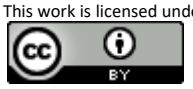
Available online at: https://ijact.in

Date of Submission	30/08/2019
Date of Acceptance	15/11/2019
Date of Publication	03/12/2019
Page numbers	3485-3493 (9 Pages)

Cite This Paper: S.Jeyalakshmi, R.Radha. A novel approach to segment leaf region from plant leaf image using automatic enhanced grabcut algorithm, 8(11), COMPUSOFT, An International Journal of Advanced Computer Technology. PP. 3485-3493.



ISSN:2320-0790

COMPUSOFT
An International Journal of Advanced Computer Technology

A NOVEL APPROACH TO SEGMENT LEAF REGION FROM PLANT LEAF IMAGE USING AUTOMATIC ENHANCED GRABCUT ALGORITHM

S.Jeyalakshmi[1], R. Radha[2]

[1]Assistant Professor and Research Scholar, Department of Computer Science,
Guru Nanak College (Autonomous), Chennai
bjeya27@gmail.com
[2]Associate Professor, Research Department of Computer Science,
SDNB Vaishnav College for Women, Chromepet, Chennai
radhasundar1993@gmail.com

Abstract: Segmentation of leaf region from background is one of the essential pre-processing steps required in the Plant Leaf Image Processing. This paper proposes an innovative segmentation approach for extracting color leaf region from the healthy or infected plant leaf image with background using an enhanced automatic GrabCut algorithm that does not take any input from the user. In this method, first GrabCut algorithm was applied on the original image. The algorithm removes background but shadows remain in the resultant image which may cause misinterpretations in further processing steps. Hence, the shadows in the image were removed by thresholding 'a' and 'b' components of CIELAB color space. This step created holes in the infected region, which had similar color as that of shadow, of the leaf image. Hence, the image obtained was binarized and holes were filled with white (foreground) colorizing Flood Fill algorithm. From this binary image containing only leaf region, the color leaf region of the image was filtered. The accuracy achieved was 98%.

Keywords: Segmentation; Grab Cut; RGB Color Space; CIELAB Color space; Threshold; Flood Fill algorithm

I. INTRODUCTION

This paper proposes an accurate algorithm to segment leaf region from background. This is a pre-processing step required in research areas such as plant species classification, Content Based Image Retrieval (CBIR), classification of plant diseases and diagnosing nutrient stress in plants from plant color leaf image. The proposed algorithm removes background from the leaf image using automatic GrabCut algorithm. The shadows, if present, were not removed in this step. One of the challenges in this work includes removing shadows from the image. Further, the colors of some of the infected regions were same as that of shadow color. As a result, these regions were also removed while removing shadows. This may cause loss of information in subsequent steps of research. Hence, further processing is required to get the leaf region without holes. Finally, the leaf region without shadows and holes were obtained. Section II describes about the related work in this field. Section III deals with the materials and the image processing methods applied in the proposed algorithm. Section IV elaborates the proposed algorithm for background removal and the results. Section V discusses

performance evaluation of the proposed algorithm as compared with GrabCut Algorithm. Section VI discusses the conclusion and future work.

II. LITERATURE SURVEY

P. A. Moghaddam et al. [1]had applied the threshold function on R, G and B components of the image to remove background as the reflectance of the sugar beet leaf region was more than the background region for monochrome images. Training-error and automatic thresholding methods were used for calculating threshold values. The background pixel values were set to zero and the foreground regions were retained for further processing. Madhogaria et al. [2] applied Convex energy function on I1I2I3 color space by calculating weighted squared sum of individual channels I1, I2 and I3 with weights 0.1, 0.45, 0.45 respectively. The result was a binary image with a '0' representing the background and a '1' representing the foreground. K. Dang et al. [3] had thresholded Hue component, of HSV color space, for segmentation of color region, as Hue component separates color information from other features such as illumination, M. Schikora et al [4] discuss three broad categories of image segmentation techniques. The first one is an energy minimization method. It detects the shape boundaries or contour of an image. The second one is graph-cut algorithm. The advantage of this approach is less computation time and approximately global optimum solution. The third, Total Variation (TV) minimization, is a hybrid approach of the both.M. Vassallo-Barco[5] and Chaudhari et al. [25] had proposed a method for segmentation by automatically selecting the threshold using Otsu method.D. Khattab et al. [6] had applied an automated GrabCut algorithm, by obtaining initial values from Orchard and Bouman Clustering technique thereby avoiding user interaction, on different color spaces, RGB, HSV, CMY, XYZ and YUV and had got best result in RGB. Rzanny et al. [7] propose a semi-automated GrabCut algorithm that was initialized with a rectangle, which marks the definite foreground and the corners of the rectangle were taken as seed points for the background. The user iterativelyrefines markers by denoting foreground and background, on requirement. Tyagi et al [23] had proposed a threshold based method, in which a pixel with intensity value greater than a threshold was segmented as foreground (the object) and the rest of the pixels as background. Auearunyawat et al. [24] hadsuggested an effective algorithm for segmenting sugarcane leaf from background. The adaptive threshold of mean was applied on gray scale image for extracting leaf from background. The resulting image was converted into YCbCr color space and only the second channel was selected to avoid warp up and remove shadow. The adaptive threshold of mean was applied on second channel. The image obtained was a good quality one but couldn't extract midrib. Hence both the results were ANDed to produce the final leaf image without background. Sannakki et al. [19] propose a membership function u(x), corresponding to leaf area, for segmenting leaf region from background by thresholding G component of RGB color space. B. Chen et al.[17] and Sun et al. [8] had scanned leaf image after placing leaf on a white paper for avoiding complex background. MeenaPrakash et al. [10] segmented plant leaf images into three clusters one each for background, green regions, and infected regions of the leaf image, after converting from RGB colorimage to L*a*b* colorimage, using K-Means clustering algorithm. Patil and Bodhe[9] had converted the input RGB image into gray scale by selecting some threshold value and segmented the image into object and background. Grand Brochier et al. [15] had compared fourteen segmentation methods such as Thresholding, MeanShift, Pyramidal MeanShift, Graphcut, Wateshed, Snakes, B-splines Snake, Grabcut and Felzenszwalb, Kurtz, Weber, Power Watershed, SLIC, GAC were performed with no input stroke and with no color distance map and with input stroke and with color distance map. The research focused on segmentation methods that can be used in smartphones.

III. MATERIALS AND METHODS

A. DataSet

The experiments and observations presented in this paper were done on PlantVillage dataset [14]. In this work, we have used Grape Leaf images suffering from Black Rot, Black Measles, Leaf Blight diseases and healthy Grape leaf images with a data set size of 63, 94, 100 and 105 respectively.

B. Methods

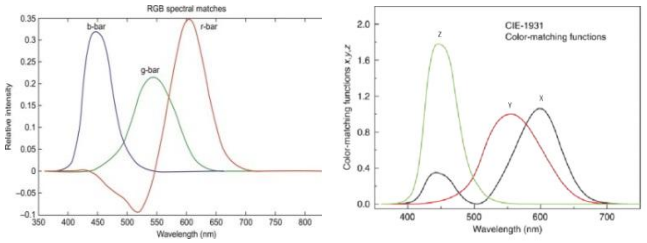
The proposed algorithm uses image processing methods such as RGB, XYZ, CIELAB color spaces, gray scale image, binary image and conversion from one color space to another. A widely used Grabcut algorithm was applied on input image. The resultant leaf image had not remove shadows from the leaf region. Hence, the output leaf image from GrabCut algorithm was further processed to remove the shadows. This resultant image had acquired holes in the process of removing shadows as the color of the shadows and the infected regions of leaf were same. Hence, Flood Fill algorithm was used to fill the holes. The final leaf image generated by the proposed algorithm was an accurate leaf image segment.

1) RGB and XYZ Color Spaces: Visible light is a section of electromagnetic spectrum that is visible to the human eye. The values for the wavelengths of colors red, green and blue were set by CIE (International Commission on Illumination), an organization responsible for color standards.

We obtain primary colors by adding different values of red, green and blue colors. The amount of red, green and blue to be added to form a color is provided by color matching experiment and the results of this experiment performed by CIE in 1931, is given by RGB Color-matching functions [12] shown in Fig. 1

In Fig.1, the red, green and blue values are negative for some wavelengths but dealing with color matching functions with negative values are inconvenient.Hence,

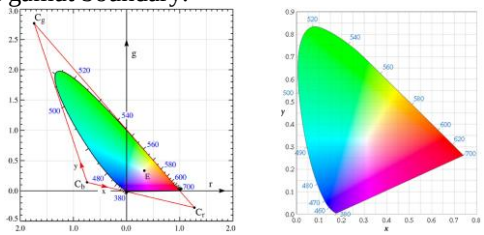
CIE introduced CIEXYZ color model [26], Fig. 2. The values of X, Y and Z required to form any particular color are called tristimulus values. The tristimulus values can be normalized to x+y+z=1 by dividing X, Y and Z values by X+Y+Z to obtain x,y and z values. Thus, a color can be represented with chromaticity co-ordinates x and y values alone, which are independent of luminance energy [16].



(a) *(b)*

Fig. 1: (a) RGB Color matching functions(CIE, 1931) [12] (b) XYZ Color matching functions (CIE,1931) [12]

The gamut is a complete subset of colors. A particular color may be represented in a color space or may be reproduced on a device if the chromaticity of a color lies within gamut boundary.



(a) (b)

Fig. 2: Chromaticity diagram [26] (a) The triangle is CIERGB 1931 (b) The triangle is CIEXYZ 1931

RGB to CIEXYZ ColorSpace Conversion: The values of X, Y and Z can be obtained by linear transformation of corresponding R, G and B values.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.431 & 0342 & 0.178 \\ 0.222 & 0.707 & 0.071 \\ 0.020 & 0.130 & 0.939 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

2) CIELAB ColorSpace: The problem with CIEXYZ is that colorimetric distance between individual colors does not correspond to observed difference in color. For example, the distance between green and greenish yellow is large whereas distance between green and blue is very small. CIELabcolor space, defined by CIE in 1976, solves the above problem. It has three components 'L' for lightness, 'a' for green-red values and 'b' for blue-yellow values.

In this color space, the differences correspond to distances when measured colorimetrically. The a axis extends from green (-a) to red (+a) and the b axis from blue (-b) to yellow (+b) and the luminosity increases from bottom to top [18], Fig. 3.

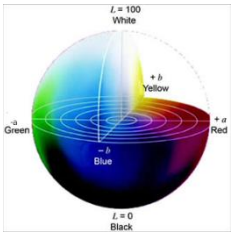


Fig. 3CIELAB Color Chart [18]

CIEXYZ to CIELAB ColorSpace Conversion:

$$L = \begin{cases} 116*\left(Y/Y_n\right)^{1/3} - 16 & \text{for } Y/Y_n > 0.008856 \\ 903.3*Y/Y_n & \text{Otherwise} \end{cases} \quad (2)$$

$$a = 500*\left(f\left(X/X_n\right) - f\left(Y/Y_n\right)\right) \quad (3)$$

$$b = 200*\left(f\left(Y/Y_n\right) - f\left(Z/Z_n\right)\right) \quad (4)$$

$$\text{where } f(t) = \begin{cases} t^{1/3} & \text{for t>0.008856} \\ 7.787*t+16/116 & \text{otherwise} \end{cases} \quad (5)$$

Here X_n, Y_n and Z_n are the tristimulus values of the reference white.

3) Gray Scale Image: A Gray scale image is a range of gray shades which has less information than color images, for each pixel. The darkest possible shade is black, the brightest being white and intermediate shades have equal level of primary colors (red, green and blue).

RGB to Gray-Scale Conversion: The gray values can be calculated using the following formula

$$Gray = (R + G + B)/3 \quad (6)$$

4) Binary image: Grayscale image can be converted into binary by thresholding gray value of each pixel. If the pixel value is greater than threshold value then pixel is assigned a value of '1' which represents white and a '0' to represent black in the binary image.

5) GrabCutAlgorithm: The unsupervised clustering algorithm GrabCut segments object from background in a digital color image. The algorithm requires the user to mark a rectangular area as initial input and the outer part of this rectangle is considered as definite background. The algorithm works iteratively on each pixel of the image and assigns either of the label 'Background' or 'Foreground'.

The pixels outside the rectangular area are considered as known background and inside are unknown background. A model is created, with this data, to find out whether the unknown pixels are foreground or background.

Gausian Mixture Model: Gaussian Mixture model is a probabilistic model that automatically learns and assigns data points into subpopulation they belong to within overall populations. It can be considered as k-means clustering for including covariance matrix of data and centers of latent Gaussians.

In GrabCut Algorithm, K component multivariate Gaussian Mixture models are created for both of the foreground and the background regions resulting in a total of 2K components. The segmentation is performed by Orchard-Bouman Clustering technique.

Orchard-Bouman Clustering Algorithm: Orchard and Bouman clustering algorithm [14] is a color quantization clustering algorithm for splitting into good clusters using Eigen vector and color covariance matrix. Initially, all pixels are placed in the same cluster. The algorithm takes initial cluster and number of clusters or components K as input.

Step 1: Initial cluster = Unknown U Foreground

Step 2: Calculate the mean μ_1, the mean of C_1 and $\sum_1$, the covariance matrix of cluster C_1.

Step 3: For i= 2 to K do

i. Find the set C_n with largest Eigen value and corresponding Eigen vector e_n

ii. Split C_n into two sets,

a. $C_i = \{x \in C_n : e^T z_n <= e_n{}^T \mu_n\}$ and

b. $C_n^* = C_n - C_i$,

iii. Compute $\mu_n^*, \sum_n^*, \mu_i,$ and $\sum_i$

This results in K pixel clusters.

6) Flood Fill Algorithm: Flood fill algorithm fills a connected component with a given color. The algorithm starts with an interior pixel (x, y) and reassigns all pixels that are currently set to a given interior color with the desired fill color. The process is repeated until all interior pixels have been reassigned with the desired color [21]. A connectivity value of four, four nearest neighbors, are used.

7) Performance Metrics: Performance evaluation is essential to select a more suitable algorithm. The evaluation algorithms require ground truth segmented image (Reference image) and the segmented image obtained by applying the proposed method (Test image). The ground truth images can be created either manually, by removing background pixels or by using some segmentation technique and correcting the errors.

Mean Square Error, Peak Signal to Noise Ratio and Structural similarity are some of the examples of Performance evaluation techniques. The methods compares reference image with the test images.

Accuracy: Accuracy is defined as the percentage of relationship between correctly classified pixels over total number of pixels [13]. This is given by the following formula:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

where TP(True Positive), TN (True Negative), FP (False Positive) and FN (False Negative) refers to the number of actual positives classified as Positive, the number of pixels correctly classified as Negative and the number of actual negatives classified as Positive, and to the number of actual positives classified as Negative respectively.

Table 1.Confusion Matrix

		Predicted class	
		Positive	Negative
True Class	Positive	TP	FN
	Negative	FP	TN

Precision: The Precision is the ratio of true positive over all pixels classified as true positive [13].

$$precision = \frac{TP}{TP + FP} \quad (2)$$

Recall: The Recall is the ratio of true positives and all the positive pixels [13].

$$recall = \frac{TP}{TP + FN} \quad (3)$$

F1 Score: F1 score tells the balance between precision and recall. The value of F1 score will be equal to 100% for high quality segmentation.

$$F1\ Score = 2 * \left(\frac{precision * recall}{precision + recall} \right) \quad (4)$$

Jaccard Index: Jaccard index depends on the amount of pixels correctly and wrongly classified. It is an overlap ratio measure and its values range from 0%, for no overlap, to 100%, for, complete overlap. JaccardIndex[20] is given by

$$Jaccard\ Index = \frac{TP}{FP + TP + FN} \quad (5)$$

Mean Squared Error: Mean Squared Error measures the pixel difference between reference image, f and test image,g.

$$MSE(f, g) = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(f_{ij} - g_{ij} \right)^2 \quad (6)$$

*Peak-Signal-to-Noise Ratio:*Peak-Signal-to-Noise Ratio, also a pixel difference measurement, expressed in decibels dB

$$PSNR(f, g) = 10 \log_{10} \left(\frac{MAXPIXEL^2}{MSE(f, g)} \right) \quad (7)$$

where MAXPIXEL is the maximum pixel value. MAXPIXEL is 1 for binary image, 255 for gray scale image, 255 for each component R, G and B of RGB color image. A high value of PSNR indicates a good quality test image as MSE is less.PSNR value will tend to infinity, when MSE is 0. For a large value of MSE, PSNR will tend to 0.

Structural Similarity Index Measure (SSIM): Structural

Similarity index is an image quality measure[11] and [22]. It used to find the similarity between two images and is given by

$$SSIM(f,g) = l(f,g)c(f,g)s(f,g) \quad (8)$$

and

$$l(f,g) = \frac{2\mu_f\mu_g + C_1}{\mu_f{}^2 + \mu_g{}^2 + C_1} \quad (15)$$

$$c(f,g) = \frac{2\sigma_f\sigma_g + C_2}{\sigma_f{}^2 + \sigma_g{}^2 + C_2} \quad (16)$$

$$s(f,g) = \frac{\sigma_{fg} + C_3}{\sigma_f\sigma_g + C_3} \quad (9)$$

In equation (14), l(f,g) is the luminance comparison function which tells how close the mean luminance of two images, c(f,g) is the contrast comparison function that measures the closeness of contrast of two images and s(f,g) gives the structure comparison function which is a measure of correlation coefficient between the two images.
SSIM values would range between 0 and 1, 0 for images with no similarity and 1 for identical images.

IV. PROPOSED METHODOLOGY AND RESULTS

The overall workflow of the proposed method is shown in Fig. 4 and ***Pi*** in the figure refers to ***Pseudo Codei.***The original RGB Leaf image with background, Fig. 5(a), is passed on to Grab Cut algorithm(***P1***). The algorithm (***P2***) takes coordinates of an initial rectangular area of interestviz. (5, 10, 230, 245) where (5, 10) is the starting point, 230, 245 are the width and height of the rectangle respectively.The regions outside the rectangular area were considered as definite background. The algorithm labels (Gaussian Mixture Models (GMMs) are created for the initial foreground and background classes using Orchard-Bouman clustering algorithm) each pixel in the image as background, foreground, probable background or probable foreground. Then, the foreground pixels are extracted from the original RGB image with the mask obtained from GrabCut algorithm. Pseudo Code ***P0*** gives the sequence of steps involved in this proposed method.

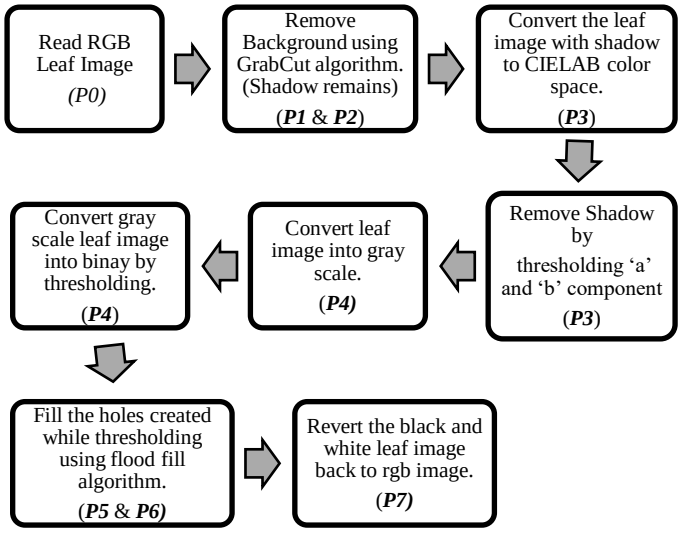


Fig.4.Overall Workflow

Pseudo Code 0(*P0*): Remove Background from leaf image using enhanced GrabCut Algorithm
Input: $Img_{original}$ Original image with background
Output: $Img_{foreground}$Foreground color Leaf Image
Procedure EnhancedGrabCut
Img_{shadow}= BgRemove($Img_{original}$)
Img_{holes}=RemoveShadow(Img_{shadow})
$Img_{holes\text{-}bw}$=Binarize(Img_{holes})
$Img_{no\text{-}holes\text{-}bw}$= FillHoles($Img_{holes\text{-}bw}I$)
$Img_{foreground}$ = RGBForeground($Img_{no\text{-}holes\text{-}bw}$)
End Procedure

Pseudo Code 1(*P1*):Automatic classification of Foreground and BackGround pixels

Input: $Img_{original}$
Output: Img_{shadow}Foreground(Leaf) Image with shadow
Procedure BgRemove
// Each pixel is marked 0 for sure background,
//1 for sure foreground, 2 for probable background //and 3 for probable foreground
// BackGround and ForeGround models are //constructed as an array of size (1, 65)
//with all the elements initialized to zero.
//Rectangle is the Region of Interest containing the //foreground (Leaf).
//The *Rectangle* takes starting point (x,y), width //and //height of the region of interest.
//The regions outside the *Rectangle* are //consideredas definite background.
//CREATE and INITIALIZE Mask
Mask=Zeros(*Img*$_{original}$.shape[:2]).
INITIALIZE *BgModel* = Zeros(1,65)
INITIALIZE *FgModel* = Zeros(1,65)
IterationCount=5
Rectangle = (5,10,230,245)
CALL grabCut(*Img*$_{original}$,*Mask, Rectangle,BgModel, FgModel, IterationCount, InitializeWithRectangle*)
IF (*Mask*==2) or (*Mask* ==0) / // Background
Mask2=0 // Black Color
ELSE // Region of Interest
Mask2=1 //White Color
ENDIF
$Img_{shadow} = Img_{leaf} * Mask2$
End Procedure

(a) (b)

Fig. 5 Results of Pseudo Code 1 &2 respectively (a) Original image (b) Foreground image after applying GrabCut algorithm

Pseudo Code 2(*P2*): grabCut algorithm

Input $Img_{original}$
Mask
Rectangle
BgModel
FgModel
IterationCount
InitializeWithRectangle
Output: Mask marked with 2/0 for background pixels and 1 for foreground pixels

Procedure grabCut
Step 1: Pixels inside the rectangle are marked as unknown. Initially, these pixels are added into foreground class.
Step 2: Pixels outside the rectangle are marked as background. These pixels are placed in background class.
Step 3: Gaussian Mixture Models (GMMs) are created for the initial foreground and background classes using Orchard-Bouman clustering algorithm.
Step 4: Every pixel in the foreground class is assigned to the most probable Gaussian component in the foreground GMM. Likewise, every pixel in the background class is assigned to the most probable Gaussian component in the background GMM.
Step 5: New GMMs learned from the pixels sets created in the previous step.
Step 6: A graph is built and GrabCut is run to find a new class of foreground and background pixels
Step 7: Repeat steps 4 to 6 until the classification converges.
End Procedure

The algorithms ***P1*** and ***P2*** perfectly remove the background but the shadows remain in the foreground. The output from this step was passed on to algorithm ***P3*** for removing shadows. This algorithm converts the RGB image into CIELAB image and removes shadow by thresholding 'a' and 'b' components. The threshold values were set to 0 to 145 and 0 to 136 for 'a' and 'b' components respectively. The values above the range remove some of the region of interest and below the range remove does not completely remove shadow.

The thresholding completely removes shadow but creates holes in the infected region of the leaf that have same value, for 'a' and 'b' components, as that of shadows Fig. 5(b). Hence these holes must be filled with the original leaf color for further processing. This is done by binarizing the thresholded image and filling the holes in the leaf with white color. This requires first converting color image into gray scale image and then thresholding the gray pixel values.

Fig. 6(a) was converted into gray scale and the resultant gray image, Fig. 6(b), was converted into binary image, Fig. 6(c), by thresholding gray pixel values in the range from 10 to 255 corresponding to foreground, algorithm ***P4***.

The binary image with holes were filled using flood fill algorithm which takes the seed pixel and keeps moving in the East, West, South and North directions and replaces old(holes in black color) with the new color (leaf area in whitecolor), algorithm ***P5*** and ***P6*** Fig. 7(a).

Finally, algorithm ***P7*** filters the color leaf region from the original input image, Fig. 7(b), by masking the foreground pixels of binary leaf image obtained from ***P5,*** Fig.7(c)**.**

Pseudo Code 3(*P3*): Shadow Removal Algorithm

Input : Img_{shadow}Foreground with shadow
Output :$Img_{no\text{-}shadow}$Foreground without shadow (having holes)

Procedure RemoveShadow
//Convert from RGB colorspace into CIELAB Color space
// Here X_n, Y_n and Z_n are the tristimulus values of the reference white.
$R = (Img_{leaf}[:,:,1]$
$G = (Img_{leaf}[:,:,2]$
$B = (Img_{leaf}[:,:,3]$
FOR each pixel in Img_{leaf}

INITIALIZE $$StdMat = \begin{pmatrix} 0.431 & 0.342 & 0.342 \\ 0.222 & 0.707 & 0.071 \\ 0.020 & 0.130 & 0.939 \end{pmatrix}$$

CALCULATE *XYZ*=StdMat *[*R G B*]

IF $\left(\frac{Y}{Y_n} > 0.008856\right)$

$$L = 116 * \left(\frac{Y}{Y_n}\right)^{1/3} - 16$$

ELSE

$$L = 903.3 * Y / Y_n$$

ENDIF

$$a = 500 * \left(f\left(X / X_n\right) - f\left(Y / Y_n\right)\right)$$

$$b = 200 * \left(f\left(Y / Y_n\right) - f\left(Z / Z_n\right)\right)$$

ENDFOR
//Threshold 'a' and 'b' components of CIELAB
IF (a>=0 and a<=145) and (b>=0 and b <=136)
ShadowMask=255
$Img_{no\text{-}shadow}$=Mask Img_{leaf} with ShadowMask
End Procedure

Pseudocode 4(*P4*): Image binarization using thresholding method

Input: Img_{leaf}
Output: Img_{bw}

Procedure Binarize

$Img_{red}, Img_{green}, Img_{blue} = Split(Img_{leaf})$

$Img_{gray} = (Img_{red} + Img_{green} + Img_{blue})/3$

$Img_{bw} = \begin{cases} 1 \text{ if } Img_{gray} >= 10 \text{ and } Img_{gray} <= 255 \\ 0 \text{ otherwise} \end{cases}$

End Procedure

Pseudo code 5(*P5*):Algorithm to fill holes in the leaf image

Input: Img_{bw}**Black and white image with holes**

Output: $Img_{no\text{-}holes}$ Leaf image without holes

Procedure FillHoles

$Mask_{floodfill}$ = zeros(*nrows*+2,*ncols*+2)

FloodFill(Img_{bw},$Mask_{floodfill}$,(0,0),255)

// Invert floodfilled leaf image

Img_{bw_inv}=bitwise_not(Img_{bw})

// Combine the two images to get the foreground in

//black and white

$Img_{no\text{-}holes} = Img_{bw} \mid Img_{bw\_inv}$

End Procedure

Pseudo code 6(*P6*): Algorithm to fill the holes(black color) with white color in the leaf image

Input: Img_{bw} image to be filled

$Mask_{floodfill}$ mask

(*x*,*y*)–seed point

fillcolor – new color to be filled by replacing old color

Output:Img_{bw}Leaf image in black and white

Procedure FloodFill

IF ($Img_{bw}(x,y)$== $Mask_{floodfill}(x,y)$)

$Img_{bw}(x,y)$=*fillcolor*

ENDIF

FloodFill(Img_{bw},$Mask_{floodfill}$,*x*+1,*y*,*fillcolor*,*oldcolor*)

FloodFill(Img_{bw},$Mask_{floodfill}$,*x*-1,*y*,*fillcolor*,*oldcolor*)

FloodFill(Img_{bw},$Mask_{floodfill}$,*x*,*y*+1,*fillcolor*,*oldcolor*)

FloodFill(Img_{bw},$Mask_{floodfill}$,*x*,*y*-1,*fillcolor*,*oldcolor*)

End Procedure

Pseudo code 7(*P7*): Revert back binary image into RGB image without background

Input: Img_{fg_bw} Foreground image in black and white

$Img_{original}$Original image

Output: Img_{fg}Segmented RGB colorLeaf image

Procedure RGBForeground

r, *c* = shape(Img_{fg_bw})

R, *G*, *B* = split($Img_{original}$)

r, *c*=shape(Img_{fg_bw})

FOR *i*= 0 to *r*-1

FOR*j*= 0 to *c*-1

IF ($img_{fg\_bw}[i,j]$ == 0)

r[*i*,*j*]=0

g[*i*,*j*]=0

b[*i*,*j*]=0

ENDIF

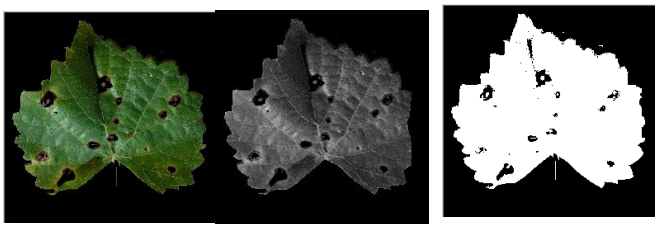
ENDFOR

ENDFOR

img_{fg}=cv2.merge((*r*,*g*,*b*))

End Procedure

Fig. 6 Results of Pseudo Code 3 & 4 respectively a) Foreground without shadow(with holes in infected region of leaf) b) gray scale image with holes c) binary image with holes



(a) (b) (c)

(a) (b) (c)

Fig. 7: Results of Pseudo code 5& 6. (a) Foreground after applying Flood fill algorithm (b)Original Input image (c) Final Output of the proposed algorithm (Result of Pseudo code 7)

V. PERFORMANCE EVALUATION

Performance evaluation can be done by comparing ground truth image with the result obtained from proposed algorithm. The ground truth images were generated manually and compared with the final output from the proposed algorithm.

Grand Brochier et al. [15] had compared several segmentation methods for extracting tree leaves from natural images and observed the results tabulated in table 2.

Table 2 with no input stroke and no color distance map [15]

Method	Precision	Recall	Jaccard Index	SSIM
Snakes	70.80%	82.05%	61.08%	0.66
B-splines Snake	76.26%	86.46%	69.28%	0.77
Grab Cut	83.66%	84.56%	79.68%	0.77
Weber	89.75%	82.69%	78.12%	0.78
Simple Linear Iterative Clustering (SLIC)	83.49%	78.33%	69.76%	0.76
Guided Active Contour (GAC) method	**92.67%**	**85.98%**	**82.42%**	**0.81**

Performance evaluation of the proposed method

Performance of the proposed method and GrabCut algorithm were evaluated using performance metrics Accuracy, Precision, Recall, F1 score, Jaccard Index, Mean Square Error, Peak Signal to Noise Ratio, Structural Similarity Index Measurement. Table (3) tabulates the

performance measure. From the results obtained and performance analysis it can be determined that the proposed method performs well and enhances the results of GrabCut algorithm.

Table 3: Performance metrics of GrabCut algorithm and proposed method

Performance Metric	GrabCut Algorithm	Proposed Method
Confusion Matrix	$\begin{bmatrix} 32983 & 1441 \\ 453 & 30660 \end{bmatrix}$	$\begin{bmatrix} 34057 & 366 \\ 840 & 30273 \end{bmatrix}$
Accuracy	96.80%	**98.00%**
Precision	95.20%	**98.80%**
Recall or Sensitivity	97.05%	**98.04%**
F1 score	96.80%	**97.60%**
Jaccard Index	93.60%	**95.40%**
Mean Square Error	0.03 dB	**0.02 dB**
Peak Signal to Noise Ratio	64.06 dB	**65.61 dB**
Structural Similarity Index Measure	0.93	**0.94**

VI. CONCLUSION AND FUTURE WORK

GrabCut is one of the widely used algorithms for removing background in color images. In spite of its wide usage, it has the shortfall that it does not remove the shadowin the segmented foreground image. In this paper, an enhanced GrabCut algorithm was presented that removes shadow in color leaf images with an accuracy of **98%**. This work can be used in areas like content based image retrieval, plant leaf image classification, plant disease detection and classification and diagnosis of plant nutrient deficiency from plant color leaf images.

VII. REFERENCES

[1] Parviz Ahmadi Moghaddam, Mohammadali Haddad Derafshi and Vine Shirzad, "Estimation of Single Leaf Chlorophyll Content in Sugar Beet using Machine Vision", Turk J Agric For Vol. 35, No. 6, (2011), pp. 563-568 © TÜBİTAK, doi: 10.3906/tar-0909-393.

[2] Satish Madhogaria, Marek Schikora, Wolfgang Koch, Daniel Cremers, "Pixel Based Classification for Detecting Unhealthy Regions in Leaf Images", Proceedings of Informatikschaff Communities. pp. 1-11, 2011.

[3] K. Dang, H. Sun, Jean-Pierre Chanet, J. Gracia-Vidal, J. M. Barcelo-Ordinas, H. L. Shi and K. M. Hou. "Wireless Multimedia Sensor Network for Plant Disease Detections", Proceedings of New Information Communication Science and Technology for Sustainable Development: France-China International Workshop pp. 1-6 2013.

[4] Marek Schikora, Adam Schikora, Karl-Heinz Kogel, Wolgang Kochi and Daniel Cremers. "Probabilistic Classification of Disease Symptoms caused by Salmonella on Arabidopsis Plants", GI Jahrestagung Journal, Vol. 2, No. 10, pp. 874-879, 2010.

[5] Marcelo Vassallo-Barco, Luis Vives-Garnique, Victor Tuesta-Monteza, Heber I. Mejia-Cabrera, Raciel Year Toledo, "Automatic Detection of Nutritional Deficiencies In Coffee Tree Leaves Through Shape and Texture Descriptor", Journal of Digital Information Management, Vol.15, No.1 2017

[6] Dina Khattab, Halamousher Ebied, Ashraf Saad Hussein AndMohamed FahmyTolba. "Color Image Segmentation Based on Different Color Space Models using Automatic Grabcut", Hindawi Publishing Corporation, The Scientific World Journal Volume 2014, Article Id 126025, 10 Pages, http://dx.doi.org/ 10.1155/2014/126025

[7] Michael Rzanny, Marco Seeland, Jana Walchen and Patrick Mader, "Acquiring And Preprocessing Leaf Images ForAutomated Plant Identification: Understanding The Tradeoff Between Effort And Information Gain", Plant methods (2017) 13:97, DOI 10.1186/S 13007-017-0245-8.

[8] Sun, J., Mao, H. and Yang, Y., "The Research on the Judgment of Paddy Rice's Nitrogen Deficiency based on image", 2009, in IFIP International Federation for Information processing, Volume 294, Computer and Computing Technologies in Agriculture II, Volume 2, eds. D. Li, S. Chunjiang, (Boston: Springer), pp. 1049-1054.

[9] Sanjay B. Patil, Dr.Shrikant K. Bodhe. "Leaf Disease Severity Measurement using Image Processing", International Journal of Engineering and Technology Vol. 3 (5), 2011, 297-301.

[10] R. MeenaPrakash, G. P. Saraswathy, G. Ramalakshmi, K. H. Mangaleswari, T. Kaviya. "Detection of Leaf Diseases and Classification using Digital Image Processing". 2017 International Conference on Innovations in Information, Embedded and Communication System (ICIIECS).

[11] D. Preethi, Dr. D. Loganathan, "Image Assessment Metrics based on Distortion Measures", International Journal of Engineering Science Invention (IJESI) ISSN(Online): 2319-6734, ISSN (Print): 2319-6726, PP. 52-58

[12] Salomon, David (2007). "Data Compression: The Complete Reference (4 ed.)", Springer-Verlag London pp.281, ISBN 978-1846286025

[13] Eduardo Fernandez-Moral, Renato Martins, Denis Wolf, Patrick Rives. "A new metric for evaluating sematic segmentation: leveraging global and contour accuracy", Workshop on Planning, Perception and Navigation for Intelligent Vehicles, PPNIV17, Sep 2017, Vancouver, Canada. Hal-01581525.

[14] Images used in this research work were from the PlantVillage dataset (https://github.com/salathegroup/plantvillage-deeplearning-paper-dataset) (Last Accessed Date: 28-10-2019) licensed under Plant Village Dataset Creative Commons 3.0 Share and Share Alike

[15] Manuel Grand-Brochier, Antoine Vacavant, GuillameCerutti, Camille Kurtz, Jonathan Weber Laure Tougne. "Tree Leaves Extraction in Natural Images: Comparative Study of preprocessing Tools and Segmentation Methods". IEEE Transactions on image processing, Institute of Electrical and Electronics Engineers (IEEE), 2015, Vol. 24(5), pp.1549-1560.

[16] https://www.sciencedirect.com/topics/engineering/color-matching-function (Last Accessed Date:15-09-2019).

[17] Baisong Chen, Zhuo Fu, Yuchun Pan, Jihua Wang, and ZhixuanZeng."Single Leaf Area Measurement Using Digital Camera Image". CCTA 2010, Part II, IFIP AICT 345, pp. 525-530. 2011. © IFIP International Federation for Information Processing 2011.

[18] https//www.colorcodehex.com/color-model.html (Last Accessed Date : 21-10-2019)

[19] Sanjeev S Sannakki, Vijay S Rajpurohit and Sagar J Birje. "Comparison of Different Leaf Edge Detection Algorithms Using Fuzzy Mathematical Morphology", International Journal of Innovations in Engineering and Technology (IJIET) Vol. 1 Issue 2 August 2012, pp. 15-21, ISSN: 2319 – 1058.

[20] http://www.di.univr.it/documenti/OccorrenzaIns/matdid/matdid453675.pdf (Last Accessed Date: 28-10-2019)

[21] Donald Hearn, M. Pauline Baker. "Computer Graphics – C Version", Second Edition, 2006, pearson education Inc. and Dorling kindersley publisher Inc., Delhi, ISBN 978-81-7758-765-4.

[22] Alain Horé and DjemelZiu. "Image quality metrics: PSNR vs. SSIM", 2010 International Conference on Pattern Recognition, 1051-4651/10 © 2010 IEEE, DOI 10.1109/ICPR.2010.579

[23] AkanshaTyagi, PriyaKumari and Dr. Mahesh Kumar Yadav. "Image Segmentation Techniques in DIP", International Journal of Advanced Research Engineering Technology and Sciences, June 2016, Volume 3, Issue - 6 ISSN: 2394-2819

[24] PitiAuearunyawat, TeerasitKasetkasem, AudthasitWongmaneeroj, Akinori Nishihara and RachapornKeinprasit. "An Automatic Nitrogen Estimation Method in Sugarcane Leaves Using Image Processing Techniques, International Conference on Agricultural, Environmental and Biological Sciences (ICAEBS' 2012) May 26-27, 2012 Phuket.

[25] PiyushChaudhary, Anand K. Chaudhari, Dr. A. N. Cheeran and ShardaGodara. "Color Transform Based Approach for Disease Spot Detection on Plant Leaf", International Journal of Computer Science and Telecommunications Vol. 3, Issue 6, June 2012

[26] http://cs.haifa.ac.il/hagit/courses/ist/Lectures/IST03_ColorXYZ.pdf (Last Accessed Date:28-10-2019)