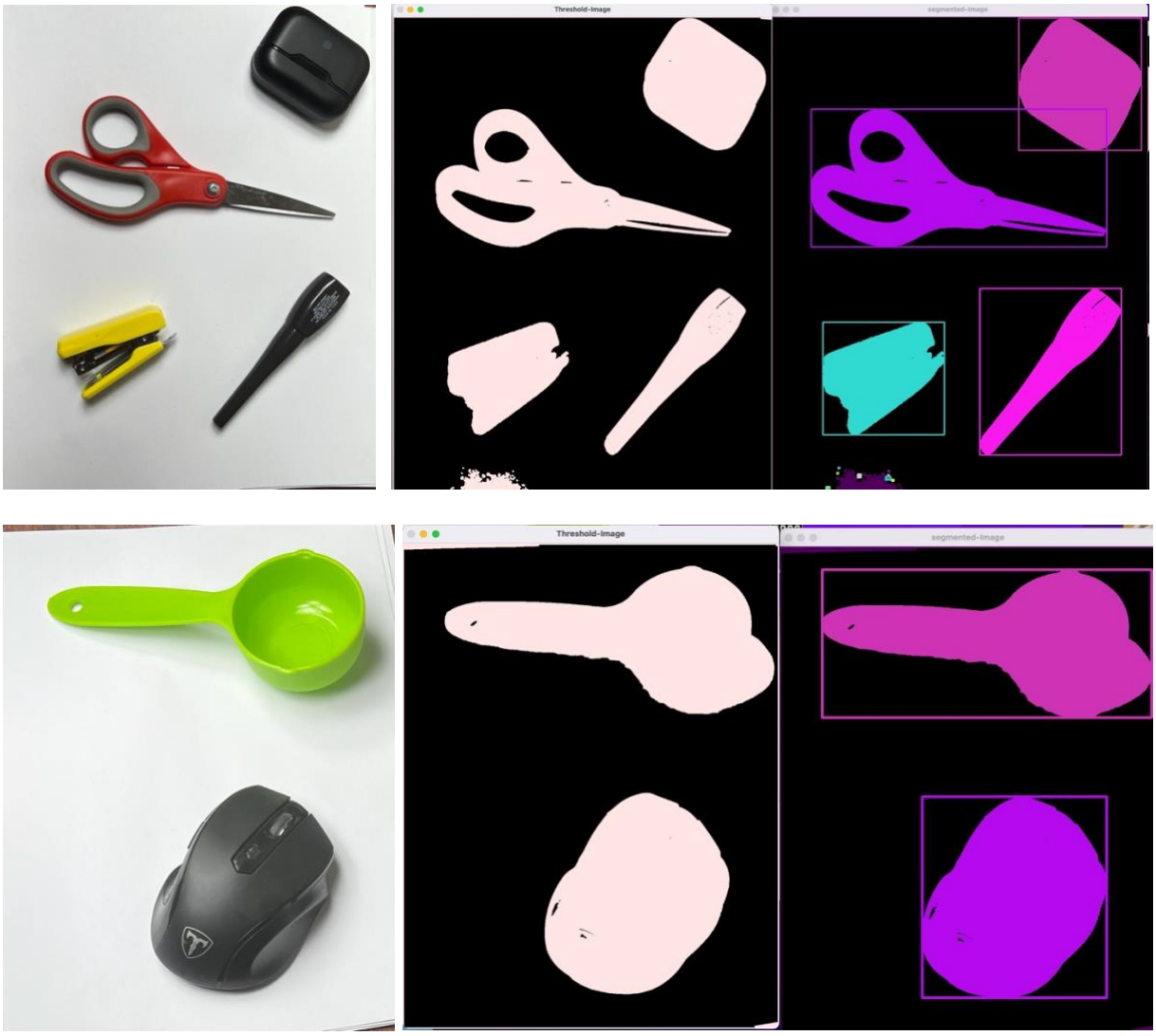

Fig Set Ext 2: *The first image shows an example of multiple objects segmented at once. The second one is another example of the same from the dataset we created. The different colors represent different objects segmented in the layout of the original image.*

Extension 3: Experiment with more classifiers and/or distance metrics for comparing feature vectors.

We have implemented three more distance metrics which are as follows:

1. Euclidean distance (Accuracy : 18%)
2. Manhattan distance (Accuracy: 12%)
3. Chisquare distance (Accuracy: 5.33%)

But the performance of this distance measures was bad compared to that of scaled Euclidean distances as shown in the confusion matrix in task 8. The low accuracy can be due to several reasons, as most items have almost similar shape. Considering the number of pixels can improve accuracy, and if we consider the color of objects as feature then it can improve much more.

Extension 4: Enable your system to learn new objects automatically by first detecting whether an object is known or unknown, and then collecting statistics for it if the object is unknown. Demonstrate how you can quickly develop a DB using this feature.

We have used 3 * standard deviation to find whether a given test image is known or unknown by first finding the distance to all data points in the database and calculating the standard deviation of the distances. Later we checked if the distance of the current test image with its best match is greater than 3 * standard deviation, in which case we have categorized it as unknown, and the user will get a prompt to enter the label of the image and it will be stored In the csv database.

Video Link: https://drive.google.com/file/d/1Ehl-1PjF0QWdUbG_-f6ovJ4nANlury8A/view?usp=sharing

Extension 5: Explored some of the object recognition tools in OpenCV using cascade filters

The above detection was done using cascades for face detection. Cascades are a set of Haar-like features that can be used to detect specific patterns in an image. These features are created by calculating the difference between the sum of pixel values in two adjacent rectangular regions.

Video Link: https://drive.google.com/file/d/1nu3h0UZ4ynFR4Pz1tCcqP8cCDSv55Kf5/view?usp=sharing

Reflections:

I thoroughly understood the functioning of hue moments for object classification and k-nearest neighbor algorithm. We have also grasped the techniques behind the failure of thresholding in particular situations and why certain thresholding methods are superior to others. Additionally,

we experimented with more classifiers and different distance metrics to perform detection. Overall, we have acquired an enhanced understanding of computer vision.

Acknowledgements:

Professor Bruce, and his lectures were helpful to understand concepts of segmentation, grassfire transform, and moments.

Piazza discussions also helped us better understand certain details necessary for this project

Online Resource References:

1. RotatedRect Class: https://docs.opencv.org/3.4/db/dd6/classcv_1_1RotatedRect.html
2. OpenCV official documentation.
3. https://opencv.org
4. https://stackoverflow.com/
5. https://learnopencv.com/shape-matching-using-hu-moments-c-python/
6. https://answers.opencv.org/question/74482/what-does-humoments-tell-me/