

# **FOOD MATURITY AND DISEASE DETECTION USING DIGITAL IMAGE PROCESSING**

Submitted in partial fulfilment of the Degree of

Bachelor of Technology



MAY – 2015

By

Tanvi Mehra (111097) ,  
Mohit Mahajan (111118) ,  
Akanksha Tripathi (111134)

Under the supervision of

Ms Pragya Gupta

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,

WAKNAGHAT

## ABSTRACT

We have developed an automated system that detects the maturity of the fruit as well as detects if the plant is diseased. In this project we worked on the fruits whose maturity can be determined on the basis of the colour. We have mainly concentrated our project on tomatoes as their maturity can be detected on the basis of the colour as well as they are one of the most eaten fruits around the globe. Further we went on to determine if the tomato plant is diseased or not on the basis of the symptoms observed on the plant leaves. Detection of the diseased plant at correct time is very important as ignorance of it can lead to the spread of disease in other plants too. This could cause destruction to the plantation on large scale. The main purpose of our project is to improve the crop yield by detection of the disease on time and the proper use of the crop on the basis of its maturity.

We have worked on two methods of digital image processing in our project. Initially we performed thresholding algorithm to determine the maturity of the fruit as well as to detect the diseased plant. To make the system more generalised and self-adapting we shifted to k-means clustering. Finally we did a comparative analysis of both the methods to analyse which method is more suitable in different conditions .

Name : Tanvi Mehra

---

Signature of Student

Name : Mohit Mahajan

---

Signature of Student

Name : Akanksha Tripathi

---

Signature of Student

Date :

---

Signature of Supervisor

(Ms Pragya Gupta)

Date :

## **CERTIFICATE**

This is to certify that project report entitled “ **FOOD MATURITY AND DISEASE DETECTION USING DIGITAL IMAGE PROCESSING**”, submitted by *Tanvi Mehra* (111097), *Mohit Mahajan* (111118), *Akanksha Tripathi* (111134) in partial fulfilment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering to Jaypee University of Information Technology, Wagnaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

**Date:**

**Supervisor’s Name: Ms Pragya Gupta**

**Designation: Assistant Professor (Grade-II)**

## ACKNOWLEDGEMENT

We are very grateful and highly acknowledge the continuous encouragement, invaluable supervision, timely suggestions and inspired guidance offered by our project supervisor **Ms Pragma Gupta**, Department of Electronics and Communication Engineering, Jaypee University of Information Technology, Waknaghat in bringing this project to a successful completion.

We are grateful to **Prof. T.S Lamba**, Dean, Academics & Research, Jaypee University of Information Technology and to **Prof. Sunil Bhooshan** Head, Department of Electronics and Communication Engineering, Jaypee University of Information Technology for their continued support and encouragement. We offer our sincere appreciation for the learning opportunities provided by them. The completion of this project could not have been accomplished without the support of our teachers and friends who have always extended all sorts of help whenever needed.

Tanvi Mehra

Mohit Mahajan

Akanksha Tripathi

# CONTENTS

1. Introduction.....	7
2. Literature Survey.....	8
3. Thresholding based image segmentation.....	10
3.1. Image segmentation.....	10
3.2. Thresholding.....	10
4. Project design and implementation of thresholding method.....	11
4.1 Flow chart.....	11
4.2 Image acquisition.....	11
4.3 Pre-processing.....	12
4.4. Detection / Segmentation.....	12
4.4.1. General algorithm of thresholding.....	12
4.4.2 Varibale thresholding.....	13
4.5. Output Analysis.....	13
5. Simulation and experimentation.....	16
6. Simulation results of thresholding.....	18
7. Discussion of results.....	18
8. Kmeans clustering based image segmentation.....	19
8.1.Clustering methods.....	19
8.1.1. Types of clustering.....	19
8.2.Kmeans clustering.....	20
8.2.1 Properties.....	20
9. Project design and Implementation.....	21
9.1.Flow chart.....	21
9.2.Image Acquisition.....	21
9.3.Pre-processing.....	22
9.4.Segmentation.....	22
10. Simulation and Experimentation.....	23
11. Simulation results.....	27

11.1	Discussion of result .....	27
11.2	Presentation of results and their analysis.....	29
11.3	Comparison of results.....	34
11.4	Conclusion.....	35
11.5	Code.....	35
12.	Disease detection in Tomato Plant .....	37
12.1	Disease detection.....	37
12.2	Flowchart.....	38
13.	Simulation and experimentation .....	40
13.1	Simulation Results.....	40
13.2	Simulation of Thresholding.....	40
13.3	Simulation Results.....	40
14.	K-means clustering for detection of infected portion of the leaf.....	45
14.1	K-means Clustering.....	45
14.2	Flowchart.....	45
15.	Simulation of K-means.....	48
15.1	Simulation of K-means.....	48
15.2	Results .....	50
15.3	Comparison of results.....	52
15.4	Future Aspects.....	53
16.	References.....	54
17.	Image References.....	55

## CHAPTER 1

### 1. Introduction and Objective:

Agriculture forms the mainstay of Indian economy, and it contributes to 13.7% of the GDP (gross domestic product). The total cultivation areas and yields for agriculture products have witnessed a tremendous increase, due to advance in cultivation technology, which is followed by tremendous market values. India has a vast potential to emerge as a major exporter of agricultural produce, but the high post-harvest losses in handling and processing, lack of quick quality evaluation techniques etc. contributes to a very low share in global market. The increase in the population and the increased expectation of food products of high quality and safety standards demands the need for the growth of accurate, fast and objective quality determination of food and quality products. Therefore, it is very important to ensure good quality food and agricultural products in our country.

Many fruits are consumed fresh, and so appropriate appearance is of utmost importance. Immaturity and ripening disorders in fruits are very common. Proper packaging and product supply requires proper sorting. The most common method for sorting of fruits is Manual sorting. Picking of fruits by hand is a tedious job and it adds to high labour costs, labour fatigue, inconsistency and low precision due to the factors like variations in ambient light intensity, differences in perception of quality and unskilled labour. Automatic fruit sorting using machine vision can, however, improve the quality of the product. Also, it is the best technique as it is non-destructive, accurate, and consistent and reduces dependence on available manpower. The degree of maturity, defects, moisture, nutrients etc. can be accurately identified by machine.

In our project, we used machine vision system in determining the maturity of various fruits like tomatoes, apples, mangoes, oranges etc. and further classified them as mature and immature. Prior to our experiment, a lot of work has been done in this field. Various methods used for the maturity classification were MRI, computed tomography, thresholding based, gradient based, region based, and classification based methods for image segmentation, size, shape, colour and texture features for object measurement and statistical fuzzy logic, k-means, and neural network methods for classification. Initially we performed thresholding based algorithm to determine the maturity based on colour. However, for improvement in results and generalization we shifted to k-means clustering algorithm. This improved the better identification and quantification of the objects in the partitioned region of image.

## CHAPTER 2

### 2. Literature Survey

This is what literature has to say about the field:

- Over the past three decades, number of methods for quality evaluation and sorting of agricultural products have been developed by the researchers. With modern technologies sophistication of non-destructive methods have been evolved. Optical methods that make use of high speed optical detection and computerized data processing are the best known methods which contribute to high speed quality evaluation and sorting with high accuracy.
- Today, non-destructive quality evaluation have improved with the use of various modern image acquisition techniques like line scan camera, X-ray scanning, ultra-sonic scanning and NMR imaging, in conjunction with image processing techniques.[1]
- Fruit quality grading can also be done by machine vision sensors. Shape grading can be done by Fourier based shape separation method and colour recognition can be done by multivariate discriminant analysis.[2]
- This study also helped us to know that image processing techniques for food maturity evaluation includes MRI (magnetic resonance imaging), computed tomography, pixel and local pre-processing approaches for image pre-processing, thresholding based, gradient based, region based, and classification based methods for image segmentation, size, shape, colour and texture features for object measurement and statistical, fuzzy logic and neural network methods for classification.
- Many fruits were taken under consideration for the maturity determination and we chose to focus on tomato as it is consumed by millions of people daily and 75% tomatoes produced are consumed fresh[3]. A machine vision based experimental sorting for tomatoes that was based on parameters like shape, size maturity (color) and defects. The eccentricity, average of colour components and 2-D pixel area respectively were the variables defining shape, maturity and size of tomatoes respectively. CCD cameras, a microcontroller, sensors and computer were used for sorting. Algorithms (otsu's method) developed based on visual basics 2008 were used to analyse the images. The overall system accuracy was 90% with defect detection, shape and size algorithm contributing to 84.4%,90.9% and 94.5% accuracies respectively.[3]
- In the Australian journal of crop science 2011, images of tomatoes were acquired using machine vision system. The algorithm had two steps, in first step the background was



removed in the RGB color space and the by using combination of RGB, HIS and YIQ spaces ripened tomatoes were extracted. Second step included the use of morphological features of an image to localise the ripe tomatoes. The total accuracy of the algorithm was 96.36%. [4]

- In a paper on detecting tomatoes in green houses using vision based method, the overall architecture is built around a method for classifying individual image regions. It is divided into two phases, in the offline learning space, for fixed sized image regions, a binary classifier is created that provides object/non-object decisions. While the online detection phase, at each location of test image, uses the classifier to perform a dense multi-scale scan reporting preliminary object decisions. The decisions are then the basis for final object detections. This approach is data driven and purely bottom up using low level visual features to detect objects. The performance, however, is dependent on the data set creation. [5]
- In a research, the ripeness level of the fruit is estimated without touching it. Two techniques used for this were colour image segmentation and fuzzy logic. Images of fruits were clicked from four different directions and the desired part of each image was separated using colour image segmentation. The mean value of primary colours (red, green, blue) of the segmented parts were calculated and given as input to FIS(Fuzzy Interference System) editor-1. The decision about the part of fruit that is ripe, under-ripe, about to ripe, about to over-ripe, over-ripe is given by this editor. The remaining three images are also operated in the same way. All the four outputs are given to FIS editor-2 which finally gives the decision whether the whole fruit is ripe, under-ripe, over ripe. [6]
- In 2004, F.mendoza ,et al got the accuracy of 98% by converting the RGB image of banana into CIELAB. [7]
- In 2003 and 2008, average values of RGB were used to evaluate maturity levels of peaches, oranges and apples. [8]
- In 2010, Zhi-yuan Wen, et al detected the maturity of citrus fruits using machine vision. B.ojeda-magana, et al detected the ripeness level of bananas and tomatoes using different portioning clustering algorithms. [9]
- Choi detected the tomato maturity rate using colour image analysis . [10]

Of all the methods we chose thresholding based image segmentation, which involved conversion of RGB to YCbCr. Since it was a user defined method so we moved on to k-means clustering, which is function defined method.

## CHAPTER 3

### 3. Thresholding based image segmentation

#### 3.1. Image segmentation:

In segmentation an image is sub-divided into its constituent regions and objects. The level of detail to which the sub-division is carried depends on the problem being solved that means when the object or the regions of interest in an application have been detected, the segmentation should stop. Segmentation of significant images is one of the toughest tasks in image processing. The accuracy with which the segmentation is carried determines the success or failure of computerized analysis behaviour. Hence, for accurate segmentation proper care should be taken. Segmentation algorithms are basically based on two properties of intensity values:

- i. Discontinuity
- ii. Similarity

In case of discontinuity, partitioning of the image is done based on abrupt changes in intensity, such as edges whereas in the second category partitioning of the image into regions that are similar according to a set of pre-defined criteria. Thresholding falls under the second category.

The simplest way to segment an image is to separate light and dark regions. **In our project we started with thresholding based segmentation method.**

Hence, we conclude that Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as super pixels). Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images.

#### 3.2. Thresholding:

The simplest method of image segmentation is called the thresholding method. This method is based on a clip-level (or a threshold value) to turn a gray-scale image into a binary image.

*The key of this method is to select the threshold value (or values when multiple-levels are selected).*

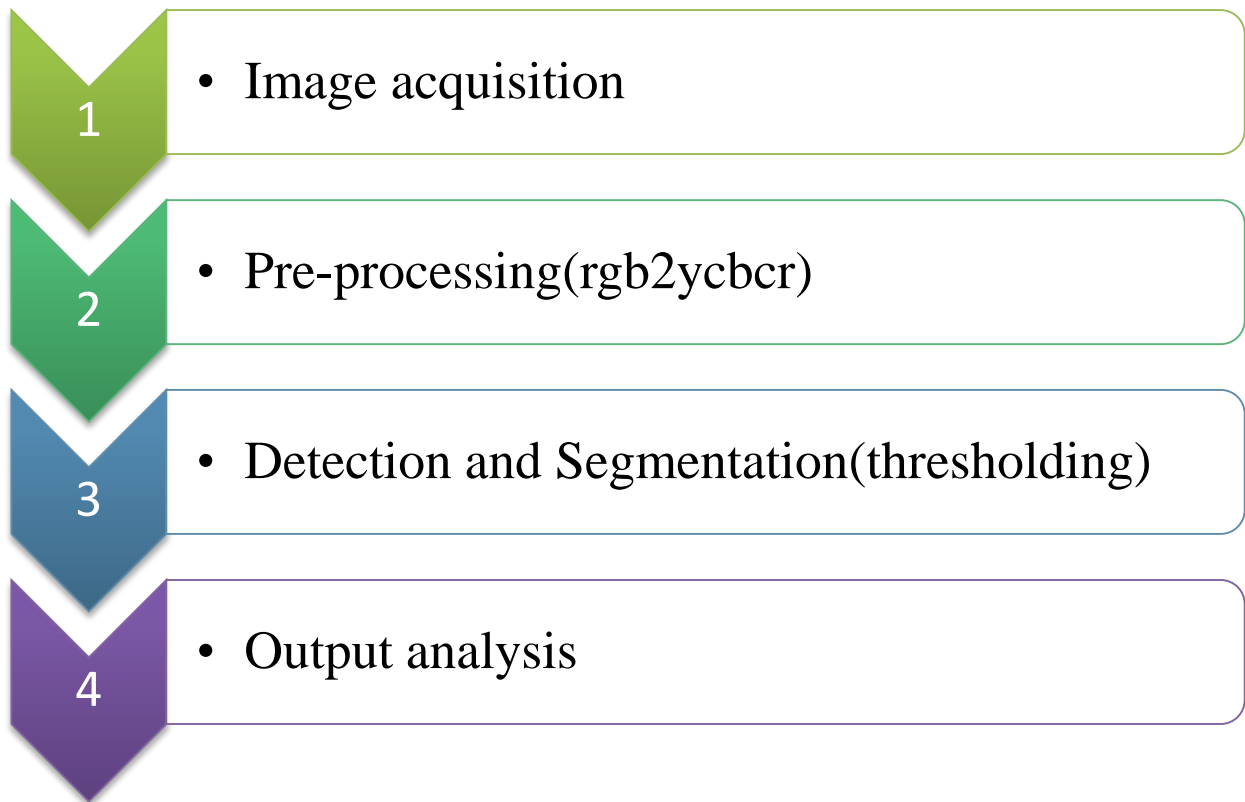
## CHAPTER 4

### 4. Project design and implementation of thresholding method:

#### 4.1. Flow chart:

The above flow chart describes the series of steps taken to carry out the whole process.

Fig 1 Flow Chart



#### 4.2. Image acquisition:

In this a digital image is produced by one or several image sensors, which, besides various types of light sensitive cameras include rate sensors, tomography devices, radar, ultrasonic cameras etc. The resulting image is an ordinary 2-D image, a 3-D volume based on the type of sensor.

We started with the basic idea of differentiating the most ripe tomatoes from a basket full of ripe and unripe tomatoes. We acquired the image from a source so it can be passed through whatever processes need to occur afterwards. The image was an unprocessed one. The image is read using `imread` command. If the file is not in the current folder, or in a folder on the MATLAB path, specify the full pathname.

### **4.3. Pre-processing:**

The process of manipulating the image so that the result is more suitable than the original image for specific application. It includes noise reduction to assure that sensor noise doesn't introduce false information. Small neighbourhood of pixels in the input image is used to get new brightness level in output image.

The next step we took was pre-processing of the image. The aim of pre-processing was improvement of the image data that suppresses unwanted distortions or enhances some image features important for further processing. We converted RGB image to YCbCr image by using command `rgb2YcbCr`. This is done because YCbCr extracts particular features of the RGB image. It basically represents an image on the basis of intensity. The indices of the processed images are stored in `m,n,k`. then equivalent matrix for segmentation is generated.

### **4.4. Detection/ Segmentation:**

The next was the detection phase. In this we did variable thresholding based on image properties. Here we computed a threshold at every point in the image based on one or more specified properties computed in the neighbourhood of the point with the common functions such as logical and arithmetic operations.

#### **4.4.1 General Algorithm of Thresholding:**

- Estimate the optimal threshold in one channel.
- Segment the overall image based on that threshold.
- Subdivide each of these regions independently using properties of the second channel.
- Repeat again for each channel until each region in the image exhibits a distribution indicative of a coherent region (**a single mode**).

The performance of thresholding algorithm is affected by various factors such as noise and non-uniform illumination. There are many ways like image smoothing and edge information that may help but this type of pre-processing is quite impractical or ineffective in solving the problem. To solve this thresholding complexity variable thresholding is done.

#### 4.4.2 Variable thresholding

It involves:

**i. Image partitioning**

- Simplest approach in which image is sub-divided into non-overlapping rectangles.
- Compensates for non-uniformities in illumination or reflectance.
- The size of rectangles is small enough for the uniform illumination of each.

**Benefits:**

- Works well when the background and the objects of interest occupy regions of comparable size.

**Drawbacks:**

- If the size of the object and the background is not comparable then this method fails because of the likelihood of the sub-divisions containing only object or background pixels.

In such situations we use variable thresholding based on local image properties.

The whole process used helped us in separating the riped i.e. red tomatoes from the rest part of the image. The processed image consisted of only the red tomatoes. The for loop is used such that  $i$  varies from 1 to  $m$ (rows) and  $j$  varies from 1 to  $n$ (columns). This is basically done to consider every pixel for thresholding. Thresholding values are given in the manner discussed; the red colour index is 0.5176, therefore we multiplied the value  $0.5176 * 255 = 130$ , and thus got the critical value of red. In the same way blue has an index of 0.627, therefore the critical value of blue comes out to be 160 and below. Similarly for green colour index value is 0.825, therefore the critical value becomes 210 and below. All these values helped us in setting thresholding conditions to distinguish between most ripe and unripe tomatoes. If the conditions are true then object is found and the region becomes black and on the other hand for the regions where the conditions do not satisfy, they turn white.

#### 4.5. Output analysis:

This was basically done to see if we can distinguish the tomatoes based on the colour. The method used is a basic one and is not very reliable because the results obtained can be improved a lot with the use of other methods. Also there can be images where this method will not work in an efficient manner. Our project, not only deals with just the ripe tomatoes but we also have to identify unripe tomatoes at the same time. This cannot be done merely based on the thresholding method we used above. Therefore we had to shift to other more reliable method.

## CHAPTER 5

### 5. Simulation and experimentation:

#### Overview of tools of MATLAB:

- **Imread:**

It reads the image from graphics file.

*Syntax used and description:*

i. `A=imread(filename,fmt)`

It reads a grayscale or color image from the file specified by string filename. If the file is not in the current folder, or in a folder on the MATLAB path, specify the full pathname. The text string fmt specifies format of the file by its standard file extension i.e. if imread cannot find a file named filename, it looks for a file named filename.fmt.

The return value A is an array containing image data. If the file contains a grayscale image, A is an M-by-N array. If the file contains a true color image, A is an M-by-N-by-3 array.

- **rgb2ycbcr:**

**RGB images:** sometimes referred to as a true color image, it is stored by an M-by-N-3 data array that defines red, green and blue color components for each individual pixel. The color of each pixel is determined by combination of red, green and blue intensities stored in each color plane at pixels location. The RGB images are stored as 24 bit images with the red, green and blue components as 8 bits each, in graphic file formats.

**YCbCr image:** also written as  $Y C_B C_R$  is a family of color spaces which represents color as brightness and two color difference signals. Here, Y is the brightness (luma/luminance), Cb is the blue difference i.e. blue minus luma(B-Y), and Cr is red difference i.e. red minus luma (R-Y).

Therefore, the command rgb2ycbcr goes as follows:

*Syntax used and description:*

`YCBCR=rgb2ycbcr(RGB)`, converts the true color image RGB to the equivalent image in YCbCr color space. Here RGB must be a M-by-N-by-3 array.

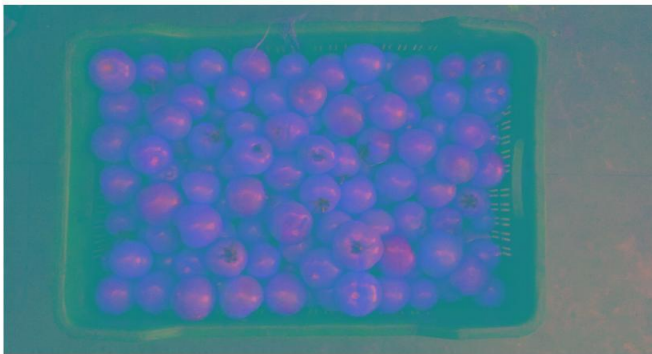
- **ycbcr2rgb:**  
RGB=ycbcr2rgb(YCbCr), converts the YCbCr image to equivalent image in rgb color space.
- **Imshow(I):** it displays the image I where I is a grayscale, RGB or a binary image.
- **Size:**  
*Syntax used and description:*  
[m,n]=size(X), returns the size of matrix X in separate variables m and n.
- **ones(m,n):**  
*Syntax used and description:*  
Y= ones(m,n) returns an m-by-n matrix of ones.
- **figure:** figure graphic objects are created. Figure objects are individual windows on the screen in which the MATLAB software displays graphical output.  
*Figure* creates a new figure object using default property values. This automatically becomes the current figure and raises it above all other figures on the screen until a new figure is created or called.
- **Imwrite:**  
*Syntax used and description*  
Imwrite(A, filename,fmt), it writes image A to the file specified by filename in the format specified by fmt. A can be an M-by-N (grayscale image) or M-by-N-by-3 (truecolor image), but it cannot be an empty array. Filename is a string that specifies the name of the output file and fmt can be any of the test strings listed in **support image types** like 'gif', 'jpg' or 'jpeg', 'tif' or 'tiff' (tagged image file formats) etc.

## Chapter 6

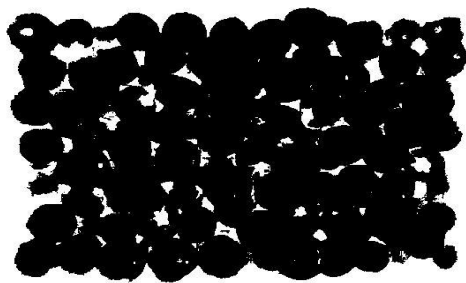
### 6. Simulation results of Thresholding



**Fig. 2 Original Image**



**Fig. 3 pre-processed image**

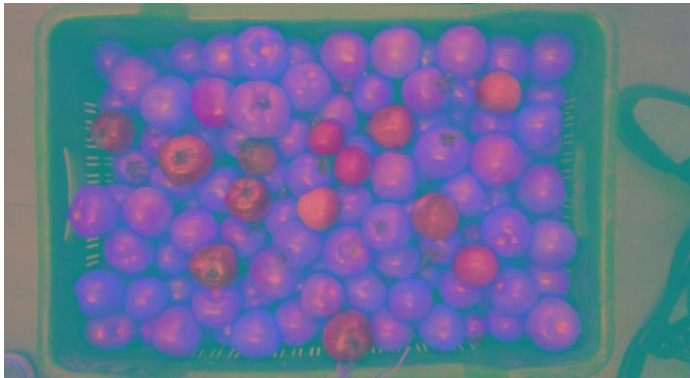


**Fig. 4 processed image**

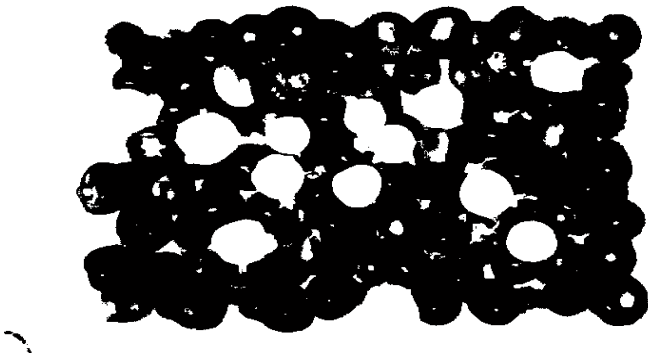




**Fig. 5 Original Image**



**Fig. 6 pre-processed image**



**Fig. 7 processed image**

## CHAPTER 7

### 7. Discussion of results:

Thresholding is basically is user input i.e. it is user defined. We here extracted objects based on user defined inputs. We gave the thresholding values ourselves so as to segment the ripe and unripe tomatoes.

On setting the critical values, which we calculated from the indices of red, green and blue we could finally segment the red and non-red tomatoes. Therefore , setting good critical values helped us segmenting the image well but it is not a reliable method as this thresholding technique only distinguished red tomatoes from other colors. It didn't give any idea about the non-red i.e. green tomatoes as the technique showed the results depicting only the red tomatoes in black color without telling about the green tomatoes and the background details. The technique gave a vague idea about the tomato being ripe and unripe. This is because there is never uniform ripening of a tomato. It is red and greenish in the beginning. For such kind of tomatoes it becomes difficult to give the critical values and therefore it becomes difficult to distinguish ripe and unripe tomatoes.

So, benefits of this method are:

- Separates the pixels in ways that tend to preserve the boundaries.
- Other scattered distributions within the object or the background are irrelevant.

But the drawbacks are:

- Problems if the characteristics change along the boundary.
- Still no guarantee you won't have extraneous pixels or holes.
- The background details are not provided when necessary.

Hence, to overcome the drawbacks and get better results we shifted to k-means clustering.

## CHAPTER 8

### 8. K-means Clustering based image segmentation :

#### 8.1. Clustering methods:

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).

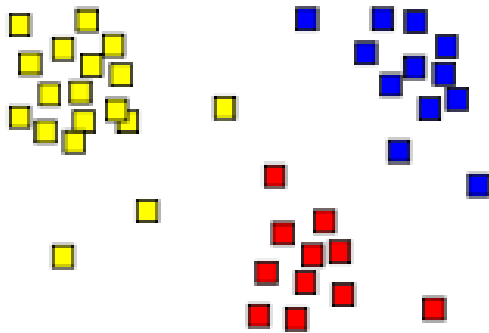
The main goal of clustering is reduction in amount of data by grouping similar data items together. Clustering algorithms basically provide automated tools to help in constructing categories. Also, the effects of human factors are minimized in the process.

Thus, clustering means:

- Organizing data into clusters such that there is
  - High intra-cluster similarity
  - Low inter-cluster similarity
- Finding natural groupings among objects.

Hence, we use clustering because

- It organizes data into clusters to show the internal structure of data.
- Sometimes the goal is partitioning.
- It is useful in knowledge discovery in data.



**Fig. 8 clustering**

### 8.1.1 TYPES OF CLUSTERING

There are basically two types of clustering methods:

i. **Hierarchical clustering:**

In this approach either large clusters are splitted into small ones or small clusters are merged into large ones. In the end this algorithm results in a tree of clusters called a dendrogram, which shows the relation between the clusters. This dendrogram can be cut at desired level to obtain disjoint groups from the clustering of data items.

ii. **Partitional clustering:**

In this approach the data set is directly decomposed in set of disjoint clusters. Here we have a large data set which is carved according to the association between the items in the set

**K-means clustering is a commonly used partitional clustering method.**

Some desirable properties of a clustering algorithm are:

- Scalability( in terms of both time and space)
- Ability to deal with different data types
- Able to deal with noise and outliers
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Interpretability and usability.

Partitioning clustering is non-hierarchical with each instance placed in exactly one of k non-overlapping clusters. The user inputs the desired no. of k clusters since only one set of clusters is output.

### 8.2.K-means clustering:

- It is a partitioning method.
- The function k-means partitions data into k mutually exclusive clusters, and returns the index of the cluster to which it has assigned each observation.
- K-means clustering operates on actual observations and creates a single level of clusters unlike hierarchical clustering. That is for large amounts of data k-means are more suitable than hierarchical.
- Each observation in the data is treated as an object having a location in space.
- The partition is such that objects within each cluster are as close to each other as possible and as far from the objects in other clusters as possible. The distance measures can be chosen depending on the kind of data we are clustering.
- The K-Means method is numerical, unsupervised, non-deterministic and iterative.
- The K-means algorithm is an iterative technique that is used to partition an image

into  $K$  clusters.

- It is relocation method.

### **8.2.1 Properties:**

- There are always  $k$  clusters.
- There is always at least one item in each cluster.
- The clusters are non-hierarchical and they do not overlap.
- Every member of a cluster is closer to its cluster than any other cluster because closeness does not always involve the 'center' of clusters.

## CHAPTER 9

### 9. Project design and implementation:

#### 9.1.Flow chart:

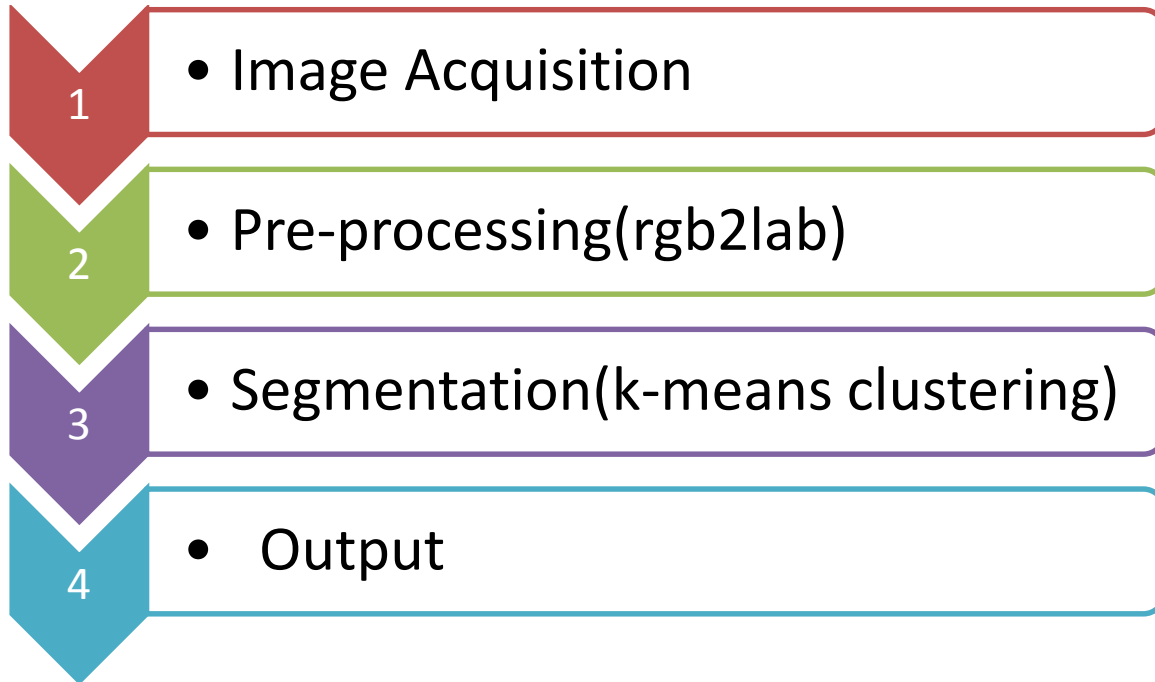


Fig 9 flow chart

#### 9.2.Image acquisition:

In this a digital image is produced by one or several image sensors, which, besides various types of light sensitive cameras include rate sensors, tomography devices, radar, ultrasonic cameras etc. The resulting image is an ordinary 2-D image, a 3-D volume based on the type of sensor.

We started with the basic idea of differentiating the most ripe tomatoes from a basket full of ripe and unripe tomatoes. We acquired the image from a source so it can be passed through whatever processes need to occur afterwards. The image was an unprocessed one. The image is read by the use of command `imread` and stored in variable `t1`. The image is displayed along with the title by the use of command `'imshow', title` and `figure`.

### 9.3.Pre-processing:

The process of manipulating the image so that the result is more suitable than the original image for specific application. It includes noise reduction to assure that sensor noise doesn't introduce false information. Small neighbourhood of pixels in the input image is used to get new brightness level in output image.

The next step we took was pre-processing of the image. The aim of pre-processing was improvement of the image data that suppresses unwanted distortions or enhances some image features important for further processing. We converted the RGB image to L\*a\*b image using `srgb2lab` command. This is done so as to enhance the image for better human vision. The L\*a\*b space consists of a

- i. Luminosity 'L\*' or brightness layer.
- ii. Chromaticity layer 'a\*' indicating where color falls along the red green axis.
- iii. Chromaticity layer 'b\*' indicating where color falls along the blue yellow axis.

Basically L\*a\*b is used for color enhancement but our approach is to choose a small sample region for each colour and to calculate each sample regions average color in 'a\*b' space. The color markers are used to classify each pixel. Actually the image that we see is different from the original view. To make it appear like the actual image L\*a\*b is used. It aspires to perceptual uniformity.

### 9.4.Segmentation/ Detection:

The next was the detection phase. In this we did k-means clustering based on image properties. K-means clustering is an iterative method. This is done because the double data type provides the largest and the smallest possible magnitudes for a number. In k-means clustering basically iterative partitioning minimizes the sum over all clusters, of within cluster sums of point to cluster centroid distance. The indices L\*a\*b space image `t2` are changed to double, as double data type provides the largest and smallest possible magnitudes for a number. It is stored in a variable 'ab'. k-means function is used to check the points whose distance sum is minimum from each and every point present in the cluster. That is why 'distance' is used and it is replicated four times to minimize the sum as it is an iterative method. 'c\_idx' contains the cluster indices of the each point present in a\*b\* space and 'c\_center' contains the centroid location of the clusters. Labelling of every pixel in the image is done using the results from k-means. Now the reshape function used serves the same purpose as described in line 16. Basically, the results we got from k-means for the centroid and cluster indices, according to them the initial image is rendered. This image is black and white, that is because in this part a\*b\* space shows the result on L\* layer and L\* is the luminosity or brightness layer. The value of L varies from 0-100 with 0=black and 100=white. Then the clustered image is shown.

Basically these steps were performed:

- Classify the colors in 'a\*b) space using k-means clustering.
- Repeat the clustering 3 times to avoid local minima.
- Label every pixel in the image using the results from

Classification of data basically assigns corresponding levels with respect to groups with homogenous characteristics. The aim is to discriminate the multiple objects from each other within the image. The level is called class. Classification is executed on the basis of defined features such as texture, color, shape etc. in the feature space. That is, feature space is divided into several classes based on decision rule. Basically there are two types of classification:

- **Supervised classification:** it is important to know spectral characteristics or features with respect to the population of each class, in order to determine a decision rule for classification. Ground based spectrometers can be used for measuring these spectral features but the atmospheric effects hinder direct use of spectral features measured on the ground i.e. spectral features measured on the ground are not always available. Therefore, estimation of population statistics, sampling of training data from the clearly identified training areas corresponding to defined classes is made. This is called supervised classification.
- **Unsupervised classification:** when the information in the area to be classified is less only the image characteristics are used. Clustering technique is used for mechanically dividing the multiple groups into homogeneous spectral classes from a randomly sampled data. Estimation of population statistics is then done through these clustered classes. This classification technique is called unsupervised classification.



## CHAPTER 10

### 10.Simulation and experimentation:

#### Overview of tools of MATLAB:

- **RGB:** A particular **RGB color space** is defined by the three chromaticity of the RED, GREEN, and BLUE.
- **SRGB:-** sRGB defines the chromaticity's of the **RED, GREEN, and BLUE** primaries. The colors where one of the three channels is nonzero and the other two are zero.

The GAMUT of chromaticity's that can be represented in sRGB is the color triangle defined by these primaries. As with any RGB color space, for non-negative values of R, G, and B it is not possible to represent colors outside this triangle, which is well inside the range of colors visible to a human.

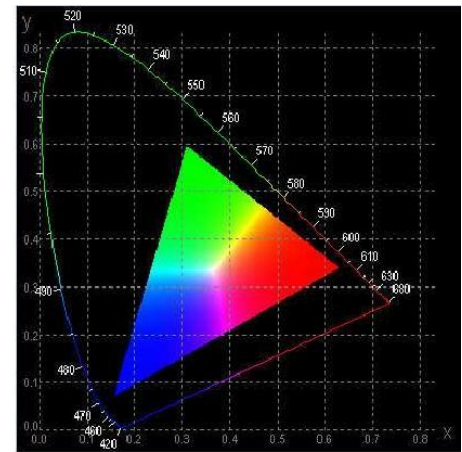


Fig 10 Color space

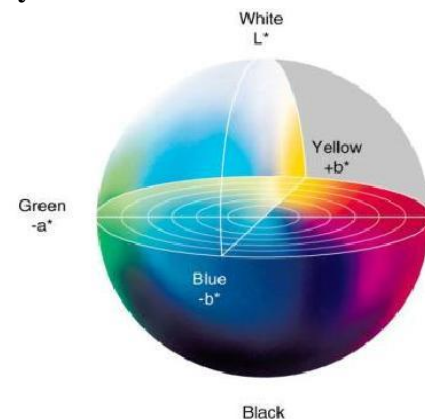
In short, the three colorants and their respective intensities are factors for producing a result of color.

The set of all possible colors is what you call the “GAUMT”.

- **LAB COLOR SPACE:-** The **L\*a\*b\* space** consists of a
  - Luminosity 'L\*' or **brightness layer**
  - Chromaticity layer 'a\*' indicating where color falls along the **red-green axis**
  - Chromaticity layer 'b\*' indicating where the color falls along the **blue-yellow axis**.

The approach is to choose a small sample region for each color and to calculate each sample region's average color in 'a\*b\*' space. We will use these color markers to classify each pixel.

L\*A\*B\* color space in image



- **makecform (type):** - creates the color transformation structure that defines the color space conversion specified by type
- **K-means clustering:**- it is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining.

K-means clustering *aims to partition n observations into k clusters* in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

- **IDX = kmeans(X,k)** partitions the points in the n-by-p data matrix X into k clusters.
  - This iterative partitioning minimizes the sum, over all clusters, of the within-cluster sums of point-to-cluster-centroid distances.
  - Rows of X correspond to points, columns correspond to variables.
- **[IDX,C] = kmeans(X,k):**returns the.
  - n-by-1 vector IDX containing the cluster indices of each point.
  - k cluster centroid locations in the k-by-p matrix C
- **Distance:** in p-dimensional space. Kmeans minimizes with respect to this parameter. Kmeans computes centroid clusters differently for the different supported distance measures.
- **'sqEuclidean':** Squared Euclidean distance (default). Each centroid is the mean of the points in that cluster.
- **'replicates':** Number of times to repeat the clustering, each with a new set of initial cluster centroid positions. kmeans returns the solution with the lowest value for sumd. You can supply 'replicates' implicitly by supplying a 3D array as the value for the 'start' parameter.
- **B = reshape(A,m,n):** returns
  - m-by-n matrix B whose elements are taken column-wise from A.
  - An error results if A does not have m\*n elements.
- **cell(m,n):** Construct cell array
  - □ Creates an m-by-n cell array of empty matrices. An error message appears if n is not a scalar.

```
C = cell(2,2)
```

```
c =
```

```
    []    []  
    []    []
```

```
{ [] – IS A MATRIX }
```

- **B = repmat(A,[m n p]):** Replicate and tile array
  - Produces a multidimensional array B composed of copies of A.

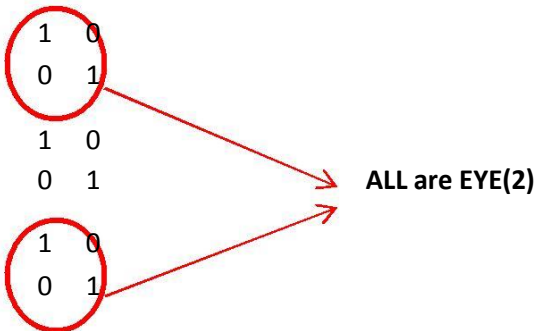
➤ The size of B is  $[\text{size}(A,1)*m, \text{size}(A,2)*n, \text{size}(A,3)*p]$ .

`B = repmat(eye(2),3,4)`

B =

1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1

ALL are EYE(2)



## Simulation results:



**Fig 11 original image**



**Fig 12 labelled image**



**Fig. 13 Clustered image\_1**

objects in cluster 2



**Fig. 14 Clustered image\_2**

objects in cluster 3



**Fig. 15 Clustered image\_3**

## Chapter 11

### 11.1 Discussion of results:

The thresholding method is based on clip level or a threshold value to turn a grayscale image in binary value. The advantages of thresholding includes that it is simple to implement, fast especially if repeating on similar images and moreover it is good for some kind of images like documents, controls, set ups etc but usually the segmentation is not very good and there is no guarantee of object coherency it may have holes, extraneous pixels etc.

On the other hand, k-means method is based on centroid based clustering. With a large number of variables, k-means is computationally faster than hierarchical clustering (if k is small). K-means may produce tighter clusters than hierarchical clustering, if the clusters are globular.

With all the advantages of k-means clustering few disadvantages can also be seen which includes difficulty in comparing quality of clusters produced, fixed number of clusters can make it difficult what k should be. Moreover, we can see that it does not work well with non-globular clusters and different initial partitions can result in different final clusters.

### 11.1 Presentation of results and their analysis:

We performed the k-means clustering by varying the value of number of clusters. Each time the number of clusters were increased , the clustering was done in a better manner and better results were obtained.



Fig. 16.Original image

objects in cluster 1

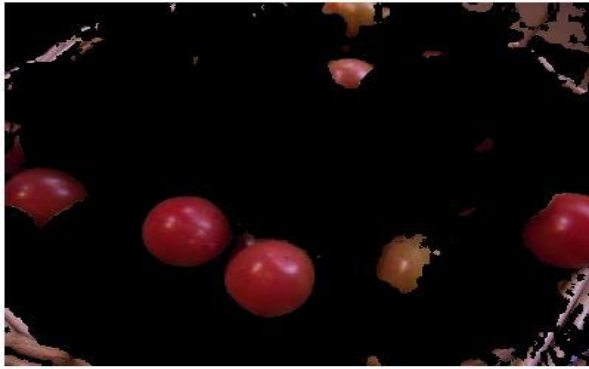


Fig. 17 Clustered image\_1

objects in cluster 2



Fig. 18 Clustered image\_2

As the number of clusters was increased better clustering was performed. Here are the results with cluster value three, five.

objects in cluster 1



Fig. 19 Clustered image\_1

objects in cluster 2



Fig 20 Clustered image\_2

objects in cluster 3



objects in cluster 1

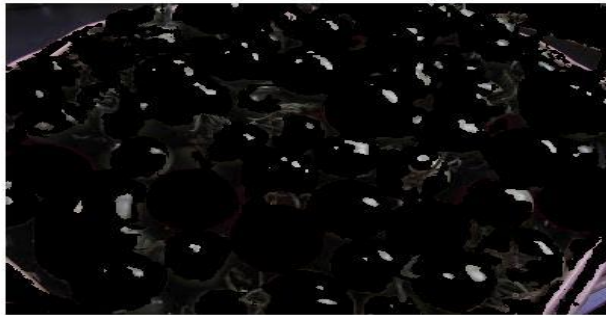


Fig. 21 Clustered image\_1

objects in cluster 2



Fig. 22 Clustered image\_2



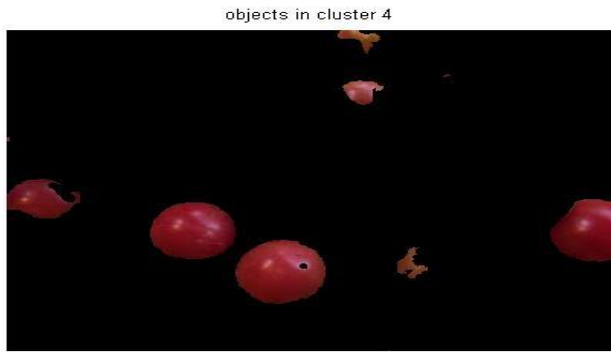


Fig. 23 Clustered image\_4

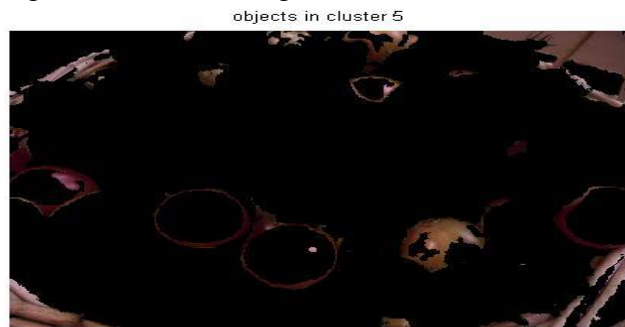


Fig. 24 Clustered image\_5

From the above figures it is clear that on increasing the value of clusters for the same image we get better clustering results.

Apart from tomatoes, we also performed the algorithm on other fruits like apples, bananas. Here are the results for

- Apples



Fig 25 Original image

objects in cluster 1



Fig 26 Clustered image\_1

objects in cluster 2



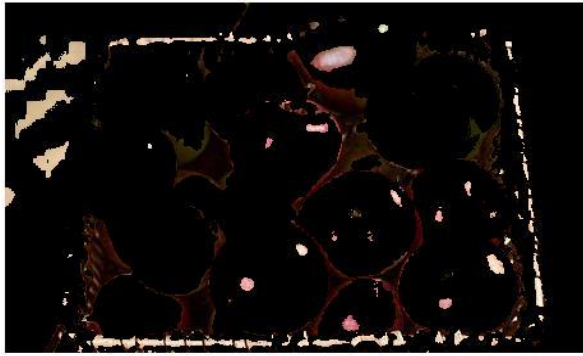
Fig 27 Cluster\_2

objects in cluster 3



Fig 28 Cluster\_3

objects in cluster 4



objects in cluster 5

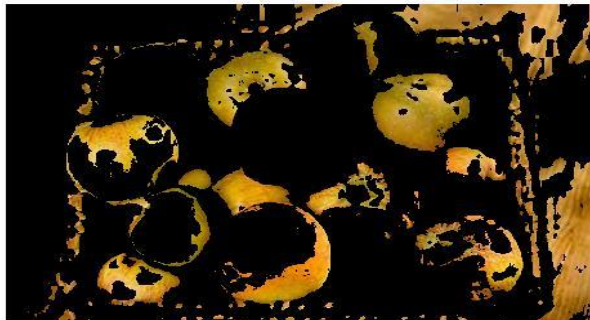


Fig 29 clustered image\_5

### 11.3 Comparison of results:

The results show that in threshold we can only distinguish red tomatoes from green ones as thresholding is user defined and we have to ourselves give the thresholding values. Setting the thresholding values is a difficult task. Whereas in k-means clustering (function defined) images, the three clusters are formed just by setting the value of clusters as three. We can increase the clusters formed by setting different values which enhances the image even further. This shows that this approach is much better than thresholding one as we can distinguish between different colours without much effort and more efficiently. As a result it becomes simpler to check the maturity of the fruit.

### 11.4 Conclusion:

The main aim of our project was to classify the fruits based on maturity taking color as a feature under consideration. Thresholding served the purpose but it was user defined so k-means clustering was done. It is a simple, easy to implement and debug, method. It basically optimized the intra-cluster similarity. Through this project we found out that fruits can be classified based on maturity as ripe and unripe. The results of thresholding based image segmentation were not very reliable. However, k-means clustering provided better results. There are other better methods which can be taken into consideration like fuzzy logic, SVM classification etc.

## 11.5 Code:

- Thresholding based segmentation:

```
1  clc
2  close all
3  clear all
4  %% step 1: image acquisition
5  t1=imread('tom2.jpg');
6  %% step 2: pre-processing
7  x=rgb2ycbcr(t1);
8  figure('Name','Original_image');imshow(ycbcr2rgb(x));
9
10 [M N K]=size(K);
11 y=zeros(M,N);% equivalent matrix for segmentation
12 %% step 3: segmentation/thresholding
13 for i=1:m
14     for j=1:n
15         if x(i,j,2)>80 && x(i,j,2)<130 && x(i,j,3)>160 && x(i,j,3)<210
16             y(i,j)=0;% object found=ve
17         else
18             y(i,j)=1;% object found=unve
19         end
20     end
21 end
22 %% step 4: output analysis
23 figure('Name','Processed_image');imshow(y);
24 imwrite(y,'processed_tom2.jpg');
```

- K-means clustering:

```
1  clc
2  clear all
3  close all
4  %% Step 1: Read Image
5  t1 = imread('tom1.jpg');
6  figure,imshow(t1), title('Tomato image');
7  %% Step 2: Convert Image From RGB Color Space to L*a*b* Color Space
8  cform = makecform('rgb2lab');
9  %makecform(type): creates the color transformation structure that defines the color space conversion specified by type
10 t2 = applycform(t1,cform);
11 %applycform(A,C) converts the color values in A to the color space specified in the color transformation structure C
12 %% Step 3: Classify the Colors in L*a*b* Space Using K-Means Clustering
13 ab = double(t2(:,1:3));
14 nRows = size(ab,1);
15 nCols = size(ab,2);
16 ab = reshape(ab,nRows*nCols,2);
17
18 nColors = 3;
19 % repeat the clustering 3 times to avoid local minima
20 [c_idx c_center] = kmeans(ab,nColors,'distance','sqEuclidean', ...
21     'Replicates',4);
22 %% Step 4: Label Every Pixel in the Image Using the Results From KMEANS
23 pixel_labels = reshape(c_idx,nRows,nCols);
24 figure,imshow(pixel_labels,[]), title('Image labeled by cluster index');
25 %% Step 5: Create Images that Segment the RGB Image By Color.
26 segmented_images = cell(1,3);
27 rgb_label = repmat(pixel_labels,[1 1 3]);
28
29 for k = 1:nColors
30     color = t1;
31     color(rgb_label ~= k) = 0;
32     segmented_images{k} = color;
```

```

File Edit Test Go Cell Tools Debug Display Window Help
Stack: four
This file uses Cell Mode. For information, see the rapid code function video, the publishing video, or help.

7 % Step 2: Convert Image from RGB Color Space to 1*ab* Color Space
8 cform = makecform('rgb2lab');
9 %makecform(type) creates the color transformation structure that defines the color space conversion specified by type
10 t2 = applycform(I,cform);
11 %applycform(A,C) converts the color values in A to the color space specified in the color transformation structure C
12 % Step 3: Classify the Colors in "ab*" Space Using K-Means Clustering
13 ab = double(t2(:,1:2,3));
14 [rows ~] = size(ab,1);
15 [cols ~] = size(ab,2);
16 ab = reshape(ab,rows*cols,2);
17
18 nColors = 3;
19 % repeat the clustering 3 times to avoid local minima
20 [c_idx c_center] = kmeans(ab,nColors,'distance','sqEuclidean', ...
21 'Replicate',4);
22 % Step 4: Label Every Pixel in the Image Using the Results from OSEANS
23 pixel_labels = reshape(c_idx,rows,cols);
24 figure,imshow(pixel_labels,[]), title('image labeled by cluster index');
25 % Step 5: Create Images that Segment the RGB Image by Color.
26 segmented_images = cell(1,3);
27 rgb_label = repmat(pixel_labels,[1 1 3]);
28
29 for k = 1:nColors
30     color = c1;
31     color(rgb_label == k) = 0;
32     segmented_images{k} = color;
33 end
34
35 figure,imshow(segmented_images{1}), title('objects in cluster 1');
36 figure,imshow(segmented_images{2}), title('objects in cluster 2');
37 figure,imshow(segmented_images{3}), title('objects in cluster 3');
38 %figure,imshow(segmented_images{4}), title('objects in cluster 4');

```

script | Ln 30 | Col 16 | OVR

## Chapter 12

### 12.1 Disease detection in Tomato Plant

A large variety of plant diseases are found in tomatoes such as Septoria leaf spot, Cercospora leaf spot, Anthracnose, Fusarium and Verticillium Wilt, Early Blight, Late Blight etc. In our project, we have worked on tomato plant diseases which can be detected through spots on the plant leaves.

The correct and timely identification of diseases is the basis for integrated management of a farm. Generally, the features of any plant or leaf are subjectively extracted by manual inspection. To make this task more efficient and accurate, various studies have been carried out and are undergoing to automate plant identification process with the aid of image processing techniques.

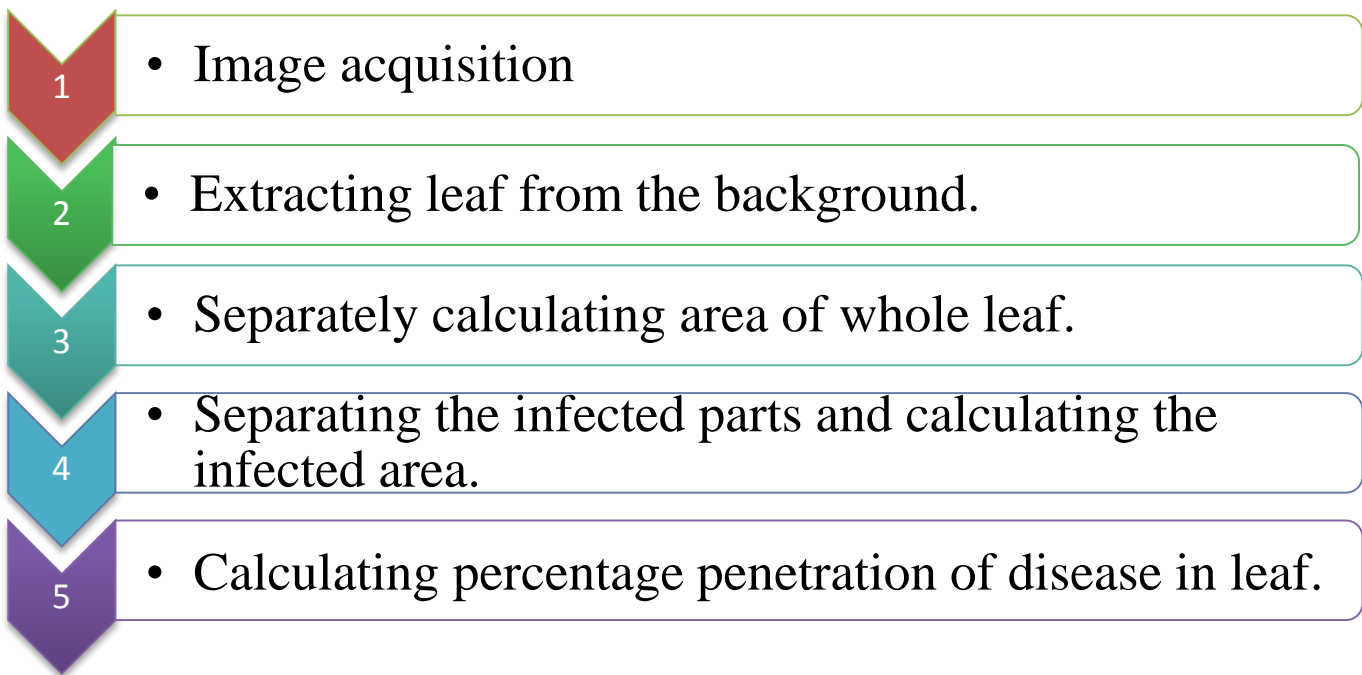
The crop of tomato is very often infected by a disease that leaves spots of brown, gray or off-white colors on the plant's leaves in winter. Scientifically, this disease is known as cercospora leaf spot or cercospora cruciferarum. It's a kind of fungus that often kills young seedlings. The fungus spreads by air and can also infect tomato plants. The size of the fungus and location on leaves give an accurate determination of crop quality under the soil, depending upon the nature and size of the fungus. A criterion is set for acceptable and rejects crop quality based on the fungus level.

Circular to angular brown spots begin occurring on the older, lower leaves in late June and July. The leaves begin to turn bright yellow, orange or red and fall from the tree. As the summer progresses, many infected trees will have dropped all but the newest leaves. Repeated infections year after year will severely stress a tree and reduce its longevity. This fungus colonizes leaf tissue and causes distinct sunken brown spots on leaves. Within the spots, there may be black fruiting structures evident with magnification.



**Fig. 30 Images of Diseased Tomato Leaves**

## 12.2 Flowchart



### 12.2.1 Image Acquisition

In this a digital image is produced by one or several image sensors, which, besides various types of light sensitive cameras include rate sensors, tomography devices, radar, ultrasonic cameras etc. The resulting image is an ordinary 2-D image, a 3-D volume based on the type of sensor. Images of spotted leaves of tomato plants were collected and then taken for further examination.

### 12.2.2 Extracting leaf from the background

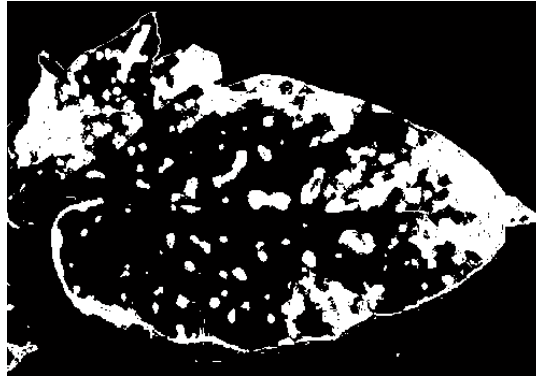
To calculate the area of the whole leaf, initially the blue component was extracted from the RGB image. Then the thresholding was performed and the binary image of the leaf was obtained. Further the leaf area was calculated by finding the sum of pixels lying inside the leaf.



**Fig. 31 Binary\_image**

### **12.2.3 Separating the infected parts and calculating the infected area.**

The RGB image was converted into YCbCr to enhance the features of the image. Then the thresholding algorithm was applied to separate the infected leaf portion from the entire leaf as shown.



**Fig. 32 Processed\_image**

### **12.2.4 Calculating percentage penetration of disease in leaf**

Now the diseased portion was calculated using the command `bwlabel`. Further the percentage of infected leaf was found by dividing the infected portion area by total leaf area.



## Chapter 13

### 13.1 Simulation and experimentation:

#### 13.1.1 Overview of tools of MATLAB:

`bwlabel`- returns the label matrix `L` that contains labels for the 8-connected objects found in `BW`. The label matrix, `L`, is the same size as `BW`.

`[L,num] = bwlabel(___)` returns `num`, the number of connected objects found in `BW`.

#### 13.2 Simulation of Thresholding

```
1 -   clc |
2 -   clear all
3 -   close all;
4 -   im=imread('tom5.jpg');
5 -   R = im(:, :, 1);
6 -   G = im(:, :, 2);
7 -   B = im(:, :, 3);
8 -   [r1 c1]=size(B);
9 -   T=zeros(r1,c1);
10 -  for i=1:r1
11 -  for j=1:c1
12 -      if(B(i,j)>=150 && B(i,j)<=255)
13 -          T(i,j)=0;
14 -      else
15 -          T(i,j)=B(i,j);
16 -      end
17 -  end
18 -  end
19 -  s=0;
20 -  for i=1:r1
21 -  for j=1:c1
22 -      if(T(i,j)==0)
23 -          s=s+1;
24 -      end
25 -  end
26 -  end
27 -  display(s)
28 -  figure,imshow(T);
29 -  x=rgb2ycbcr(imread('tom5.jpg'));
30 -  figure('Name','Original_image');
31 -  imshow(ycbcr2rgb(x));
32 -  figure ('Name','ycbcr_image');
33 -  imshow(x);
```

```

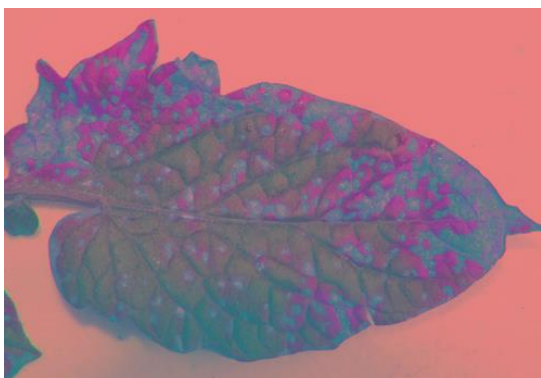
34 - [m n k]=size(x);
35 - y=ones(m,n);
36 - for i=1:m
37 -     for j=1:n
38 -         if (x(i,j,2)>80 && x(i,j,2)<130) && (x(i,j,3)>131 && x(i,j,3)<160)
39 -             y(i,j)=1;
40 -             else
41 -                 y(i,j)=0;
42 -             end
43 -         end
44 -     end
45 - figure('Name','Processed_image')
46 - imshow(y)
47 - imwrite(y,'1.jpg')
48 - STATS=regionprops(y,'ALL');
49 - [L,num] =bwlabel(y);
50 - sum=0;
51 - for i= 1:num
52 -     display(i);
53 -     [r,c]= find(L==i);
54 -     rc=[r,c]
55 -     [x2 y2]=size(rc)
56 -     sum=sum+x2
57 - end
58 - STATS=regionprops(y,'area')
59 - leafarea=(350*500)-s;
60 - penetration=sum/leafarea;
61 - per=penetration*100

```

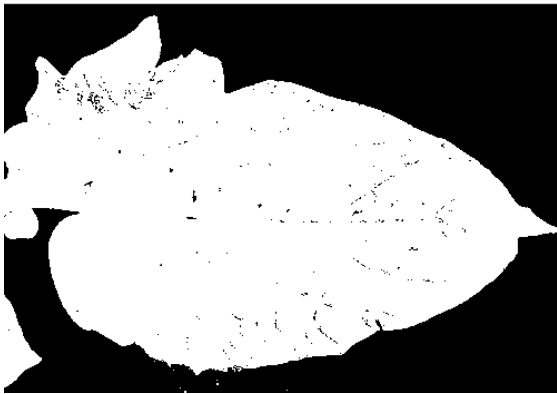
### 13.3 Simulation Results



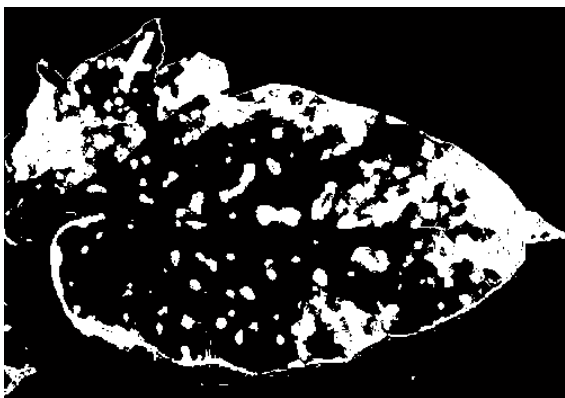
**Fig. 33 Original Image**



**Fig. 34 ycbcr\_image**



**Fig. 35 Binary\_image**



**Fig. 36 Processed\_image**

```
sum =  
    33398  
  
leafarea =  
    98330  
  
penetration =  
    0.3397  
  
per =  
    33.9652  
  
The leaf is more than 1/3rd diseased. Hence plant rejected.  
>> |
```



**Fig. 37 Original Image**

```
per =  
    2.0586  
The plant is accepted.
```



**Fig. 38 Binary\_image**



**Fig. 39 Processed\_image**

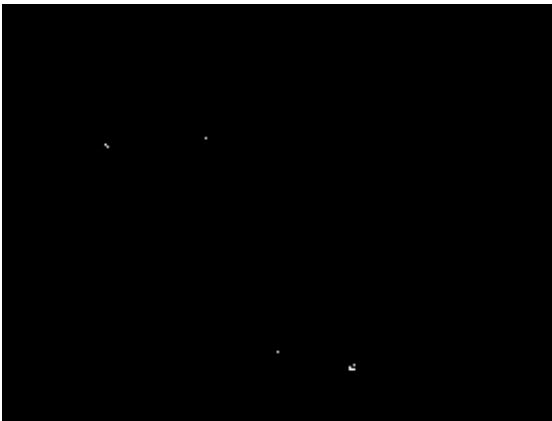


**Fig. 40 Original Image**



```
per =  
  
    0.0606  
  
The plant is accepted.
```

**Fig. 41 Binary\_image**



**Fig.42 Processed\_image**

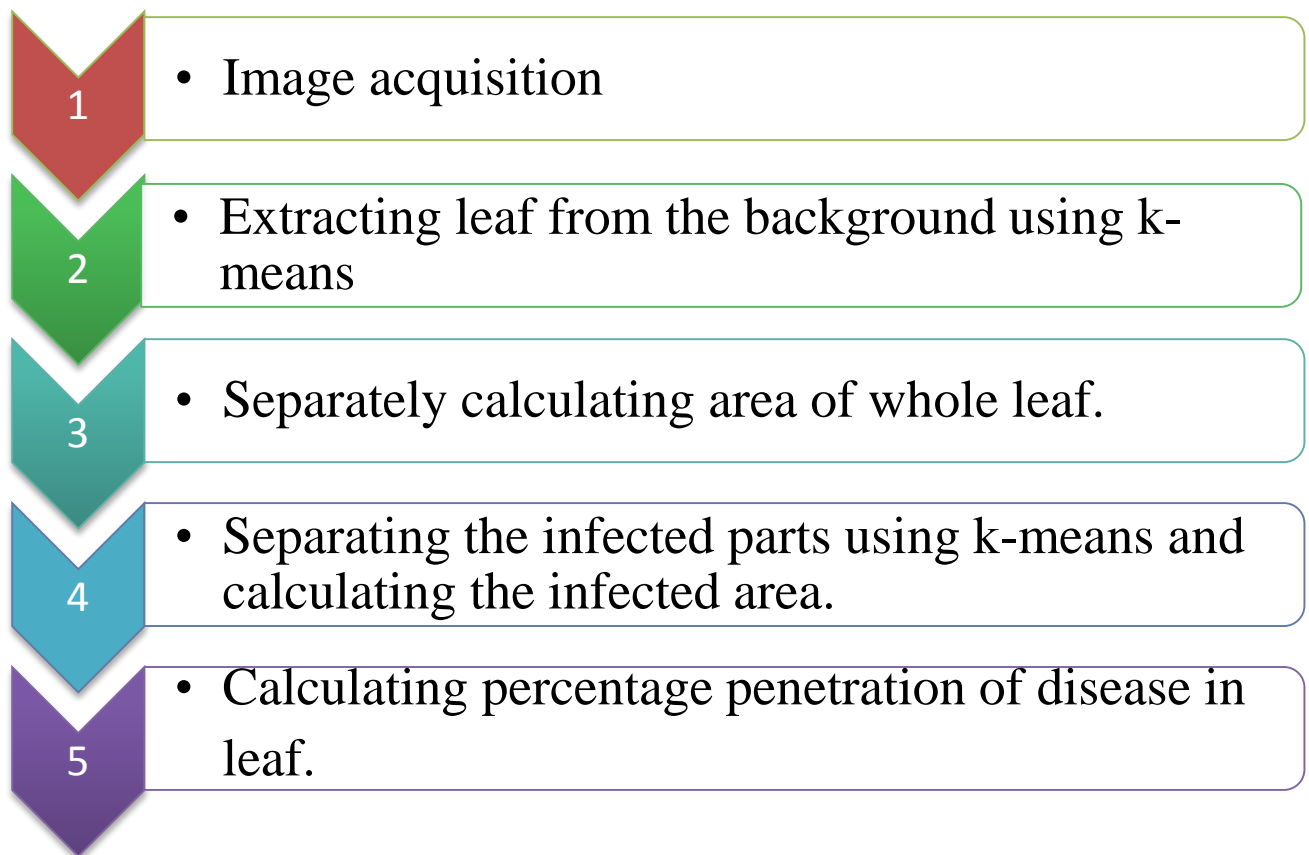
### **13.4 Output Analysis**

Thresholding is basically is user input i.e. it is user defined. We here extracted the infected leaf area from the total leaf. We gave the thresholding values ourselves so as to segment the infected leaf portion. On setting the critical values, which we calculated by varying the indices of red, green and blue we could finally segment the infected leaf. Therefore, setting good critical values helped us segmenting the image well. Since the method was user defined we decided to switch to k-means clustering as done in case of tomato maturity also.

## Chapter 14

**14.1 K-means clustering for detection of infected portion of the leaf**As discussed in section 8, k-means clustering is a function defined method .Given below is the flow chart of the disease detection using k-means clustering:

### 14.2 Flowchart



#### 14.2.1 Image Acquisition

In this a digital image is produced by one or several image sensors, which, besides various types of light sensitive cameras include rate sensors, tomography devices, radar, ultrasonic cameras etc. The resulting image is an ordinary 2-D image, a 3-D volume based on the type of sensor. Images of spotted leaves of tomato plants were collected and then taken for further examination.

#### 14.2.2 Extracting leaf from the background and area calculation

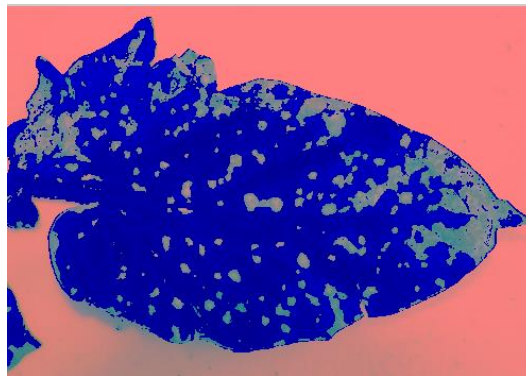
In order to separate the leaf from the background we applied the k-means clustering algorithm. The region of leaf turned to white while the background was black. Thus by calculating the sum of white pixels we could determine the leaf area and by calculating the the black pixels we calculated the background area.



**Fig. 43 Binary\_image**

### **14.2.3 Separation of infected parts using k-means**

For separating the infected parts, k-means clustering was applied to the image with  $l*a*b^*$  space format. By using this format the number of shades of different colors were reduced ,after application of k-means we got several clusters, the first cluster was the one where the infected region was separated from the leaf.this cluster was taken for further calculations.



**Fig. 44 clustered\_image1**

### **14.2.4 Calculation of area of infected region**

The leaf obtained in cluster 1 was converted to black and white and later bwlabel was used to calculate the area of infection.



**Fig. 45 Processed\_image**

#### **14.2.5 Percentage penetration calculation:**

Further the percentage of infected leaf was found by dividing the infected portion area by total leaf area.



## Chapter 15

### 15.1 Simulation of k-means

```
1 -   clc
2 -   clear all
3 -   close all
4 -   %% Step 1: Read Image
5 -   t1 = imread('tom 5.jpg');
6 -   figure,imshow(t1), title('Tomato image');
7 -   %% Step 2: Convert Image from RGB Color Space to L*a*b* Color Space
8 -   cform = makecform('srgb2lab');
9 -   t2 = applycform(t1,cform);
10 -  %% Step 3: Classify the Colors in 'a*b*' Space Using K-Means Clustering
11 -  ab1 = double(t1(:,:,2:3));
12 -  nrows1 = size(ab1,1);
13 -  ncols1 = size(ab1,2);
14 -  ab1 = reshape(ab1,nrows1*ncols1,2);
15 -  nColors = 2;
16 -  [c_idx c_center] = kmeans(ab1,nColors,'distance','sqEuclidean', ...
17 -                          'Replicates',7);
18 -  %% Step 4: Label Every Pixel in the Image Using the Results from KMEANS
19 -  pixel_labels = reshape(c_idx,nrows1,ncols1);
20 -  figure,imshow(pixel_labels,[]), title('image labeled by cluster index');
21 -  %% Step 5: Create Images that Segment the H&E Image by Color.
22 -  segmented_images1 = cell(1,3);
23 -  rgb_labell = repmat(pixel_labels,[1 1 3]);
24 -  [r2,c2]=size(pixel_labels);
25 -  P=pixel_labels;
26 -  s2=0,b2=0;
27 -  for i=1:r2
28 -  for j=1:c2
29 -     if(P(i,j)==1)
30 -         s2=s2+1;
31 -     end
32 - end
33 -
34 - end
35 - leafarea1=(r2*c2)-s2;
36 - for k = 1:nColors
37 -     color = t1;
38 -     color(rgb_labell ~= k) = 0;
39 -     segmented_images1{k} = color;
40 - end
41 - figure,imshow(segmented_images1{1}), title('objects in cluster 1');
42 - figure,imshow(segmented_images1{2}), title('objects in cluster 2');
43 - figure,imshow(segmented_images1{3}), title('objects in cluster 3');
44 - [r1 c1]=size(segmented_images1{1});
45 - T= segmented_images1{1};
46 - ab = double(t2(:,:,2:3));
47 - nrows = size(ab,1);
48 - ncols = size(ab,2);
49 - ab = reshape(ab,nrows*ncols,2);
50 - nColors = 3;
51 - [c_idx c_center] = kmeans(ab,nColors,'distance','sqEuclidean', ...
52 -                          'Replicates',12);
53 - %% Step 4: Label Every Pixel in the Image Using the Results from KMEANS
54 - pixel_labels = reshape(c_idx,nrows,ncols);
55 - figure,imshow(pixel_labels,[]), title('image labeled by cluster index');
```

```

56 %% Step 5: Create Images that Segment the H&E Image by Color.
57 segmented_images = cell(1,3);
58 rgb_label = repmat(pixel_labels,[1 1 2]);
59
60 for k = 1:nColors
61     color = t2;
62     color(rgb_label ~= k) = 0;
63     segmented_images{k} = color;
64 end
65
66 figure,imshow(segmented_images{1}), title('objects in cluster 1');
67 %figure,imshow(rgb2gray(segmented_images{1}));
68 V=segmented_images{1};
69 R = V(:, :, 1);
70 G = V(:, :, 2);
71 B = V(:, :, 3);
72
73 [r3 c3]=size(B);
74 T=zeros(r3,c3)
75 for i=1:r3
76     for j=1:c3
77         if(B(i,j)>=150 && B(i,j)<=255)
78             T(i,j)=0;
79             else
80                 T(i,j)=B(i,j);
81             end
82         end
83     end
84     figure,imshow(T);
85     [L,num] =bwlabel(T);
86     sum=0;
87     for i= 1:num
88         display(i);
89         [r,c]= find(L==i);
90         rc=[r,c]
91         [x2 y2]=size(rc)
92         sum=sum+x2
93     end
94     infectedarea=sum-s2;
95     penetration=infectedarea/leafareal;
96     per=penetration*100;
97     if(per>40)
98         display('The leaf is more than 1/3rd diseased. Hence plant rejected. ');
99     else
100         display('The plant is accepted. ');
101     end
102
103
104 figure,imshow(segmented_images{2}), title('objects in cluster 2');
105 figure,imshow(segmented_images{3}), title('objects in cluster 3');
106

```

## 15.2 Results



Fig. 46 Binary\_image

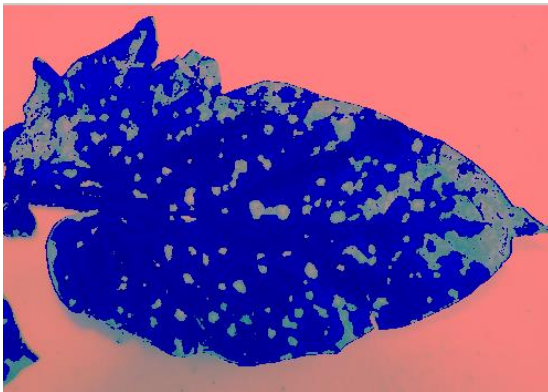
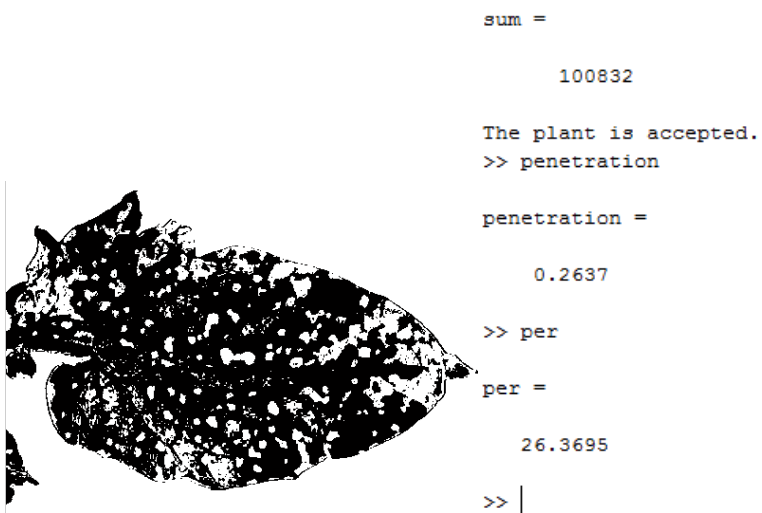
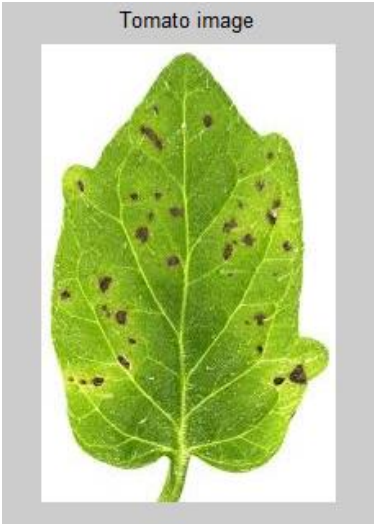


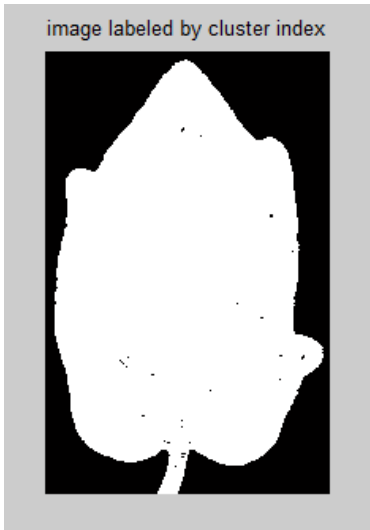
Fig. 47 clustered\_image1

Fig. 48 Processed\_image





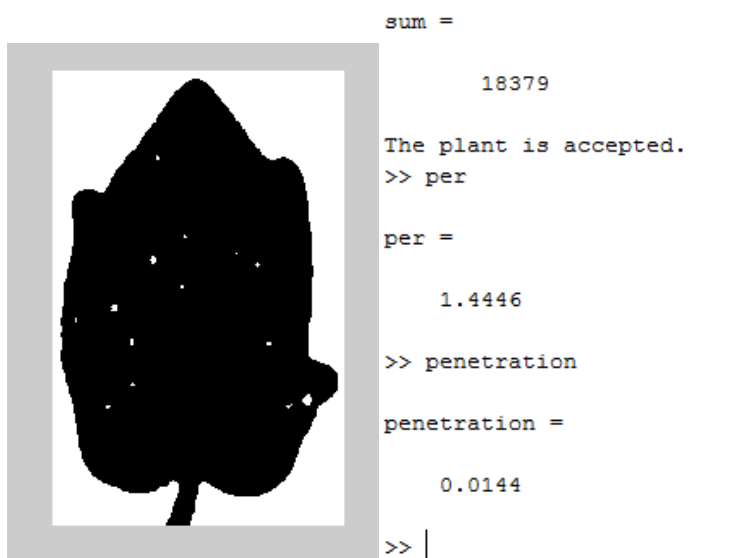
**Fig. 49 Original Image**



**Fig. 50 Binary\_image**

**Fig. 51 clustered\_image1**





**Fig. 52 Processed\_image**

### 15.3 Comparison of results

We have separated the infected portion of the leaf using both thresholding and k-means clustering. It can be clearly seen that the results obtained from the two algorithms are close enough. We switched to k-means clustering because it was because it was function defined method, we did not have to set any clip value, The program worked efficiently for all the leaves without any changes required.

### 15.4 Future Aspects

As we have seen that with advantages of k-means clustering method there are disadvantages too which includes difficulty in comparing quality of clusters produced, fixed number of clusters can make it difficult what k should be. Moreover, we can see that it does not work well with non-globular clusters and different initial partitions can result in different final clusters.

Due to all these problems, we can shift to some other more reliable methods like neural networks. Using neural networks we can work on database having information of large variety of diseased leaves and thus work on other diseases.

## References:

- [1] <http://www.ijeat.org/attachments/File/v2i1/A0803102112.pdf>
  - a. F. Mendoza, J .M. Aguilera, “Application of Image Analysis for Classification of Ripening Bananas”, Journal of Food Science. 69(9), 2004, pp.415-423.
- [2] J. Blasco, N. Aleixos, E., “Molto. Machine Vision System for Automatic Quality Grading of Fruit”, Biosystems Engineering, vol. 85 (4),2003, pp.415-423.
- [3] B. Ojeda-Magana, R. Ruelas, J. Quintanilla-Dominguez and D. Andina, “Colour Image Segmentation by Partitional Clustering Algorithms”, IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society, 7-10 Nov., Glendale, AZ, 2010, pp.2828 -2833.
- [4] Choi K. H., “Tomato maturity evaluation using colour image analysis”, Transactions of the ASAE, Vol. 38(1), 1995, pp. 171-176.
- [5] Chiunheium Lin, ching- Hung Su,Hsuan Shu Huang and Kuo-Chin.
- [6] Fan, “Colour Image Segmentation Using Relative values of RGB in Various Illumination, Circumstances”, International Journal of computers, vol.5(2) , 2011, pp.252-261.
- [7] <http://cs.haifa.ac.il/~dkeren/acv/clustering-ponce-forsyth-16.pdf>.
- [8] <http://www.cs.cmu.edu/afs/andrew/course/15/381-f08/www/lectures/clustering.pdf>
- [9] <http://tim.sagepub.com/content/27/2/65.short>
- [10] <http://www.sciencedirect.com/science/article/pii/S002186349180030I>
- [11] [http://www.unido.org/fileadmin/import/32382\\_fruitsDec21.2.pdf](http://www.unido.org/fileadmin/import/32382_fruitsDec21.2.pdf)
- [12] [http://home.deib.polimi.it/matteucc/Clustering/tutorial\\_html/kmeans.html](http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html)
- [13] <http://www.cs.cmu.edu/afs/andrew/course/15/381f08/www/lectures/clustering.pdf>
- [14] [http://www.academicjournals.org/article/article1381133860\\_Gastelum-Barrios%20et%20al.pdf](http://www.academicjournals.org/article/article1381133860_Gastelum-Barrios%20et%20al.pdf)
- [15] <http://www.sciencedirect.com/science/article/pii/S016816990000171X>
- [16] <http://www.bhg.com/gardening/vegetable/vegetables/tomato-plant-diseases>
- [17] Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features S. Arivazhagan, R. Newlin Shebiah\*, Ananthi, S. Vishnu Varthini March, 2013 Agric Eng Int: CIGR Journal Open
- [18] <https://www.google.co.in/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=diseased%20tomato%20leaves>

## Image References:

- [1] Flow Chart (10)
- [2] Original Image (15)
- [3] Pre-processed image(15)
- [4] Processed image (15)
- [5] Original Image (16)
- [6] Pre-processed image (16)
- [7] Processed image (16)
- [8] Clustering (18)
- [9] Flow chart (20)
- [10] Color space (23)
- [11] Original image (27)
- [12] Labelled image (27)
- [13] Clustered image\_1 (27)
- [14] Clustered image\_2 (28)
- [15] Clustered image\_3 (28)
- [16] Original image (29)
- [17] Clustered image\_1 (30)
- [18] Clustered image\_2 (30)
- [19] Clustered image\_1 (30)
- [20] Clustered image\_2 (31)
- [21] Clustered image\_1 (31)
- [22] Clustered image\_2 (31)
- [23] Clustered image\_4 (32)
- [24] Clustered image\_5 (32)
- [25] Original image (32)
- [26] Clustered image\_1 (33)
- [27] Clustered image\_2 (33)
- [28] Clustered image\_3 (33)
- [29] Clustered image\_5 (34)
- [30] Images of Diseased Tomato Leaves (37)
- [31] Binary\_image (39)
- [32] Processed\_image (39)
- [33] Original Image (41)
- [34] Ycbr\_image (41)
- [35] Binary\_image (42)
- [36] Processed\_image (42)
- [37] Original Image (43)
- [38] Binary\_image (43)
- [39] Processed\_image (43)
- [40] Original Image (44)
- [41] Binary\_image (44)
- [42] Processed\_image (44)

[43]	Binary_image	(46)
[44]	Clustered_image1	(46)
[45]	Processed_image	(46)
[46]	Binary_image	(49)
[47]	Clustered_image1	(49)
[48]	Processed_image	(50)
[49]	Original Image	(50)
[50]	Binary_image	(50)
[51]	Clustered_image1	(51)
[52]	Processed_image	(51)