CS 5330 - Pattern Recognition and Computer Vision

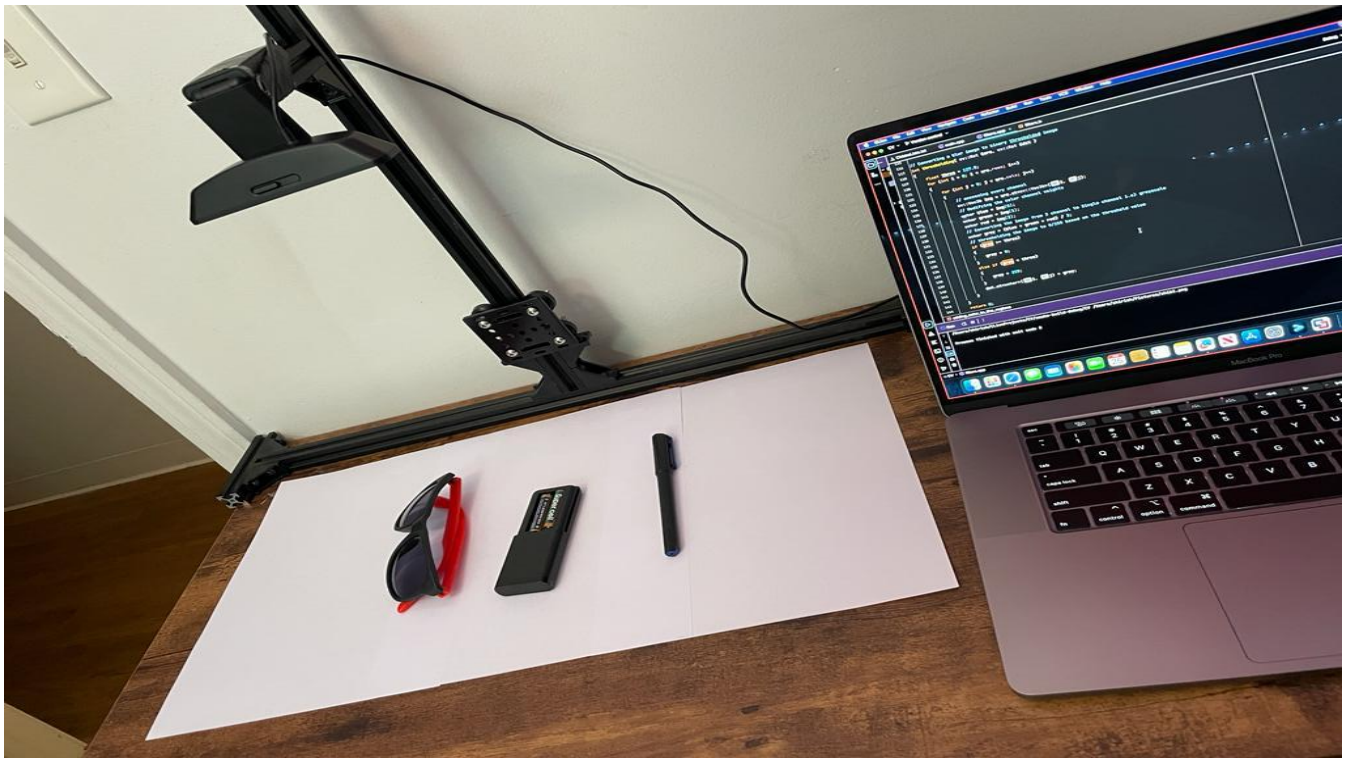
Project 3 : Real-time 2-D Object Recognition

Group Members : Shirish Kishore Kumar (002980081), Soorya Vijayaragavan (002921909)

Introduction:

The project aims to develop a 2D object recognition system that can detect and segment objects placed on a white surface from a camera looking directly down. The system should be invariant to changes in translation, scale, and rotation, and should be able to recognize single as well as multiple objects in real-time from a video sequence. The system will provide oriented bounding boxes and segmented regions to accurately identify the objects in the frame.

Camera Setup:



Task 1: Threshold Input Image

To process the image, the first step is to convert it to grayscale. Next, a Gaussian blur is applied to remove any noise and smooth strong transitions. A threshold is then selected from the range of 0-255 for the grayscale intensity values in the image, which is set to 127.5 in this case. The pixels in the image with intensity values equal to or greater than the threshold are set to 255, while those with intensity values below the threshold are set to 0.

Note: This task is implemented from scratch

Input Video stream

Thresholded video stream

Task 2: Cleanup the binary Image:

In order to eliminate small holes and inconsistencies in the binary image, the process of closing is employed through morphological operations. This technique involves performing dilation followed by erosion. The reason closing is useful in this case is because dilation fills in small gaps around regions in the binary image, while erosion restores the region to its original size after dilation. Coming to our case, we perform 4-connected dilation twice followed by 4 -connected erosion twice.

Note: This task is implemented from scratch

Cleaned Up Image

Task 3: Segment the images into regions

To separate object areas in an image, we utilized a 4-connection connected components analysis. This resulted in labels, areas, and centroids being generated by the OpenCV algorithm for the distinct object regions identified in the binary image. These properties were subsequently employed to assign unique colors to each object region. Here we used different shades of pink color to differentiate the objects.

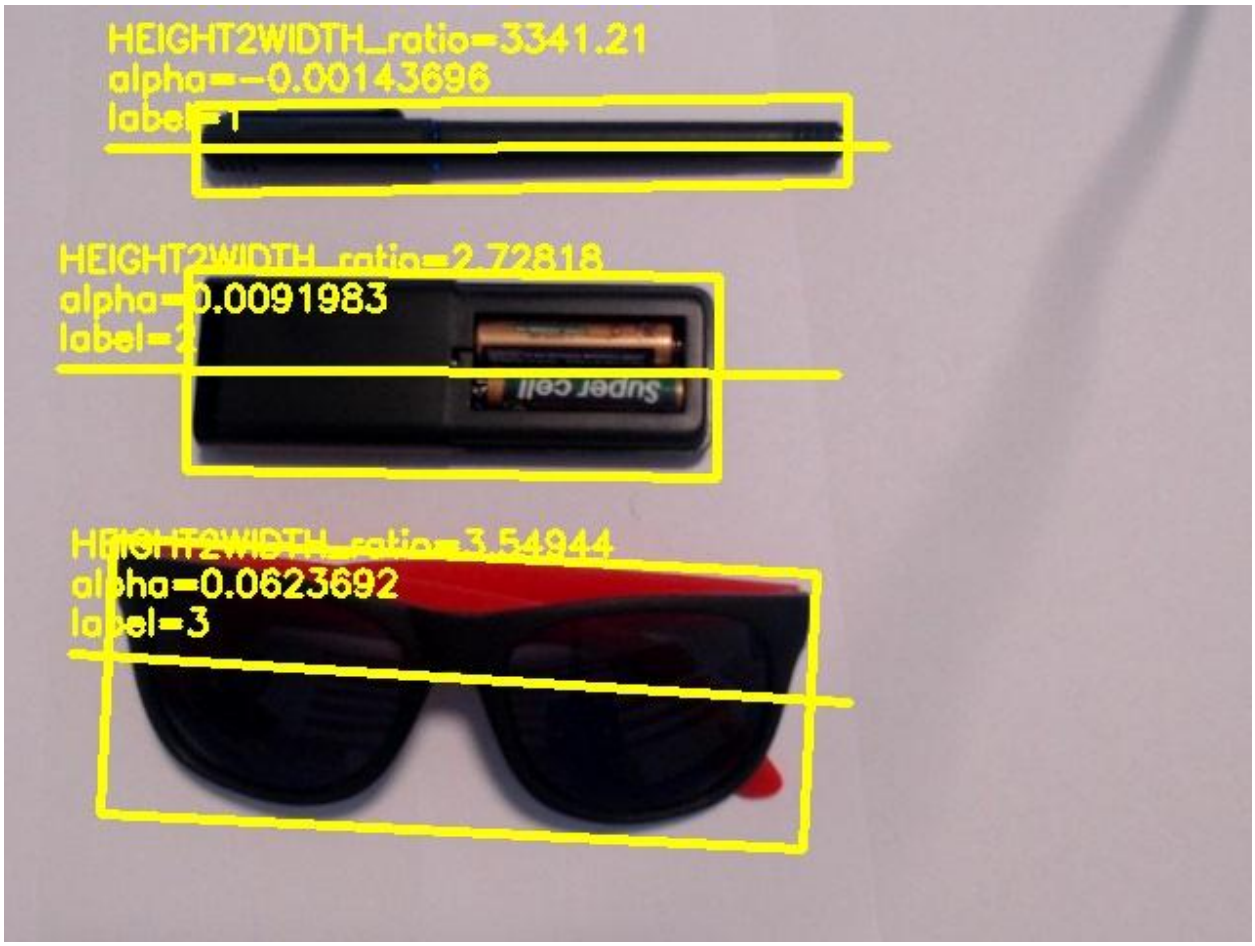
Segmented Objects

Task 4: Compute features for each major region

Features Used : Fill percentage, height to width ratio, MU22alpha, Hu[0], Hu[1], Hu[2], Hu[3], Hu[4], Hu[5], Hu[6]

The OpenCV moments function was used to get moments of each object within the frame. Specifically, we utilized these moments to compute the axis associated with the least central moment and subsequently determined the oriented bounding box that aligns with this axis.

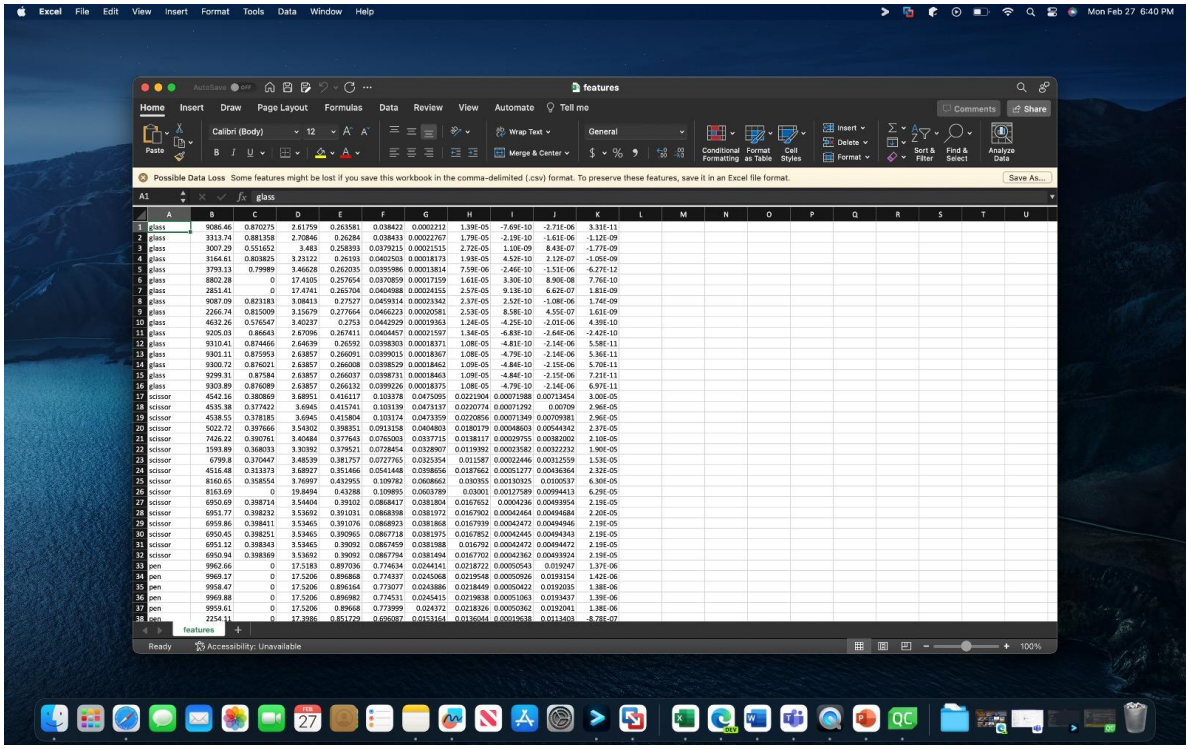
The 10 main features that are calculated are the fill percentage, height to width ratio MU22 alpha which is scale, rotation, translation invariant and the 7 Hu moments. These 7 Hu Moments can be used to describe an image. The first humoment measures the image's overall brightness, while the second and third humoments measure the pixel distribution along the x-y and diagonal directions, respectively. The fourth humoment calculates the difference between the moments of inertia of the foreground and background. The fifth humoment measures the image's skewness, while the sixth moment is rotation-invariant. Finally, the seventh humoment measures the similarity of the image to its mirror image.



Objects with bounding box, axis of least central moment, features

Task 5: Collect training data

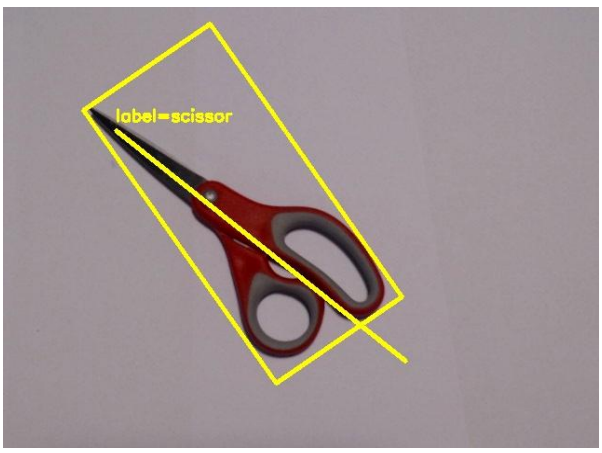
QtWidgets GUI was used to get the label names, In order to perform the training process, we have to press'N/n' to get into training mode, it will display the label number for all the objects in the video, press the label number in the keyboard example: '1' for naming the label 1 object, you can also do for multiple objects that are placed one by one. Once done with training press 'W/w' to save the values in the CSV file.

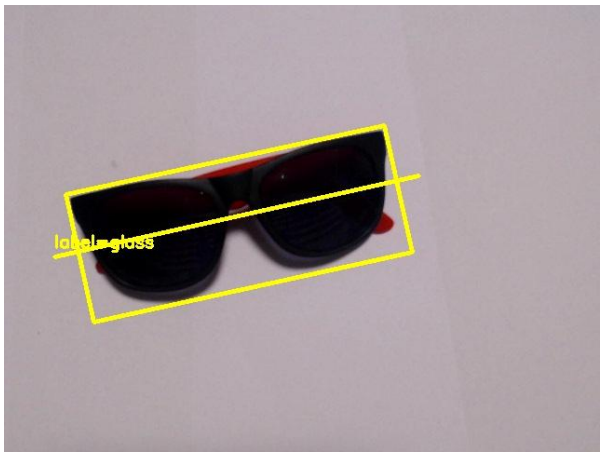


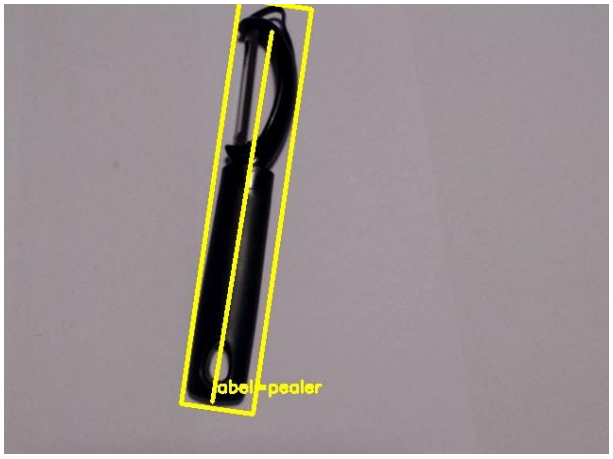
Features of each object is saved in a CSV file

Task 6: Classify new images

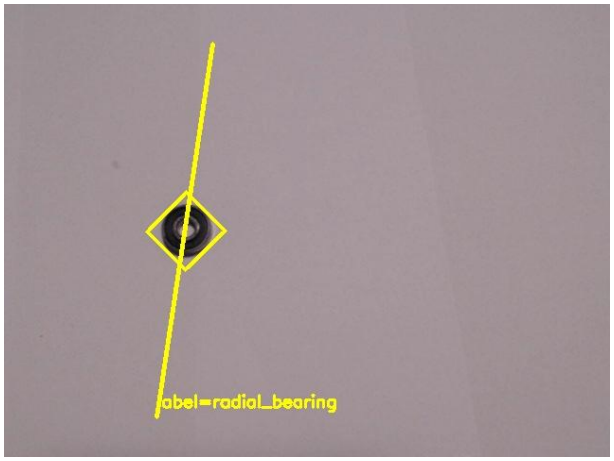
We calculated scaled euclidean values since the features had different units and used them to classify the image.

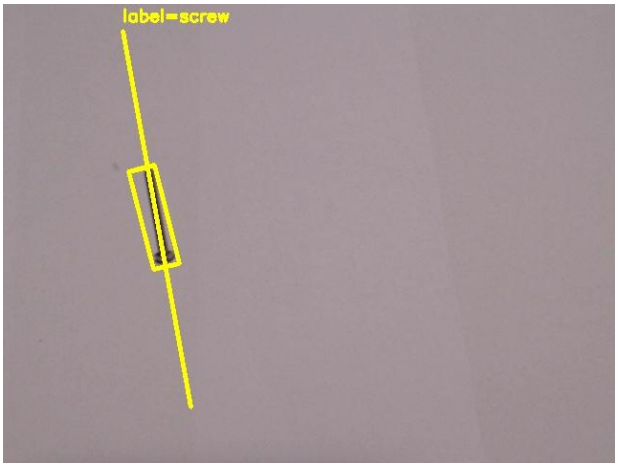


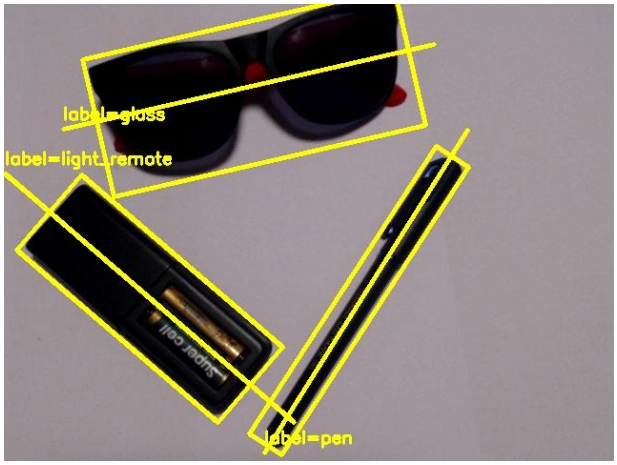


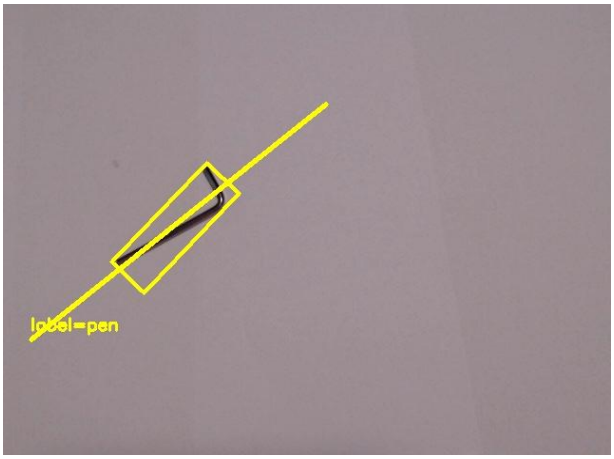

label=pealer

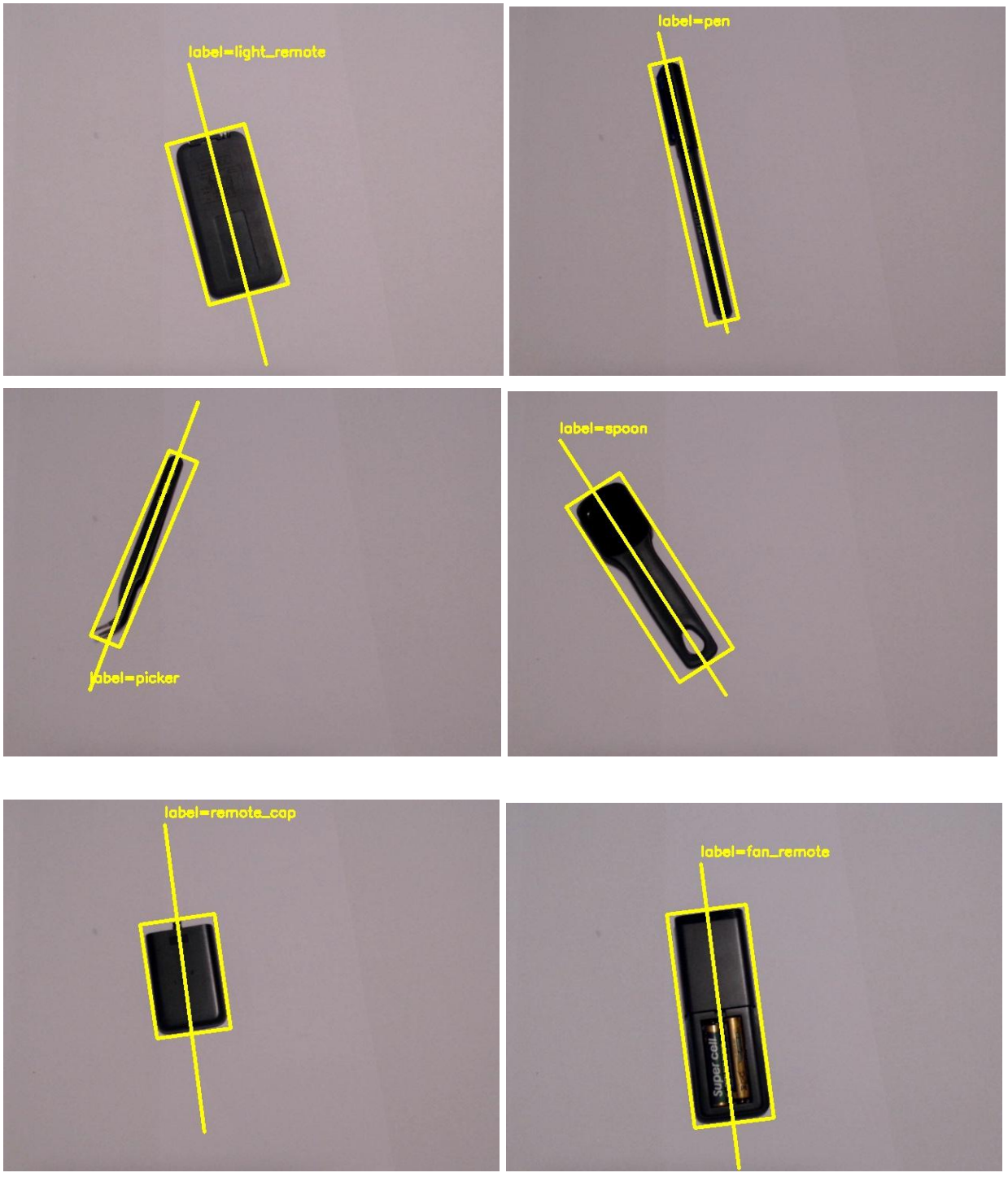

label=xbox_controller


label=radial_bearing


label=screw


label=glass
label=light_remote
label=pen


label=pen



Task 7: Implement a different classifier

We implemented K-Nearest Neighbors as the different classifier with k = 10

Task 8: Evaluate the performance of the system

Each was checked for classification for about 10 trials, below are the results stated as a confusion matrix.

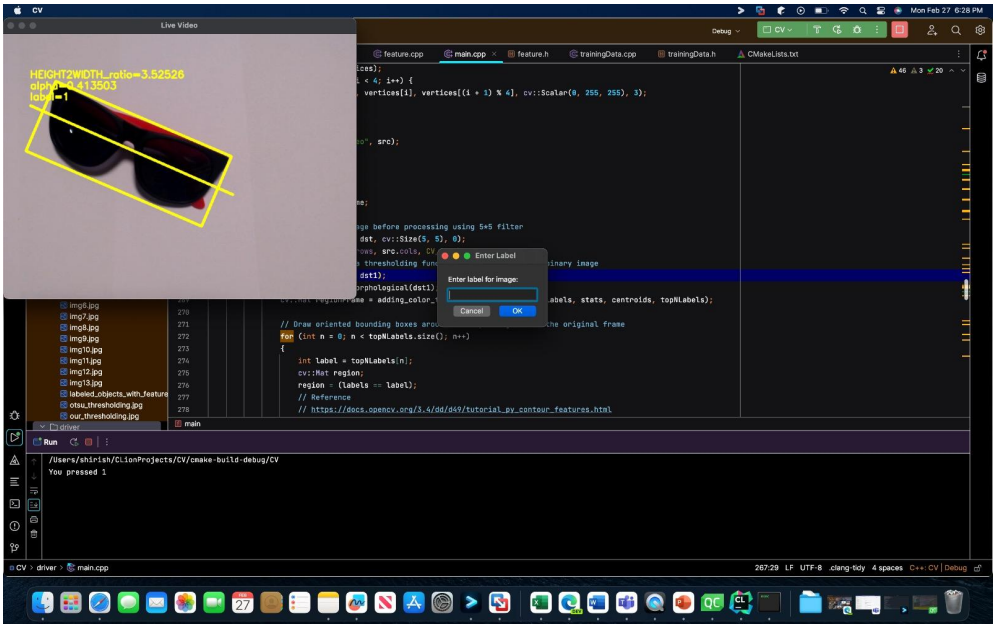
	Glass	Scissor	Pen	Picker	Spoon	Remote Cap	Light Remote	Xbox Controller	Screw	Radial Bearing	Pealer	Fan Remote	Allen Wrench
Glass	**10**												
Scissor		**10**											
Pen			**10**										**8**
Picker				**10**									
Spoon					**10**								
Remote cap						**10**							
Light Remote							**10**						
Xbox controller								**10**					
Screw									**10**				
Radial Bearing										**10**			
Pealer											**10**		
Fan Remote												**10**	
Allen wrench													**2**

Task 9: Capture a demo of your working system

The demo video of the working system is here - working system demo video

Extension 1:

QTwidgets - used this to get the input from the users in a GUI tab in all cases when we get input



Label of the Image is asked as input using

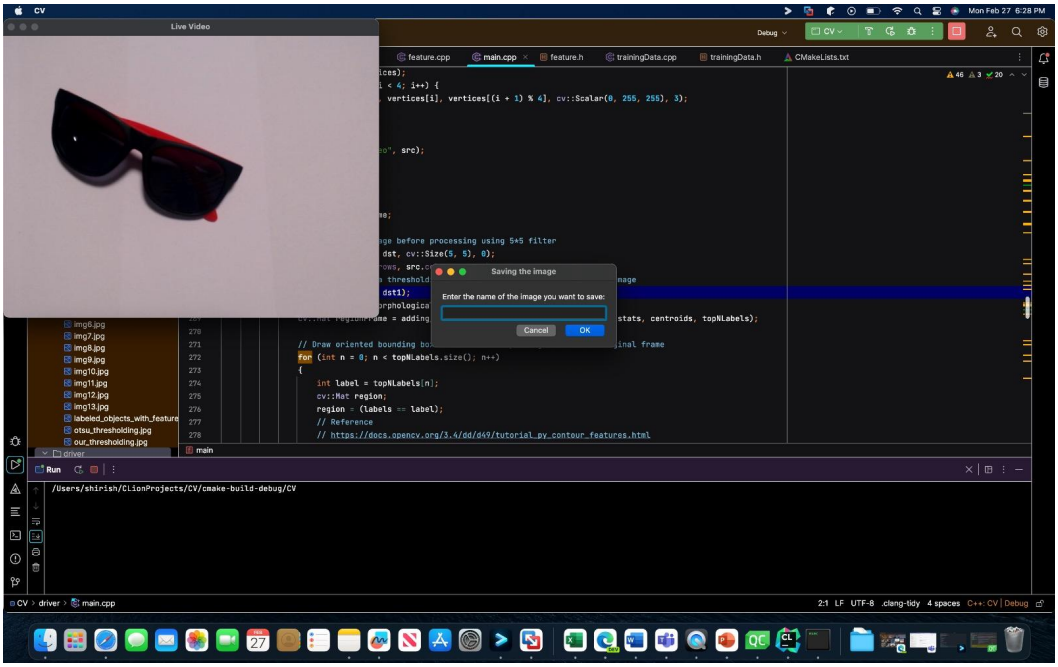


Image name is asked as input to save the image

Extension 2:

13 objects were detected , and the results are displayed in the above task itself.

Extension 3:

Thresholding value was found by otsu method, generates histogram and using it we find threshold values

Extension 4:

We also used counter approximation which is an implementation of douglas- algorithm

Extension 5:
We performed Segmenting multiple objects, labeling multiple objects and detecting multiple objects.

Conclusion:

We used the knowledge gained in our class to perform object recognition , beginning with thresholding, morphological processing, segmentation, Computed the features using the moment() and Humoment(), Classified the objects using KNN and scaled euclidean feature values.

References:

- Connected component analysis
  https://stackoverflow.com/questions/29108270/opencv-2-4-10-bwlabel-connected-components/30265609#30265609
- Structural Analysis and Shape Descriptors
  https://docs.opencv.org/3.0-beta/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=connectedcomponents#connectedcomponents
- otsu thresholding
  https://learnopencv.com/otsu-thresholding-with-opencv/oriented bounding box
  https://docs.opencv.org/2.4/doc/tutorials/imgproc/shapedescriptors/bounding_rotated_ellipses/bonding_rotated_ellipses.html
- Drawing Bounding Box for a rotated object
  https://stackoverflow.com/questions/40404031/drawing-bounding-box-for-a-rotated-object