WARSAW UNIVERSITY OF TECHNOLOGY
Faculty of Mathematics
and Information Science

Introduction to Image Processing and Computer Vision

Project 1:
Image Segmentation with OpenCV

Submitted by:
Amir Ali
317554

Master’s in Data Science

19 December 2021

Table of Content

1. Introduction

Image segmentation is the most significant job in many image processing systems, such as pattern recognition, image retrieval, and small surveillance. The outcome of Segmentation is mainly used for image content understanding and visual entity recognition through the identification of the region of interest. Image segmentation is a technique of dividing a digital image into multiple segments so as to simplify an image. This simplification helps in the study of images for further analysis [1].

In this study, we will do image segmentation of Leaf. And we will separate leaf from the background. There are many types of leaves in different shapes and colors. As you can see some sample below:

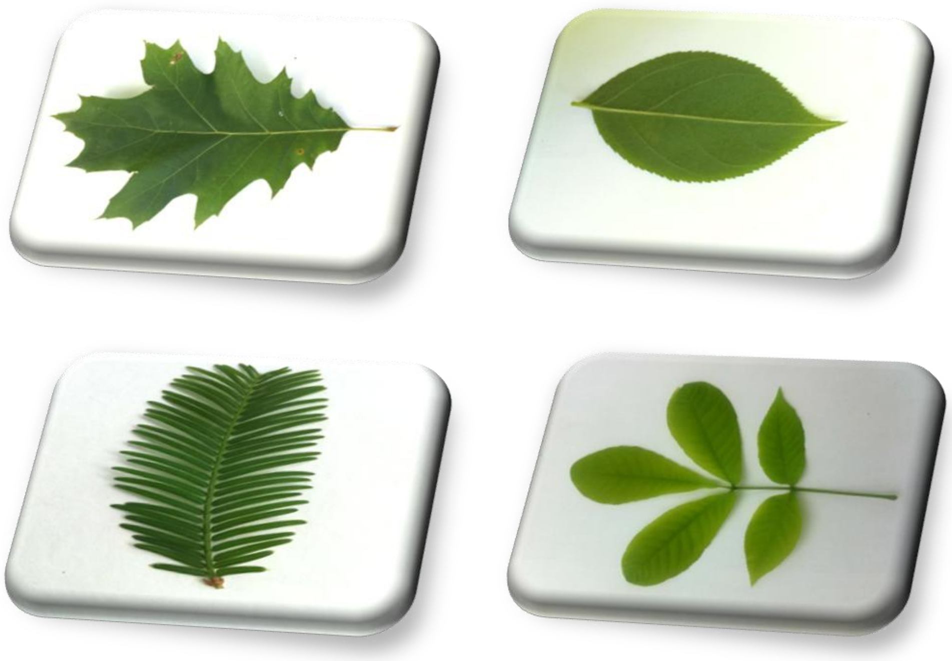

Figure 1.1: Sample of Input Images

2. Literature Review

In this paper [2], they used a fuzzy numerical morphology calculation to differentiate the edge. By contrasting and the results of alternate calculations, such as Sobel and binary morphology, the effectiveness of edge recognition in light of fuzzy mathematical morphology was demonstrated. The edge identification calculation proposed by this paper has points of interest like no threshold required, high invulnerability to noise, texture extraction, and vein extraction. In addition, research is being conducted to determine the area of a tobacco leaf and the length of a vein based on the results of edge identification.

In this paper [3], they introduced an algorithm that consisted of three main phases. To begin, remove the undifferentiated from disease spots (stems and disease spots) from green plants. Second, a gray histogram is used to identify some unusual properties; after that, the divided image is converted into 8-bit grayscale images using single thresholding. Finally, compare the ranges of the diseased areas and the stems, and then segment the linked images using area thresholding. The results of the test show that this calculation is effective in dividing cotton malady spots; the calculation's usual right extraction rate is 94.79 percent.

In this Paper [4], the illness areas were identified using Region Based Segmentation. They developed procedures to choose the best one out of the three primary Regions-based segmentation techniques: Region Growing, Region Merging, and Region Splitting. When compared to mean-shift segmentation approaches, Region growth had the most peaks representing separate regions with the least discontinuous entropy and highest grey level energy, according to timing analysis and Quality Metrics. As a result, Region Growing Segmentation was discovered to be a viable segmentation approach for further processing.

In this paper [5], they proposed an approach for various plant research based on fuzzy edge and clustering segmentation. For various plant leaf acknowledgments and grouping, segmenting the plant from foundation items was a challenging task. Before implementing the proposed technique, pre-processing procedures such as image change, middle channel commotion reduction, morphological operation, and wavelet change have been completed. In light of fluffy limit and grouping systems, the proposed technique for discovering the most homogeneity district in plant leaf photos produced excellent results. Variation in Information, Energy, Entropy, and Evaluation Time were used to evaluate the relative execution of the traditional and new procedures. It was shown that the proposed technique produces satisfactory grouping and acknowledgment results.

3. Problem Statement

The technique of segmenting a digital image into a number of regions of interest is known as segmentation. The purpose of segmentation is to rearrange and transform the representation of an image into something more meaningful and less time consuming to analyze. The result of picture segmentation is a set of areas that, all things considered, cover the entire image, with each pixel in a region being compared to some trademark or registered feature, such as color, intensity, or texture.

In this study, we will use Leaf Dataset for Image segmentation [6]. The dataset contains 300 images of Leaf. And also contains the ground truth images as well which we will use later to determine the accuracy by comparing the result using the IoU method.

In this dataset two different subsets are included:

- leaves_testing_set_1: 150 leaf images for which state-of-the-art methods produce already faithful segmentation results;

- leaves_testing_set_2: 150 leaf images for which state-of-the-art methods partially or totally fail.

4. Description of Solution

First, we convert images into Grayscale.

# convert original image into gray scale

```
image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

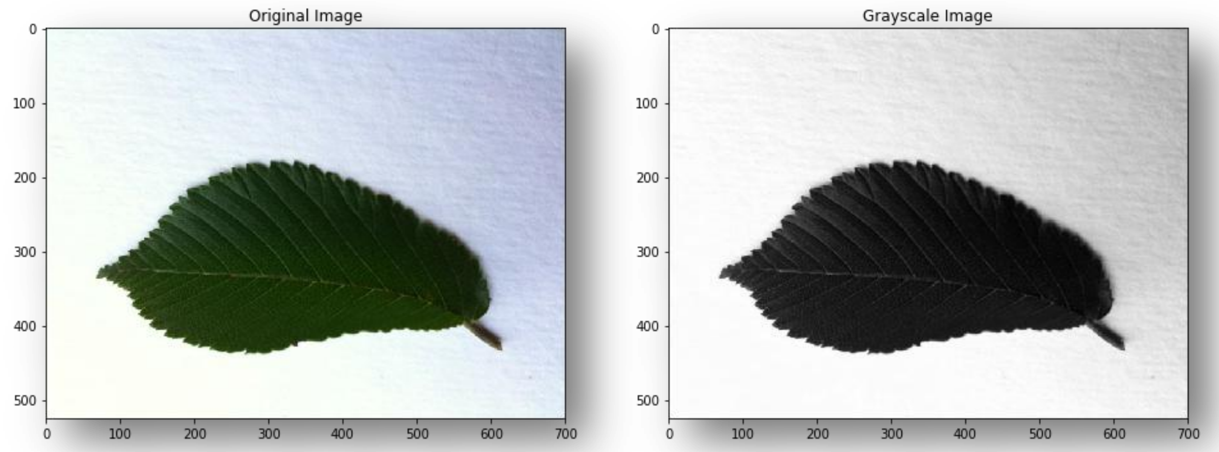

Figure 4.1: Original and Grayscale Image

Then we apply the Basic thresholding method in our case we apply the Binary Inverse Method because we want the leaf in white and the background in black color.

```
#Basic Thresholding
```

Cv2.threshold(source, thresholdvalue, maxvalue, thresh_techniques)

Inverted Binary

$$\mathtt{dst}(x,y) = \begin{cases} 0 & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ \mathtt{maxval} & \text{otherwise} \end{cases}$$

```
_, segmented1 = cv2.threshold(image_gray , 127,255,cv2.THRESH_BINARY_INV)
```

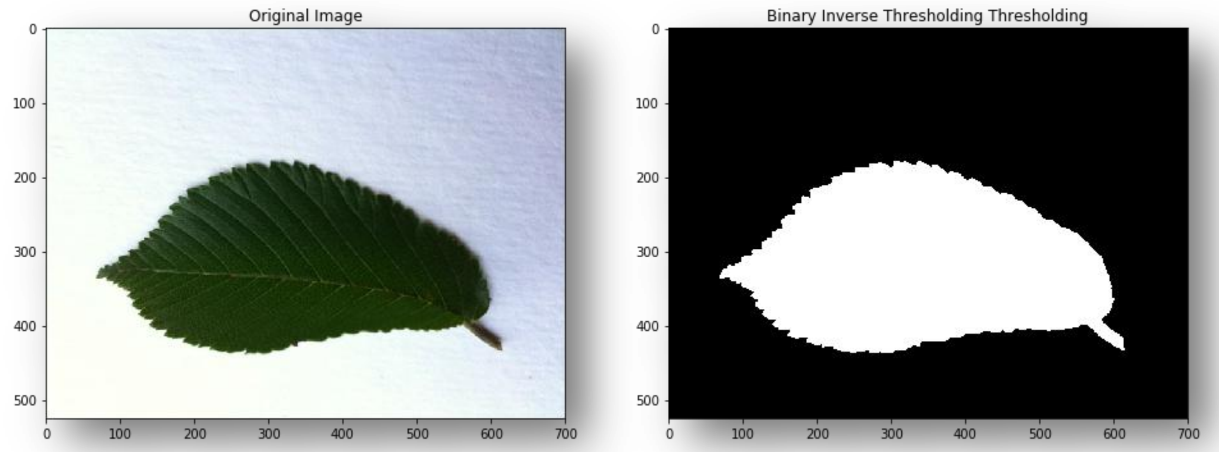

Figure 4.2: Original and Binary Inverse Thresholding

Then we apply find contours method. In that The first one is source image, second is contour retrieval mode, third is contour approximation method

```
# find the contours
contours, hierarchy = cv2.findContours(segmented1, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
largest_areas = sorted(contours, key= cv2.contourArea)
```

```
# Draw contours
                    Cv2.DrawContours(img, contours, contourIdx, color, thickness)

mask = np.zeros(image_gray.shape, np.uint8)
img_contour = cv2.drawContours(mask, [largest_areas[-1]], 0, (255, 255, 255, 255), -1)
```

Bitwise operator are used for image manipulation and used for extracting essential parts in the image. Various Bitwise operator and here we used and operator

```
# Apply Bitwise operator
img_bitcontour = cv2.bitwise_and(image, image, mask= mask)

# apply Hue, Saturation, and value(HSV)
hsv = cv2.cvtColor(img_bitcontour , cv2.COLOR_BGR2HSV)

# set the value for green color(leaf images in our case)
lower_green = np.array ([29, 21, 21])
upper_green = np.array ([85, 265,  194])

# apply inrange method
mask = cv2.inRange(hsv, lower_green, upper_green)
```

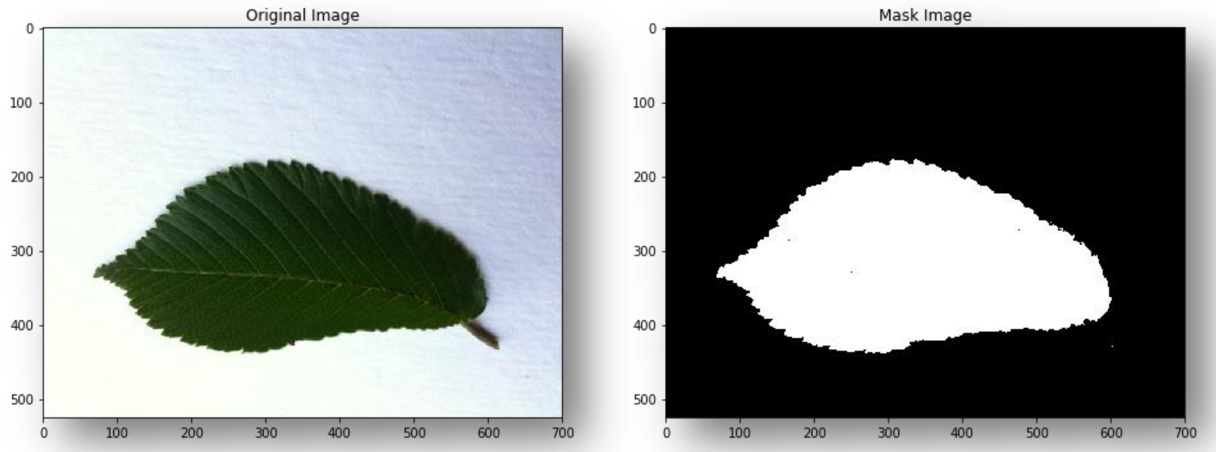

Figure 4.3: Original and Final Mask

```
# Adaptive Gaussian Segmented Thresholding
    cv2.adaptiveThreshold(Inputarray(src), maxvalue, adaptive_method, thresholedtype, blocksize, C)

Adaptive_Gaussian_Segmented = cv2.adaptiveThreshold(mask,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
cv2.THRESH_BINARY,15,2)
```

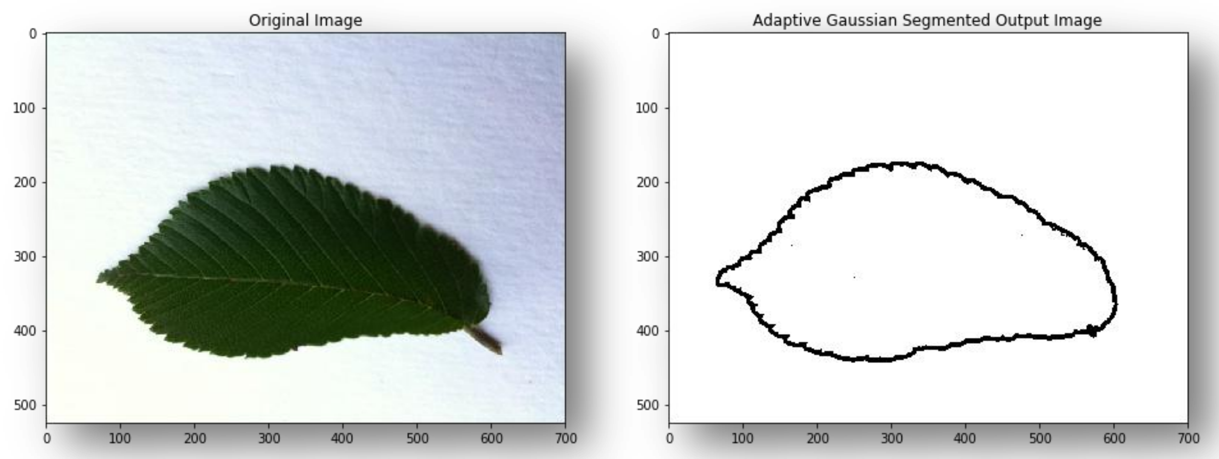

Figure 4.4: Original and Adaptive Gaussian Thresholding

```
# find the contours
cont,_ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)

# draw contours (leaf image)
cont_img  =  cv2.drawContours(cv2.cvtColor(cv2.imread("13002228288852.png"),  cv2.COLOR_BGR2RGB),
cont, -1,255,3)
```

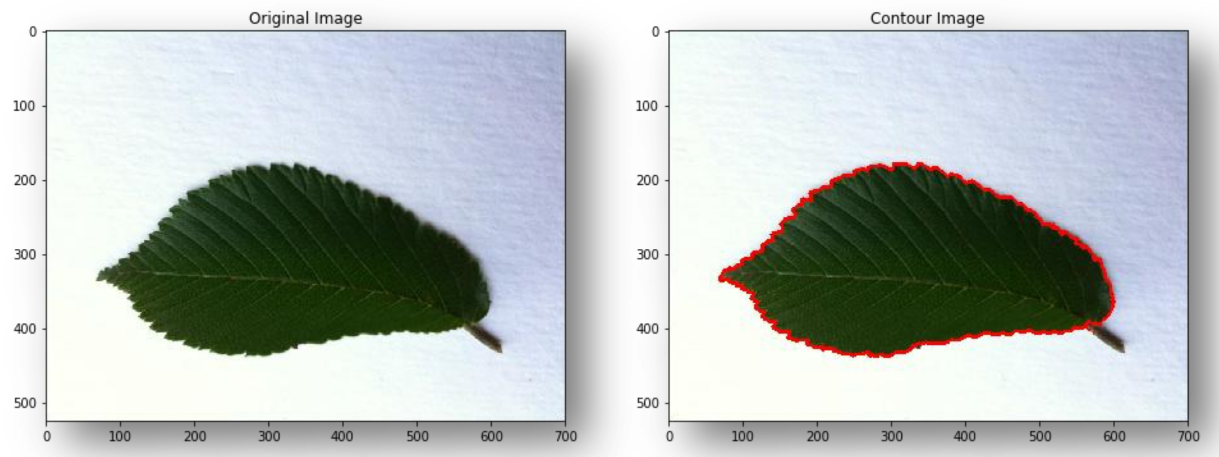

Figure 4.5: Original and Draw Contours

```
# Take a maximum area of contours for Rectangle (Leaf)
c = max(cont, key = cv2.contourArea)

# return the x coordinate,y coordinate, height and weight
x, y, w, h = cv2.boundingRect(c)

# draw rectangle
bounding_box = cv2.rectangle(cv2.cvtColor(cv2.imread("13002228288852.png"),
cv2.COLOR_BGR2RGB), (x, y), (x+w, y+h), (0, 255, 0), 3)
```

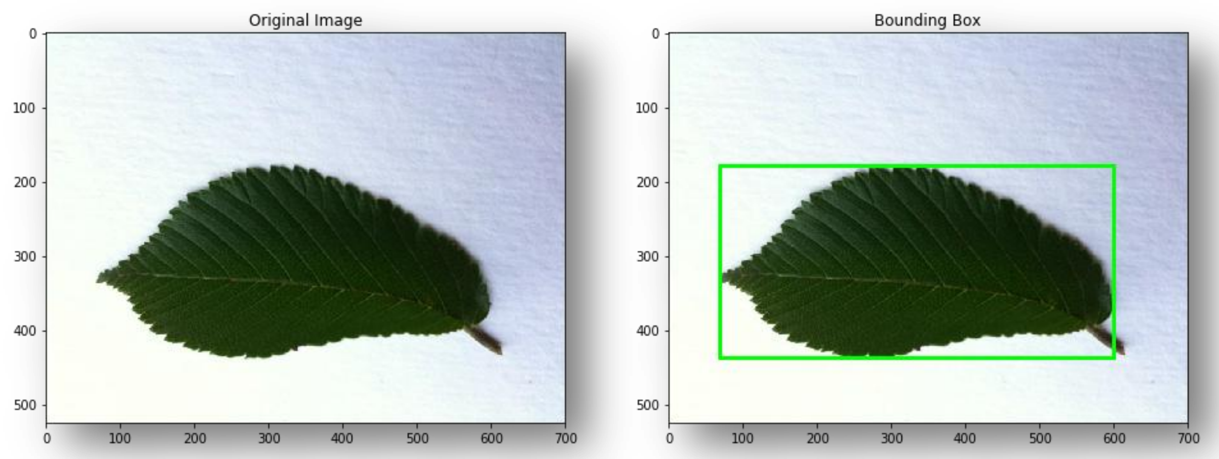

Figure 4.6: Original and Bounding Box

Here the Final summary of Description Solution

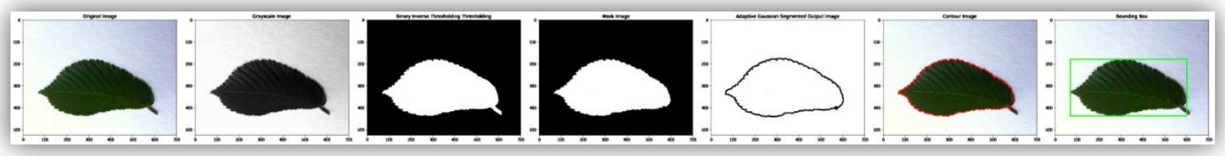

Figure 4.7: Summary of Description Solution

5. Results

The proposed method is experimented with on Leaf Images for all the datasets (leaves_testing_set_1, leaves_testing_set_2). However, It gives much better results on leaves_testing_set_1 as compared to leaves_testing_set_2. And the Accuracy obtained for Set1 is 77% and Set2 is 70%. And the Dice Similarity for Set1 is 82% and 78% respectively.

In some images, the mask is not perfectly images due to the intensity of light in images because in some cases the images are dark and in some cases are light images. Therefore, the mask didn’t apply perfectly to light images especially.

5.1 leaves_testing_set_1

Here are the visualization results of 150 observations for IoU Score and Dice Similarity. And the mean observation is above 80 percent. As you can see below:

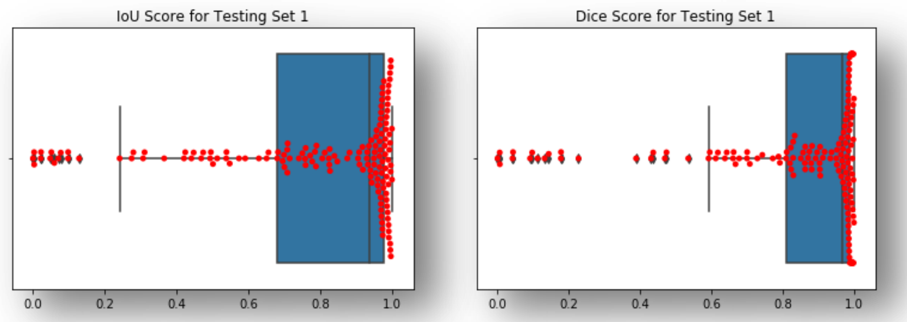

Figure 5.1: Boxplot for IoU and Dice (Set1)

Here is some output result for test set 1.

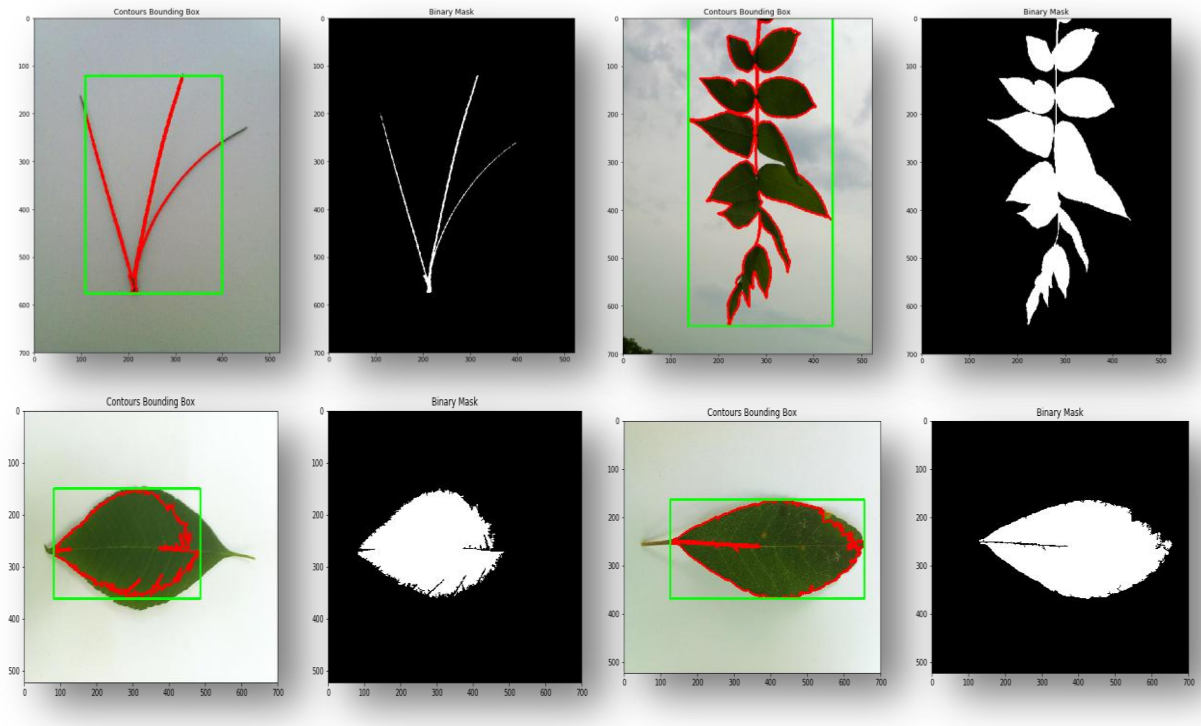

Figure 5.2: Some Result of Test Set 1

5.2: leaves_testing_set_2

Here are the visualization results of 150 observations set 2 for IoU Score and Dice Similarity. And the mean observation is above 80 percent. But you can see some observation result is 0 or close to 0. But in set 1 not too much observation is 0. That's why accuracy is a bit low here because in set 2 most of the images are light and my proposed solution didn't work correctly. As you can see below:

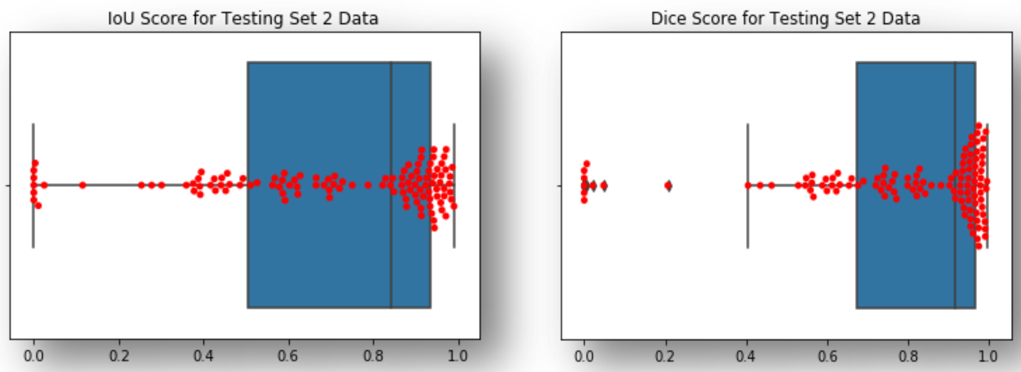

Figure 5.3: Boxplot for IoU and Dice (Set2)

Here is some output result for test set 2

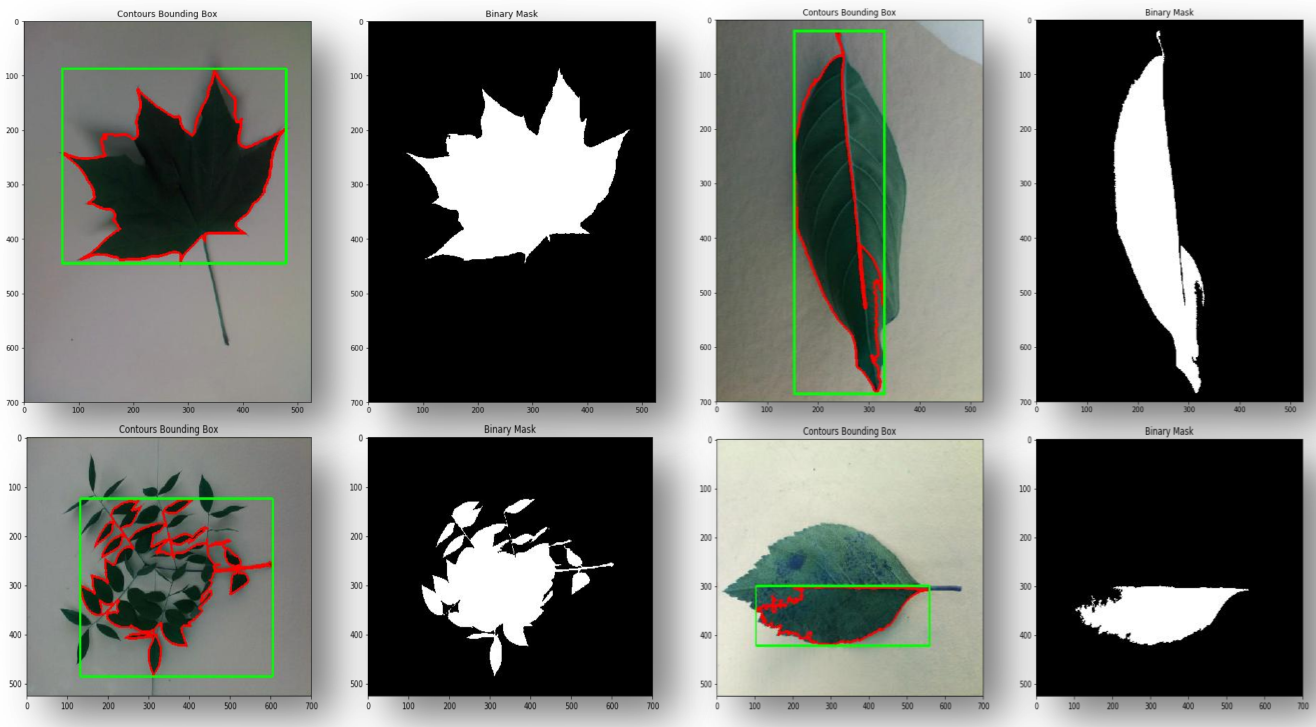

Figure 5.4: Some Result of Test Set 2

5.3: Final Result

In the final result, I combine both (test_set_1 and test_set_2) results in order to get the final IoU score and Dice Similarity result which is 73% and 80% respectively. Below you can see the visualization of 300 observations for both Dice and IoU.

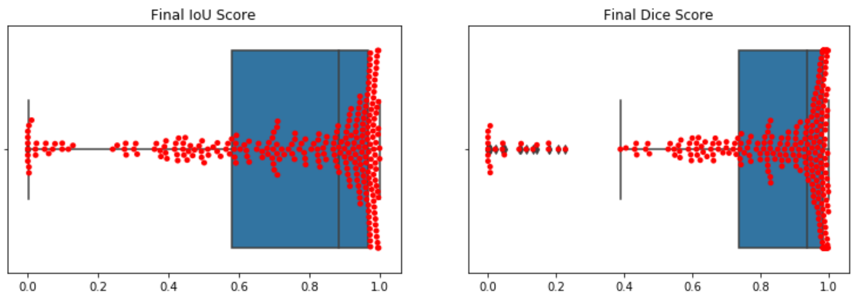

Figure 5.5: Boxplot for IoU and Dice (Final Result)

6. Conclusions

In this project, I did Image Segmentation of Leaf Dataset using OpenCV. I apply Basic Thresholding, HSV, and Bitwise End operator to apply the mask on the leaf to separate from the background. However, it was somehow I get results in dark images and not so good results that I achieved in light images. The description solution achieves the 73% IoU score and 80% in Dice score of 93.88%.

Remarks:

Thank you Dr. RAFAŁ JÓŹWIAK for assigning such a good project. I am a beginner in computer vision and it's really new domain for me as a Data Science student. But it was very interesting to work with OpenCV for image segmentation. And I really enjoyed it to finish it. It was challenging for me especially in light images but I try my best to do it.

7. Reference

[1] https://www.igi-global.com/dictionary/improved-lymphocyte-image-segmentation-using-near-sets-for-all-detection/13839

[2] Shen Pan, "Edge Detection of Tobacco Leaf Images Based on Fuzzy Mathematical Morphology", The 1st International Conference on Information Science and Engineering (ICISE2009), Nanjing, pp. 1219-1222, 2009.

[3] P. Umorya, R. Singh, "A Comparative Based Review on Image Segmentation of Medical Image and its Technique", International Journal of Scientific Research in Computer Science and Engineering, Vol.5, Issue.2, pp.71-76, 2017.

[4] Darshana A., Jharna Majumdar, Shilpa Ankalaki, "Segmentation Method for Automatic Leaf Disease Detection", IJIRCCE, Vol. 3, Issue 7, pp.1-7, 2015.

[5] N. Valliammal, S.N. Geethalakshmi , " A Novel Approach for Plant Leaf Image Segmentation using Fuzzy Clustering" International Journal of Computer Applications, Vol.44, No.13, pp.10-20, 2012

[6] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. C. Lopez, J. V. B. Soares. Leafsnap: A Computer Vision System for Automatic Plant Species Identification. ECCV 2012. (http://leafsnap.com/)