

*Class project report on*

# **Plant Leaf Image Pre-processing for Classification and Disease Detection**

*By*

**Shrikant M. Patil - 202SP024**

Under the Guidance of

**Dr. Shyam Lal**

Department of Electronics and Telecommunication, NITK  
Surathkal

*Date of Submission: 24-01-2021*

In partial fulfillment for the award of the degree

**Master of Technology**  
in  
**Signal Processing and Machine Learning**

at



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**  
**National Institute of Technology Surathkal, Karnataka**

**DEPT. OF ELECTRONICS & COMMUNICATION ENGINEERING**  
**National Institute of Technology Surathkal -Mangalore**  
**2020 – 21**



**CERTIFICATE**

This is to certify that the report entitled “**Plant Leaf Image Pre-processing for Classification and Disease Detection**” submitted by **Shrikant Madhukar Patil (202SP024)**, to the NITK Surathkal in partial fulfillment of the M.Tech. degree in Signal Processing and Machine Learning is a Bonafede record of the work carried out by him under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

---

**Dr. Shyam Lal**  
Assistant Professor  
Dept.of ECE  
NITK Surathkal  
Mangalore

## DECLARATION

I **Shrikant Madhukar Patil** hereby declare that the project report “**Plant Leaf Image Pre-processing for Classification and Disease Detection**” for partial fulfillment of the requirements for the evaluation of Master of Technology to the National institute of technology Surathkal, Mangalore is a Bonafede work done by me under supervision of Professor **Dr. Shyam Lal**.

This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources.

I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission.

Mangalore  
24-01-2021

Shrikant Madhukar Patil

## ACKNOWLEDGEMENT

With immense pleasure I am presenting the “**Plant Leaf Image Pre-processing for Classification and Disease Detection**” Project report as a part of the curriculum of “**EC861–Image Processing and Computer Vision**” under the department of Electronics and Communication Engineering, National Institute of Technology, Karnataka". I wish to thank all people who gave me the unending support. I express my profound thanks to our Professor, **Dr. Shyam Lal**, and all those who have indirectly guided and helped me in the preparation of this project.

## ABSTRACT

Nutrient status of a plant can be diagnosed by detecting the edges and veins of the leaf images since the deficiency symptoms are generally found in interveinal areas along the edges. Now a day's image processing is used to solve different types of problems. Plant disease detection using plant leaf images is under rapid progress now. However, plant leaf image processing is considered as difficult task because of its complex structure and shape. Though modern deep learning techniques are highly capable of doing plant classification and disease detection, preprocessing of plant leaf image is considered to be most fundamental and not avoidable important task. Quality of preprocessing greatly affect the end results after applying deep learning methods. Canny edge detector is being popularly used for extracting texture and shape information of image. However, in case of plant leaf image, due to its complex texture and shape direct application of canny edge detector is not sufficient. Preprocessed image delivered to further deep learning model should not be ambiguous or else It will lead to unnecessary complications in result. This project presents the novel approach to deal with this problem by applying k-means clustering to the image for image segmentation before canny edge detector and then image dilation afterwards if necessary. Despite its simplicity, it is shown that this approach generalizes for diverse plant leaf image conditions.

*Keywords: Image preprocessing, canny edge detector, image segmentation, k-means clustering, deep learning, image dilation etc.*

## Contents

<b>1</b>	<b>Introduction</b>	<b>8</b>
<b>2</b>	<b>Literature Survey</b>	<b>9</b>
	2.1 Image Segmentation Using K-Means Clustering	9
	2.2 Conventional Canny Edge detection Algorithm	10
	2.2.1 Noise Reduction	10
	2.2.2 Gradient calculation using sobel filter:	11
	2.2.3 Non-Maximum Suppression	11
	2.2.4 Double Threshold	11
	2.2.5 Edge Tracking by Hysteresis	12
	2.3 Image Dilation	12
<b>3</b>	<b>Implementation Details</b>	<b>13</b>
	3.1 Image Segmentation using K-Means clustering	13
	3.2 Conversion from RGB to Grayscale Image	14
	3.3 Noise reduction by applying Gaussian Blur	14
	3.4 Gradient Calculation using Sobel Filter	15
	3.5 Non-Maximum Suppression	15
	3.6 Double Thresholding and edge tracking by Hysteresis	15
	3.7 Image Dilation	16
<b>4</b>	<b>Results</b>	<b>17</b>
<b>5</b>	<b>Compilation of work</b>	<b>18</b>
<b>6</b>	<b>Conclusion</b>	<b>19</b>

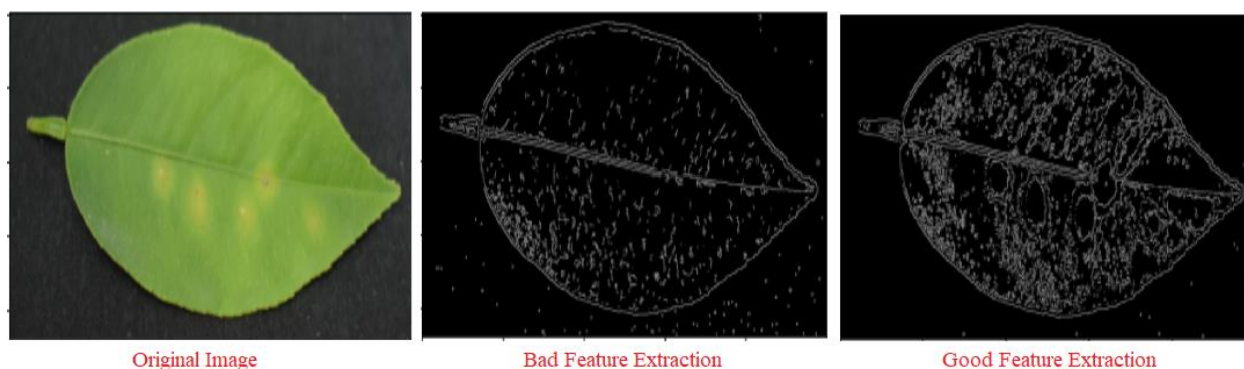
## List of Figures

<b>Figure 1.1</b> Example of good and bad feature extraction of plant leaf image.	8
<b>Figure 2.1</b> Conventional Canny Edge Detector.	10
<b>Figure 3.1</b> Proposed Method Algorithm	13
<b>Figure 3.2</b> Input image and Clustered image.	13
<b>Figure 3.3</b> Gray-Scale image and image after removal of background Cluster from clustered image	14
<b>Figure 3.4</b> Image after applying Gaussian Blur	14
<b>Figure 3.5</b> X-direction Derivative [Gx], Y-direction Derivative [Gy] and Final Magnitude image with edge information.	15
<b>Figure 3.6</b> Image after Non-Maximum Separation.	15
<b>Figure 3.7</b> Final Image after Double Thresholding and edge tracking by Hysteresis.	16
<b>Figure 3.8</b> Final Image after dilation operation.	16
<b>Figure 4.1</b> Results for different plant leaf images	17

# 1. Introduction

All animals including human beings are dependent on plants directly or indirectly. If any damage occurs to the plants, it will reflect on the whole ecological system. Plant diseases are one of the most prominent factors that cause damage to the plants which hampers the supply of crops, fruits, vegetables etc. [1]. In general, farmers and people detect disease by visual observation by the naked eye. Many times, they need to send the images of their plants to the expert for analysis or just observation as every time field visit is not possible. This approach seems more beneficial when people seek help from interstate and even international experts. Observing this raw image and stating the conclusion is like blind guess as image acquisition itself can have thousands of possibilities. Modern deep learning methods can combine experts experience or knowledge with high-speed computers for accurate estimation of end results of various tasks.

To recognize and successfully classify different plant diseases or plants itself, it is a prerequisite to extract different types of features efficiently [1]. These features are fundamental requirement of deep learning methods for further analysis and application. These features may be color, texture, shape, pattern etc. The accuracy and complexity of classification and disease detection fully depend on how features are extracted from plant leaf image or more specifically from the diseased area on the leaf. The good feature extraction technique from plant leaf can increase the efficiency of further deep learning-based classifier or disease detector [2]. This project targets on extracting shape and texture features of leaf which are most of the time sufficient feature for accurate classification and disease identification.



**Figure 1.1** Example of good and bad feature extraction of plant leaf image.

As shown in Figure 1.1, Good feature extracted image not only preserve the texture information but also the disease spots information is also preserved. However Bad feature extracted image does not preserve the disease spots and of no use for further deep learning application while good feature extracted image accurately preserve the important information and leads to the more accurate and easy analysis by further deep learning application. If feature extraction is done properly, chances of getting highly accurate end results increase significantly.



## 2. Literature Survey

### 2.1] Image Segmentation Using K-Means Clustering:

Clustering is one of the most common exploratory data analysis technique used to get an intuition about the structure of the data. It can be defined as the task of identifying subgroups in the data such that data points in the same subgroup (cluster) are very similar while data points in different clusters are very different [1]. In other words, we try to find homogeneous subgroups within the data such that data points in each cluster are as similar as possible according to a similarity measure such as Euclidean-based distance or correlation-based distance. The decision of which similarity measure to use is application-specific.

The algorithm works as follows:

- First, we initialize k points, called means, randomly.
- We categorize each item to its closest mean and we update the mean's coordinates, which are the averages of the items categorized in that mean so far.
- We repeat the process for a given number of iterations and at the end, we have our clusters

Kmeans algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster [1].

Euclidian distance is calculated as follows,

$$D = \sum_{i=1}^m \sum_{j=1}^n W[i, j] |x^i - \mu_j|^2 \text{ ----- [1]}$$

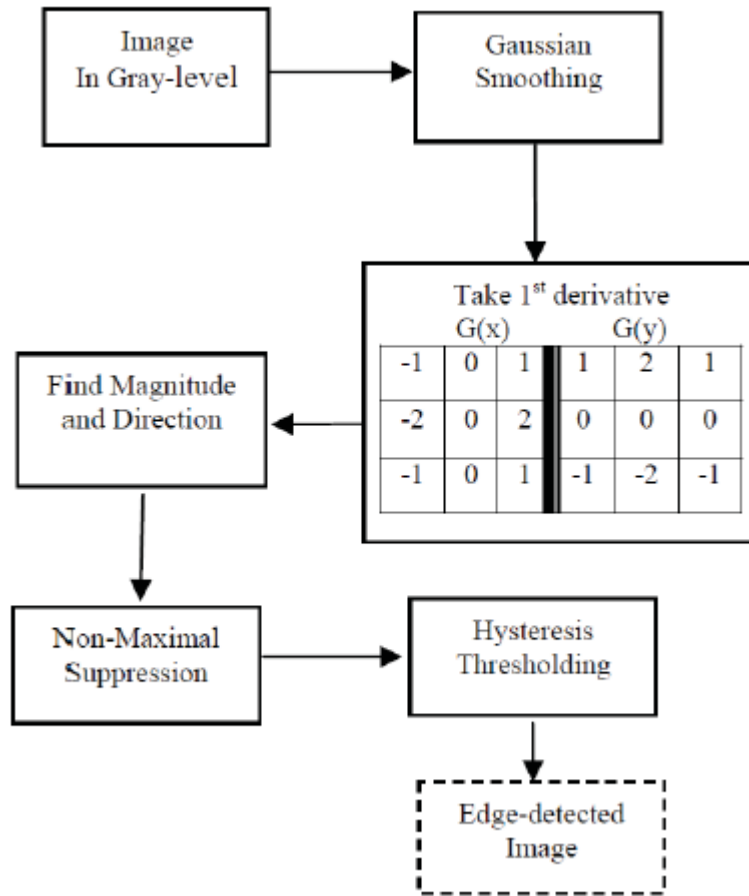
Where D = Euclidian distance

W = 1 if it belongs to cluster k

= 0 otherwise

$\mu$  = Centroid of Xi's cluster

## 2.2] Conventional Canny Edge detection Algorithm:



**Figure 2.1** Conventional Canny Edge Detector [3].

### 2.2.1] Noise Reduction:

Since the mathematics involved behind the scene are mainly based on derivatives (Gradient calculation), edge detection results are highly sensitive to image noise [3]. One way to get rid of the noise on the image, is by applying Gaussian blur to smooth it. To do so, image convolution technique is applied with a Gaussian Kernel (3x3, 5x5, 7x7 etc....). The kernel size depends on the expected blurring effect. Basically, the smallest the kernel, the less visible is the blur.

The equation for a Gaussian filter kernel of size  $(2k+1) \times (2k+1)$  is given by:

$$H = \frac{1}{2\pi\sigma^2} \exp \left( -\frac{(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2} \right) ; 1 \leq i, j \leq (2k+1) \text{ ----- [2]}$$

### 2.2.2] Gradient calculation using sobel filter:

The Gradient calculation step detects the edge intensity and direction by calculating the gradient of the image using edge detection operators. Edges correspond to a change of pixels' intensity. To detect it, the easiest way is to apply filters that highlight this intensity change in both directions: horizontal (x) and vertical (y) When the image is smoothed, the derivatives  $G_x$  and  $G_y$  w.r.t.  $x$  and  $y$  are calculated. It can be implemented by convolving image with Sobel kernels  $K_x$  and  $K_y$ , respectively:

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad K_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Then, the magnitude  $G$  and the slope  $\theta$  of the gradient are calculated as follow:

$$|G| = \sqrt{G_x^2 + G_y^2} \quad \text{-----} [3]$$

$$\Theta(x, y) = \arctan\left(\frac{G_y}{G_x}\right) \quad \text{-----} [4]$$

### 2.2.3] Non-Maximum Suppression:

Here we check basically if the pixels on the same direction are more or less intense than the ones being processed. When the pixel (i, j) is being processed, and the pixels on the same direction (i, j-1) and (i, j+1), If one those two pixels are more intense than the one being processed, then only the more intense one is kept. If say Pixel (i, j-1) seems to be more intense, the intensity value of the current pixel (i, j) is set to 0. If there are no pixels in the edge direction having more intense values, then the value of the current pixel is kept as it is.

### 2.2.4] Double Threshold:

The double threshold step aims at identifying 3 kinds of pixels: strong, weak, and non-relevant [3]:

- Strong pixels are pixels that have an intensity so high that we are sure they contribute to the final edge.
- Weak pixels are pixels that have an intensity value that is not enough to be considered as strong ones, but yet not small enough to be considered as non-relevant for the edge detection.
- Other pixels are considered as non-relevant for the edge.

So, the double thresholds hold for:

- High threshold is used to identify the strong pixels (intensity higher than the high threshold)
- Low threshold is used to identify the non-relevant pixels (intensity lower than the low threshold)
- All pixels having intensity between both thresholds are flagged as weak and the Hysteresis mechanism (next step) will help us identify the ones that could be considered as strong and the ones that are considered as non-relevant.

The result of this step is an image with only 2-pixel intensity values (strong and weak).

#### **2.2.5] Edge Tracking by Hysteresis:**

Based on the threshold results, the hysteresis consists of transforming weak pixels into strong ones, if and only if at least one of the pixels around the one being processed is a strong one.

### **2.3] Image Dilation:**

Basics of dilation:

- Adds pixels at boundaries.
- Increases the object area.
- Used to accentuate features.

Working of dilation:

- A kernel (a matrix of odd size (5,5) is convolved with the image
- A pixel element in the original image is '1' if at least one pixel under the kernel is '1'.
- It increases the white region in the image or size of foreground object increases.

### 3. Implementation Details

Following flow chart gives the brief overview of implemented method.



**Figure 3.1** Proposed Method Algorithm

#### 3.1] Image Segmentation using K-Means clustering:

Role of this algorithm in this project is to classify the image pixels into different no. of important groups so that all possibilities will be considered and preserved by strong artificial edges.

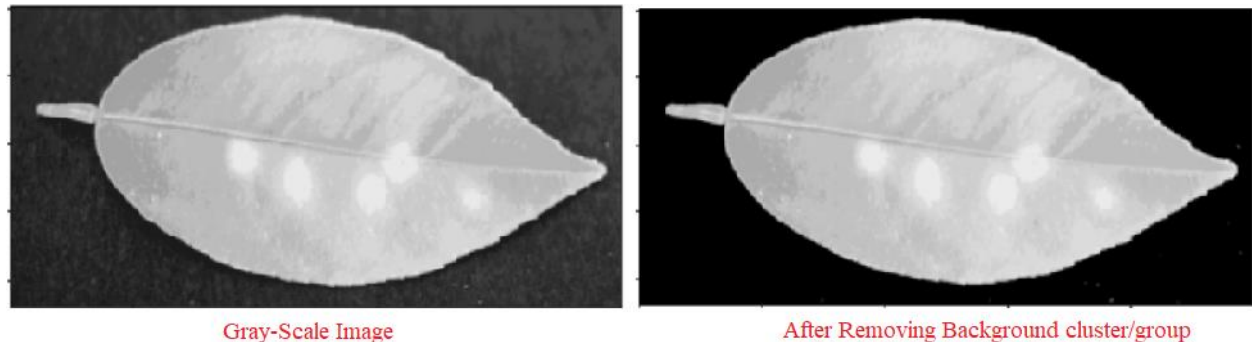


**Figure 3.2** Input image [Left side] and Clustered image [Right side]

As seen from figure 3.1, clustered image not only preserved the disease spots but also highlighted it. This also can be considered as strong artificial edge formed at the diseased spot which is more possibly preserved in the incoming operations on the image.

### 3.2] Conversion from RGB to Grayscale Image:

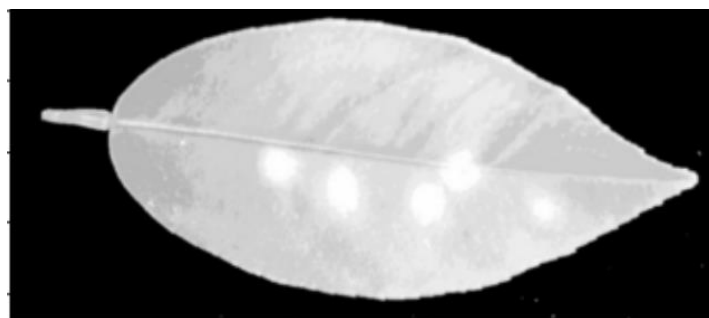
Three channel RGB image is then converted to one channel Gray Scale image as shown in Figure 3.3 below. Also, the background noise (or cluster defining the background image) is removed by Histogram analysis of clustered image.



**Figure 3.3** Gray-Scale image [Left side] and image after removal of background Cluster from clustered image [Right side]

### 3.3] Noise reduction by applying Gaussian Blur:

Mathematics involved in this edge detection method involves the calculation of derivatives, it is highly sensitive to image noise. Among mean, median, adaptive median, Gaussian etc. noise removal filters, Gaussian is known for preserving the edges in the image. Figure 3.4 below shows the image after Gaussian blur is applied.



**Figure 3.4** Image after applying Gaussian Blur

### 3.4] Gradient Calculation using Sobel Filter:

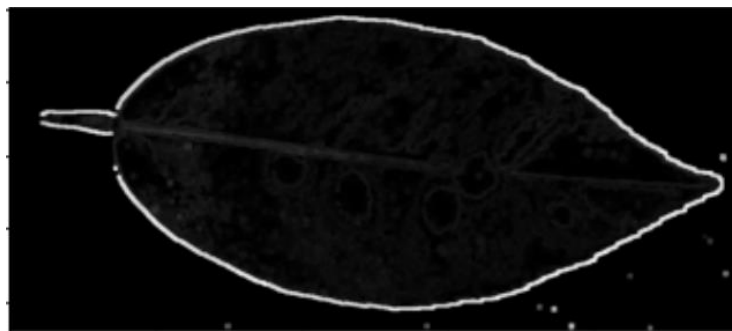
Gradient calculation is basically calculating the derivative of image which also means that detecting the high intensity value changes in the image. First gradient in x and y direction is calculated then merged together to form a magnitude image which contains every possible edge in the image.



**Figure 3.5** X-direction Derivative [Gx], Y-direction Derivative [Gy] and Final Magnitude image with edge information.

### 3.5] Non-Maximum Suppression:

As evident from Figure 3.5, Magnitude image consists of all strong and weak edges, Also, some of the edges are thicker while some are thinner which results into loss of clarity in important part of image. Gradient calculation beside of magnitude also gives angle orientation of each pixel. We can use this information for thinning out the thicker ones as shown In Figure 3.6 below.

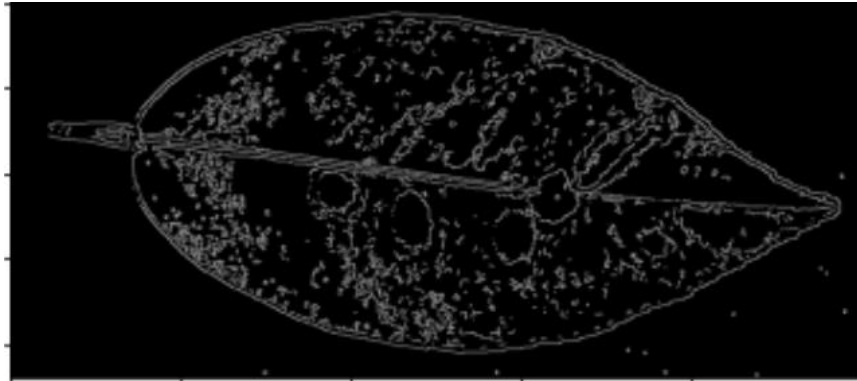


**Figure 3.6** Image after Non-Maximum Separation.

### 3.6] Double Thresholding and edge tracking by Hysteresis:

From Figure 3.6, we notice that still some pixels are brighter than others. The result of double thresholding is an image with only two-pixel values (strong and weak). By edge tracking

by hysteresis weak pixels are transformed into strong edges if and only if it is surrounded by at least one strong pixel.

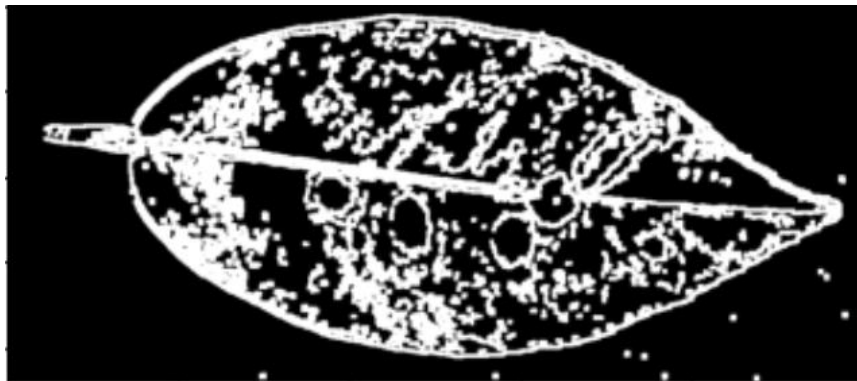


**Figure 3.7** Final Image after Double Thresholding and edge tracking by Hysteresis.

It is evident from the Figure that method proposed in this project gives better edge detected image for further analysis by deep learning. Diseased spots are clearly visible or preserved as it is.

### 3.7] Image Dilation (\*if required):

Sometimes but not always, Disease spots at very early stage are not traceable even after complete image pre-processing. Also, some deep learning methods may require thicker strong edges which are thinned at non-maximum separation stage. Image dilation basically adds pixels at the boundaries. Figure 3.8 shows the image dilation effect in this case (\*though not required).


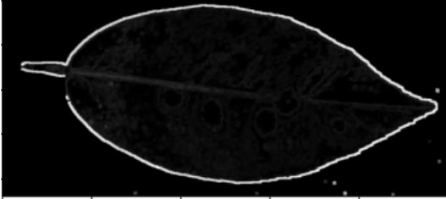
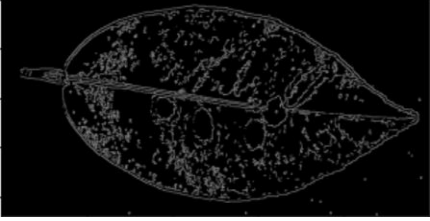

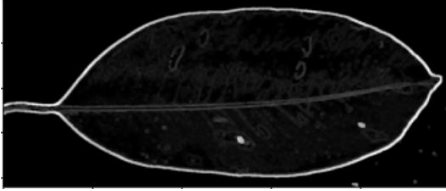
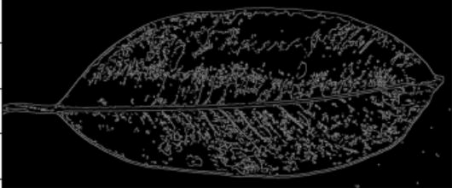

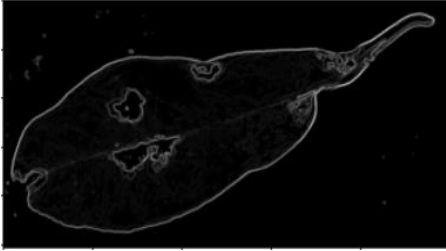
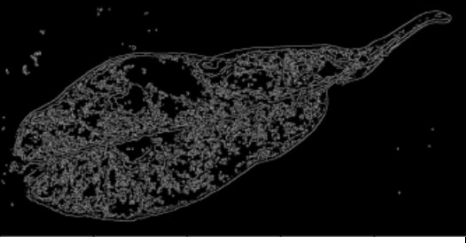

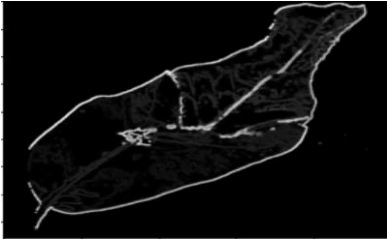
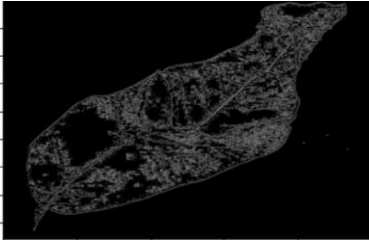

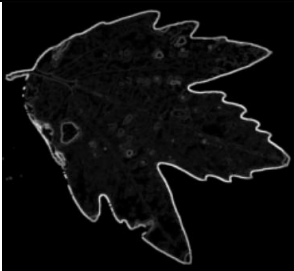
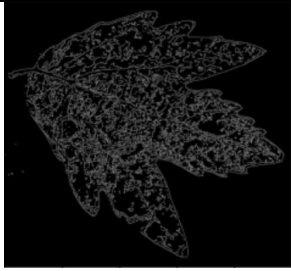


**Figure 3.8** Final Image after dilation operation.



## 4. Results

Results for different plant leaf images with this algorithm are as shown in Figure 4.1 below.

Input Image	Output Image (Without Double Thresholding but with dilation)	Output Image (With Double Thresholding but without dilation)
<b>Lemon Leaf</b> 		
<b>Jamun Leaf</b> 		
<b>Pomegranate Leaf</b> 		
<b>Guava Leaf</b> 		
		
<b>Chinar Leaf</b>		

**Figure 4.1** Results for different plant leaf images

## 5. Compilation of Work

The python code for the model has been compiled as a python notebook and can be successfully compiled on Google Collaboratory. The notebook can be accessed [here](#) . The files required for running this notebook has been compiled into a publicly accessible Google Drive directory, which can be found [here](#) .

## 6. Conclusion

As seen from Figure 4.1,

- 1] If image dilation is performed right after Non-Maximum Suppression without performing double thresholding & edge tracking by hysteresis then resultant output image is better suited for disease detection while other edge information (Leaf veins) for plant classification is lost.
- 2] On the other hand, if all the steps in algorithm are followed in order but image dilation (last step) is not performed then resultant output image is better suited for plant classification as veins details are retained while disease information is lost.

Experiments with different leaf images have shown that this algorithm is better suited for image pre-processing of plant leaf images. The processed images can then be used further by modern deep learning methods for Plant classification as well as plant disease detection.

## References

- [1] *Plant Leaf Disease Detection using Mean Value of Pixels and Canny Edge Detector*, Dr. S.M. Taohidul Islam, Md. Abdul Masud<sup>±</sup>, Md. Arif Ur Rahaman, Md. Mehedi Hasan Rabbi, 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), 24-25 December, Dhaka.
- [2] *An Effective Algorithm for Edges and Veins Detection in Leaf Images*, R. Radha, S. Jayalakshmi, 2014 World Congress on Computing and Communication Technologies.
- [3] *Advanced Plant Leaf Classification Through Image Enhancement and Canny Edge Detection*, Jibi G. Thanikkal<sup>1</sup>, Ashwani Kumar Dubey<sup>2</sup>, Thomas M.T.<sup>3</sup>, 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions).