A

Project Report

On

Leaf Disease Detection Using Image Processing

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR, ANANTHAPURAMU

In partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

S IMRAN	**(21KH1A0478)**
V BHARGAVI	**(21KH1A0490)**
P A BHARATH KUMAR REDDY	**(22KH5A0415)**
C KRISHNA MOHAN	**(22KH5A0405)**

Under the guidance of

Mr. A. M. Shafeeulla, **M. Tech., (Ph.D.)**

Assistant Professor, Department of ECE.

SVCE KADAPA

EDUCATION FOR A BETTER SOCIETY

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SRI VENKATESWARA COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

Balaji Nagar, KADAPA– 516003

2020-2024

SRI VENKATESWARA COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

Balaji Nagar, KADAPA– 516003

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SVCE KADAPA

EDUCATION FOR A BETTER SOCIETY

CERTIFICATE

This is to certify that the project report entitled **"Leaf Disease Detection Using Image Processing"** *bonafide record of the project work done and submitted by*

S IMRAN	**(21KH1A0478)**
V BHARGAVI	**(21KH1A0490)**
P A BHARATH KUMAR REDDY	**(22KH5A0415)**
C KRISHNA MOHAN	**(22KH5A0405)**

for the partial fulfillment of the requirements for the award of B.Tech Degree in **ELECTRONICS AND COMMUNICATION ENGINEERING**, *JNT University Anantapur, Ananthapuramu.*

GUIDE **Head of the Department**

External Viva-Voce Exam Held On_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We here by declare that the project report entitled "**Leaf Disease Detection Using Image Processing"** submitted to the department of requirement for the award of the degree of **BACHELOR OF TECHNOLOGY.** This project is the result of our own effort and that it has not been submitted to any other university or institution for the award of any degree or diploma other than specific above.

Submitted by:

S IMRAN	**(21KH1A0478)**
V BHARGAVI	**(21KH1A0490)**
P A BHARATH KUMAR REDDY	**(22KH5A0415)**
N SAI TEJESWAR REDDY	**(22KH5A0405)**

ACKNOWLEDGEMENT

We are thankful to our guide **Mr. A. M. Shafeeulla,** M.Tech., (Ph.D.), Assistant professor for his valuable guidance and encouragement. His helping and suggestions have helped us in the successful completion of the project.

We would like to express our gratefulness and sincere thanks to **Dr. A. Valli Basha,** M. Tech., Ph.D., Head of the department of ELECTRONICS AND COMMUNICATION ENGINEERING, for his kind help and encouragement during the course of our study and in the successful completion of the project work.

We have great pleasure in expressing our hearty thanks to our beloved Principal **Dr. R. Veera Sudarsana Reddy,** M. Tech., Ph.D., for spending his valuable time with us to complete this project.

Successful completion of any project cannot be done without proper support and encouragement. We sincerely thanks to the **Management** for providing all the necessary facilities during the course of study.

We would like to thank our parents and friends, who have the greatest contributions in all our achievements, for the great care and blessings in making us successful in all our endeavors.

Submitted by:

S IMRAN	**(21KH1A0478)**
V BHARGAVI	**(21KH1A0490)**
P A BHARATH KUMAR REDDY	**(22KH5A0415)**
N SAI TEJESWAR REDDY	**(22KH5A0405)**

CONTENT

ABSTRACT

Agriculture is a key source of livelihood. Agriculture provides employment opportunities for village people on large scale in developing country like India. India's agriculture is composed of many crops and according to survey nearly 70% population depends on agriculture. Most of Indian farmers are adopting manual cultivation due to lagging of technical knowledge. Farmers are unaware of what kind of crops that grows well on their land. When plants are affected by heterogeneous diseases through their leaves that will affect on production of agriculture and profitable loss. Also, reduction in both quality and amount of agricultural production. Leaves are important for fast growing of plant and to increase production of crops. Identifying diseases in plants leave is challenging for farmers also for researchers. Currently farmers are spraying pesticides to the plants but it effects human directly or indirectly by health or also economically. To detect these plant diseases many fast techniques, need to be adopt. In this paper, we have done survey on different plants disease and various advance techniques to detect these diseases.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

DIP	Digital Image Processing
CBIR	Content-Based Image Retrieval
CNN	Convolutional Neural Network
GCM	Grayscale Co-occurrence Matrix
MCCM	Modified Co-occurrence Matrix
HSV	Hue, Saturation, and Value
RGB	Red, Green, Blue
SVM	Support Vector Machine
JPEG	Joint Photographic Experts Group
PNG	Portable Network Graphics
GIF	Graphics Interchange Format
HIS	Hue, Intensity, Saturation
RMS	Root Mean Square
IDM	Inverse Difference Moment
K-Means	Clustering Algorithm
RAM	Random Access Memory
SD	Standard Deviation
GUI	Graphic User Interface

CHAPTER 1
INTRODUCTION

Indian economy is dependent of agricultural productivity. Over 70% of rural homes depend on agriculture. Agriculture pays about 17% to the total GDP and provides employment to over 60% of the population. Therefore, detection of plant diseases plays a vital key role in the arena of agriculture. Indian agriculture is composed of many crops like rice, wheat. Indian farmers also grow sugarcane, oilseeds, potatoes and non-food items like coffee, tea, cotton, rubber. All these crops grow based on strength of leaves and roots. There are things that lead to different disease for the plant leaves, which spoiled crops and finally it will effect on economy of the country. These big losses can be avoided by early identification of plant diseases. Accurate detection of plant disease is needed to strengthen the field of agriculture and economy of our country. Various types of Disease kill leaves in a plant. Farmers get more difficulties in identifying these diseases, they are unable to take precaution on those plants due to lack of knowledge on those diseases. Biomedical is one of the fields to detect plant diseases.

In current day among this field, the image processing methods are suitable, efficient and reliable field for disease detection with help of plant leaf images. Farmers need fast and efficient techniques to detect all types of diseases of plants that can save time. These systems that can reduce efforts and use of pesticides.

1.1 Introduction For Digital Image Processing

Digital Image Processing (DIP) is a field of study that focuses on the manipulation and analysis of images using computer algorithms. It involves techniques such as enhancement, restoration, compression, and segmentation to improve image quality and extract meaningful information. Unlike traditional analog image processing, which relies on optical and electrical methods, digital image processing converts images into numerical data that can be modified and analysed using mathematical operations. This field has gained immense importance with advancements in computing power and artificial intelligence, making it a key component in various technological applications.

With the rapid growth of digital media and imaging technologies, digital image processing is widely used in fields such as medical imaging, satellite remote sensing, biometrics, industrial automation, and computer vision. It plays a crucial role in tasks like facial recognition, object detection, and medical diagnostics, where accurate image interpretation is essential. As research and technology continue to evolve, digital image processing is becoming more sophisticated, enabling real-time applications and automation in diverse industries.

1.1.1 Definition of Digital Image Processing

Digital Image Processing (DIP) is a field of computer science and engineering that involves the manipulation, enhancement, and analysis of digital images using mathematical and algorithmic techniques. It transforms raw image data into a more useful format by applying various processing methods such as filtering, contrast adjustment, noise reduction, and feature extraction. These techniques enable better visualization, interpretation, and automated decision-making in numerous applications, including medical imaging, remote sensing, security, and artificial intelligence.

In digital image processing, images are represented as numerical data composed of pixels, where each pixel carries information about color and intensity. Processing these images involves modifying pixel values to achieve desired effects, such as sharpening, blurring, or detecting edges. Advanced methods like machine learning and deep learning further enhance the capabilities of DIP, allowing for object recognition, image segmentation, and real-time processing. By leveraging powerful computational tools, digital image processing continues to play a vital role in various industries, improving accuracy, efficiency, and automation.

An image is a two-dimensional picture, which has a similar appearance to some subject usually a physical object or a person.

Image is a two-dimensional, such as a photograph, screen display, and as well as a three-dimensional, such as a statue. They may be captured by optical devices—such as cameras, mirrors, lenses, telescopes, microscopes, etc. and natural objects and phenomena, such as the human eye or water surfaces.

The word image is also used in the broader sense of any two-dimensional figure such as a map, a graph, a pie chart, or an abstract painting. In this wider sense, images can also be rendered manually, such as by drawing, painting, carving, rendered automatically by printing or computer graphics technology, or developed by a combination of methods, especially in a pseudo-photograph.

Fig 1.1.1(a) General image

An image is a rectangular grid of pixels. It has a definite height and a definite width counted in pixels. Each pixel is square and has a fixed size on a given display. However different computer monitors may use different sized pixels. The pixels that constitute an image are ordered as a grid (columns and rows); each pixel consists of numbers representing magnitudes of brightness and color.

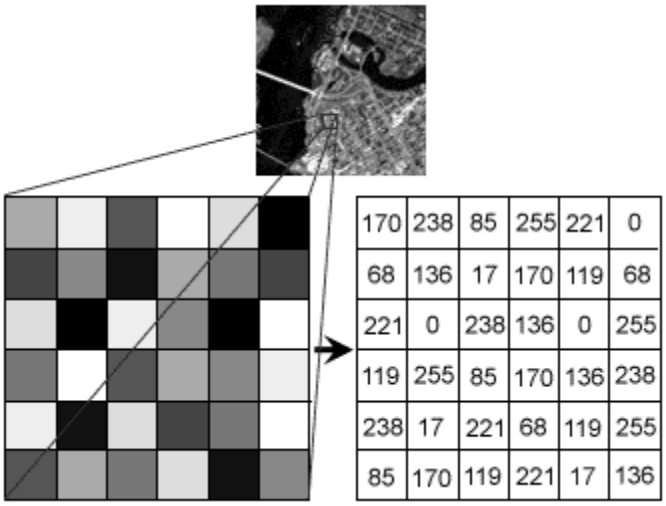


Fig 1.1.1(b) Image pixel

Each pixel has a color. The color is a 32-bit integer. The first eight bits determine the redness of the pixel, the next eight bits the greenness, the next eight bits the blueness, and the remaining eight bits the transparency of the pixel.

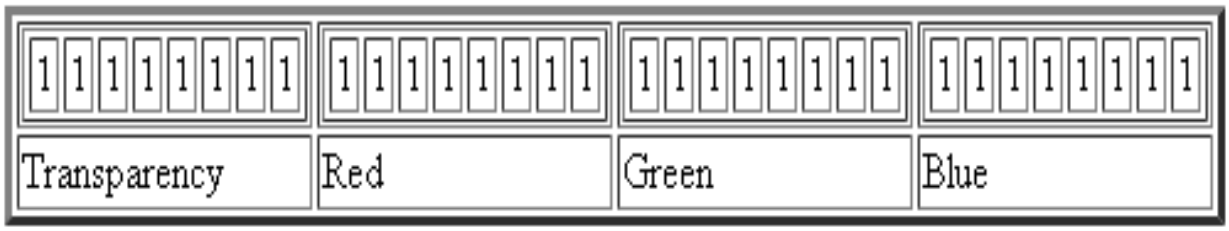


Fig 1.1.1(c) Transparency image

1.1.2 Fundamentals of Digital Image Processing

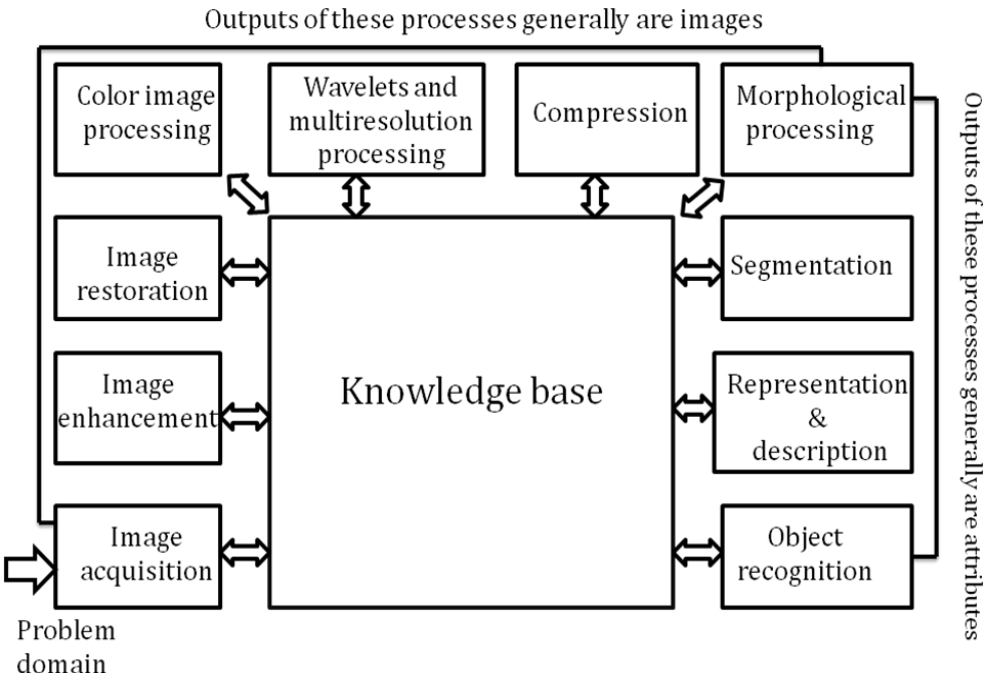


Fig 1.2 Image fundamental

Image Acquisition:

Image Acquisition is to acquire a digital image. To do so requires an image sensor and the capability to digitize the signal produced by the sensor. The sensor could be monochrome or color TV camera that produces an entire image of the problem domain every 1/30 sec. the image sensor could also be line scan camera that produces a single image line at a time. In this case, the objects motion past the line.

Fig 1.2.1 Digital camera image

Scanner produces a two-dimensional image. If the output of the camera or other imaging sensor is not in digital form, an analog to digital converter digitizes it. The nature of the sensor and the image it produces are determined by the application.

Fig 1.2.1(a) digital camera cell

Image Enhancement:

Image enhancement is among the simplest and most appealing areas of digital image processing. Basically, the idea behind enhancement techniques is to bring out detail that is obscured, or simply to highlight certain features of interesting an image. A familiar example of enhancement is when we increase the contrast of an image because "it looks better." It is important to keep in mind that enhancement is a very subjective area of image processing.

Fig 1.2.2 Image enhancement

Image restoration:

Image restoration is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation.

Fig 1.2.3 Image restoration

Enhancement, on the other hand, is based on human subjective preferences regarding what constitutes a "good" enhancement result. For example, contrast stretching is considered an enhancement technique because it is based primarily on the pleasing aspects it might present to the viewer, where as removal of image blur by applying a deblurring function is considered a restoration technique.

Color image processing:

The use of color in image processing is motivated by two principal factors. First, color is a powerful descriptor that often simplifies object identification and extraction from a scene. Second, humans can discern thousands of color shades and intensities, compared to about only two dozen shades of gray. This second factor is particularly important in manual image analysis.

Fig 1.2.4 Color & Gray scale image

Wavelets and multiresolution processing:

Wavelets are the formation for representing images in various degrees of resolution. Although the Fourier transform has been the mainstay of transform-based image processing since the late1950's, a more recent transformation, called the wavelet transform, and is now making it even easier to compress, transmit, and analyze many images. Unlike the Fourier transform, whose basis functions are sinusoids, wavelet transforms are based on small values, called Wavelets, of varying frequency and limited duration.

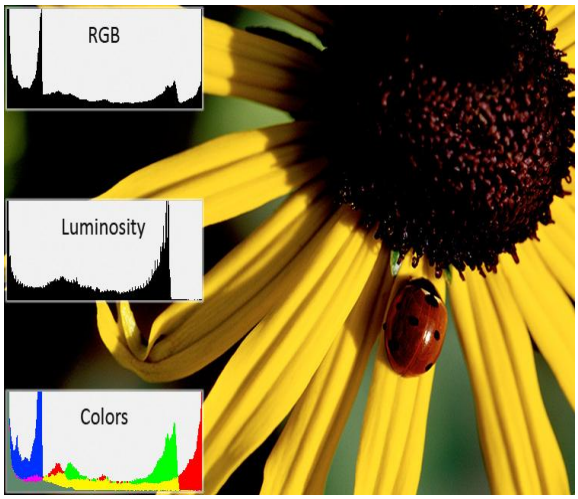


Fig 1.2.5 rgb histogram image

Wavelets were first shown to be the foundation of a powerful new approach to signal processing and analysis called **Multiresolution** theory. Multiresolution theory incorporates and unifies techniques from a variety of disciplines, including sub band coding from signal processing, quadrature mirror filtering from digital speech recognition, and pyramidal image processing.

Compression:

Compression, as the name implies, deals with techniques for reducing the storage required saving an image, or the bandwidth required for transmitting it. Although storage technology has improved significantly over the past decade, the same cannot be said for transmission capacity. This is true particularly in uses of the Internet, which are characterized by significant pictorial content. Image compression is familiar to most users of computers in the form of image file extensions, such as the jpg file extension used in the JPEG (Joint Photographic Experts Group) image compression standard.

Morphological processing:

Morphological processing deals with tools for extracting image components that are useful in the representation and description of shape. The language of mathematical morphology is set theory. As such, morphology offers a unified and powerful approach to numerous image processing problems. Sets in mathematical morphology represent objects in an image. For example, the set of all black pixels in a binary image is a complete morphological description of the image.

Fig 1.2.7 blur to deblur image

In binary images, the sets in question are members of the 2-D integer space Z^2, where each element of a set is a 2-D vector whose coordinates are the (x,y) coordinates of a black(or white) pixel in the image. Gray-scale digital images can be represented as sets whose components are in Z^3. In this case, two components of each element of the set refer to the coordinates of a pixel, and the third corresponds to its discrete gray-level value.

Segmentation:

Segmentation procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually.

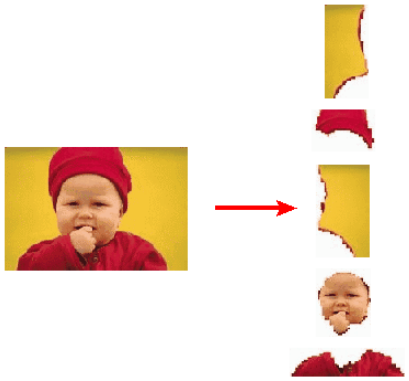


Fig 1.2.8 Image segmentation

On the other hand, weak or erratic segmentation algorithms almost always guarantee eventual failure. In general, the more accurate the segmentation, the more likely recognition is to succeed.

***Representation and description*:**

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region (i.e., the set of pixels separating one image region from another) or all the points in the region itself. In either case, converting the data to a form suitable for computer processing is necessary. The first decision that must be made is whether the data should be represented as a boundary or as a complete region. Boundary representation is appropriate when the focus is on external shape characteristics, such as corners and inflections.

Regional representation is appropriate when the focus is on internal properties, such as texture or skeletal shape. In some applications, these representations complement each other. Choosing a representation is only part of the solution for transforming raw data into a form suitable for subsequent computer processing. A method must also be specified for describing the data so that features of interest are highlighted. Description, also called feature selection, deals with extracting attributes that result in some quantitative information of interest or are basic for differentiating one class of objects from another.

Object recognition:

The last stage involves recognition and interpretation. Recognition is the process that assigns a label to an object based on the information provided by its descriptors. Interpretation involves assigning meaning to an ensemble of recognized objects.

Knowledgebase:

Knowledge about a problem domain is coded into image processing system in the form of a knowledge database. This knowledge may be as simple as detailing regions of an image when the information of interests is known to be located, thus limiting the search that has to be conducted in seeking that information. The knowledge base also can be quite complex, such as an inter related to list of all major possible defects in a materials inspection problem or an image data base containing high resolution satellite images of a region in connection with change deletion application. In addition to guiding the

operation of each processing module, the knowledge base also controls the interaction between modules. The system must be endowed with the knowledge to recognize the significance of the location of the string with respect to other components of an address field. This knowledge glides not only the operation of each module, but it also aids in feedback operations between modules through the knowledge base. We implemented preprocessing techniques using MATLAB.

1.1.3 Overview of Digital Image Processing

Digital Image Processing (DIP) is a field of computer science and engineering that focuses on the analysis, manipulation, and enhancement of digital images using mathematical and algorithmic techniques. It involves converting images into a digital format where pixel values can be processed to improve quality, extract information, or prepare them for further analysis. DIP techniques include image enhancement, restoration, compression, segmentation, and object recognition, which are widely used in various applications such as medical imaging, remote sensing, industrial inspection, and multimedia processing.

One of the key advantages of digital image processing is its flexibility in applying automated and precise modifications to images. With advancements in artificial intelligence and deep learning, modern DIP systems can perform complex tasks like facial recognition, autonomous navigation, and medical diagnostics with high accuracy. The field continues to evolve with improvements in computing power, enabling real-time image processing for applications in security, robotics, and augmented reality. As digital imaging technology progresses, DIP will remain a crucial component in various industries, enhancing efficiency and decision-making capabilities.

1.1.4 Applications of Digital Image Processing

- **Medical Imaging** – DIP is essential in medical diagnostics, where techniques such as MRI, CT scans, and X-ray image enhancement help doctors detect diseases and abnormalities with greater accuracy. Image segmentation and pattern recognition also assist in identifying tumour's, fractures, and other medical conditions.
- **Remote Sensing** – Satellites and aerial imaging use DIP to analyze Earth's surface for environmental monitoring, disaster management, and agricultural assessments. Techniques like image classification and object detection help in mapping and studying geographical changes.
- **Biometrics and Security** – Face recognition, fingerprint scanning, and iris detection rely on image processing algorithms for identification and authentication purposes. These technologies are widely used in surveillance, access control, and law enforcement.
- **Industrial Inspection** – DIP is used in manufacturing for quality control by detecting defects in products using automated visual inspection systems. High-speed cameras and image analysis ensure precision in industries such as electronics, automotive, and pharmaceuticals.
- **Computer Vision and Artificial Intelligence** – Image processing is a fundamental component of AI-driven applications, including autonomous vehicles, robotics, and augmented reality.

It enables object detection, motion tracking, and scene recognition for improved automation and decision-making.

- **Forensics and Crime Investigation** – DIP helps enhance low-quality images from surveillance footage, restore damaged or blurred images, and analyze facial features for forensic investigations. It plays a crucial role in criminal identification and evidence processing.
- **Multimedia and Entertainment** – Image and video enhancement techniques are used in photography, filmmaking, and gaming to improve visual quality. Applications such as image compression, noise reduction, and special effects processing enhance the user experience.
- **Document Processing and OCR** – Optical Character Recognition (OCR) systems use DIP to convert scanned documents into editable text. This is useful for digitizing books, automating data entry, and improving accessibility.

1.1.5 Types of Images

The toolbox supports four types of images:

1. Intensity images;
2. Binary images;
3. Indexed images;
4. R G B images.

Intensity Images:

An intensity image is a data matrix whose values have been scaled to represent intentions. When the elements of an intensity image are of class unit8, or class unit 16, they have integer values in the range [0,255] and [0, 65535], respectively. If the image is of class double, the values are floating point numbers. Values of scaled, double intensity images are in the range [0, 1] by convention.

Binary Images:

Binary images have a very specific meaning in MATLAB.A binary image is a logical array 0s and1s.Thus, an array of 0s and 1s whose values are of data class, say unit8, is not considered as a binary image in MATLAB .A numeric array is converted to binary using function logical. Thus, if A is a numeric array consisting of 0s and 1s, we create an array B using the statement.

B=logical (A)

If A contains elements other than 0s and 1s.Use of the logical function converts all nonzero quantities to logical 1s and all entries with value 0 to logical 0s.

Using relational and logical operators also creates logical arrays.

To test if an array is logical we use the I logical function: islogical(c).

If c is a logical array, this function returns a 1.Otherwise returns a 0. Logical array can be converted to numeric arrays using the data class conversion functions.

Indexed Images:

An indexed image has two components:

A data matrix integer, x

A color map matrix, map

Matrix map is an m*3 arrays of class double containing floating point values in the range [0, 1]. The length m of the map are equal to the number of colors it defines. Each row of map specifies the red, green and blue components of a single color. An indexed images uses “direct mapping” of pixel intensity values color map values. The color of each pixel is determined by using the corresponding value the integer matrix x as a pointer in to map. If x is of class double, then all of its components with values less than or equal to 1 point to the first row in map, all components with value 2 point to the second row and so on. If x is of class units or unit 16, then all components value 0 point to the first row in map, all components with value 1 point to the second and so on.

RGB Image:

An RGB color image is an M*N*3 array of color pixels where each color pixel is triplet corresponding to the red, green and blue components of an RGB image, at a specific spatial location. An RGB image may be viewed as “stack” of three gray scale images that when fed in to the red, green and blue inputs of a color monitor

Produce a color image on the screen. Convention the three images forming an RGB color image are referred to as the red, green and blue components images. The data class of the components images determines their range of values. If an RGB image is of class double the range of values is [0, 1].

Similarly the range of values is [0,255] or [0, 65535].For RGB images of class units or unit 16 respectively. The number of bits use to represents the pixel values of the component images determines the bit depth of an RGB image. For example, if each component image is an 8bit image, the corresponding RGB image is said to be 24 bits deep.

CHAPTER 2
LITERATURE SURVEY

Many researchers had done research on various plants and their diseases also they had given some techniques to identify that disease. To get understanding of this research area, we carry out a study on various types of plants with diseases. This survey will help to propose novel idea for identification of diseases. A. DIFFERENT TYPES OF PLANT DISEASE The reason for this section is that researchers can understand type of image processing operation and type of feature need to be considered by observing various diseases. Disease to the plants that takes place when a virus, bacteria infects a plant and disorders its normal growth. Effect on plant leaves can vary from discoloration to death. Disease causes due to including fungi, microbes, viruses, nematodes. Here we are discussing some common diseases in Maize, arecanut, coconut trees, Papaya, Cotton, Chilli, Tomato, Brinjal. The images of plant disease are shown in Fig.1. Several variants of Diseases are explained further.

- Rust: It is usually found on leaves lower surfaces of mature plants. Initially raised spots on the undersides of leaves. As time passes these spots become reddish orange spore masses. Later, leaf pustules turn to yellow green and eventually black. Severe infestations will bend and yellow leaves and cause leaf drop[2].
- Kole Roge: It is a major disease of arecanut. The pathogen is a fungus Phytophthora palmivora[3].

- Yellow leaf disease: This disease caused by pathogen Phytoplasma in arecanut where green leaves tuning into yellow that gradually decline in yield.

- Leaf rot: It is caused in coconut tree. It is caused by fungi or bacteria. Leaf spot vary in size, shape and colors [4].
- Leaf curl: Disease is characterized by leaf curl. It can cause by fungus, genus Taphrina or virus[5].
- Angular leaf spot: Most of cotton plants die due to this disease because it appears on leaves first then water soaked. Finally turn black and form holes in leaves [6].
- Leaf spot: It is serious bacterial disease found in chili spread by Xanthomonas campestris pv vesicatoria [7]. The symptoms like small yellow green legions and patches on leaves.
- Late Blight: Late Blight spreads rapidly. The development of the fungus due to Cool and wet weather. It forms irregularly shaped ashen spots signs on leaves. Around the spots there will be a ring of white mold [8].
- Bacterial wilt: Brinjal cultivation yield drops due to bacterial wilt. Entire plant has fall down due to wilting of the foliage [9].

  Automation of identifying disease is of great interest in the agriculture field around the world. Many researchers are carried out in detection of those diseases. Below are the various plant disease studies of the techniques used by the researchers in meeting the Endeavor. The survey has been made on the major disease on various plants. Gittaly Dhingra [10] describes application of agriculture using computer

vision technology to recognize and classify disease of plant leaf. The paper deals with correlation between disease symptoms and impact on product yield. It also deals with increase the number of training data and testing to accomplish better accuracy.

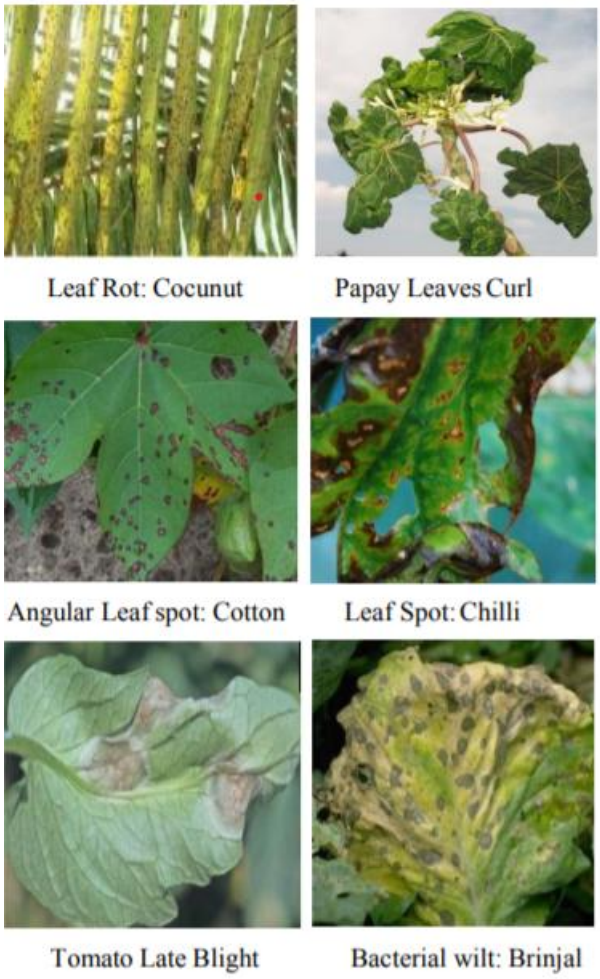


Fig. 2. Plant Diseases in various Plants

Shitala Prasad [11] proposed mobile based client-server design for leaf disease detection using Gabor wavelet transom (GWT). In this system first carried out color conversion from the device dependent to colour space model. Mobile pre-processing can be done after acquiring leaf and converting color space. For

human vision system Human perception*a*b colour space was designed. Making human perception of lightness more accurate by changing output curves in a & b components. To perform analysis of leaf image, K-means unsupervised algorithm was used. To perform feature extraction Gabor wavelet conversion was used. Author of this paper experimented with homemade dataset. In future proposing efficiently processing of Captured Leaf images in a complex background with

different lightening condition. Shanwen Zhang[12], discussed hybrid clustering method. Leaf segmentation is important in detection of plant diseases which affects reliability of feature extraction. Author used super pixel clustering in which neighboring pixels with some feature with respect to brightness, texture, color are grouped into homogeneous regions. This can reduce complexity of images from more pixels. Author suggest that Expectation maximization (EM) algorithm may be good approach for color image segmentation. Keyvan Asefpour Vakilian[13] demonstrate that detect two types of fungus in cucumber plant leaves. ANN model with 3 layers were utilized to identify P.cubensis and S.fuliginea infection. Author has taken real time germinated seeds of cucumber on moist paper which is at degree c for 3 days. Further research is needed to increase the ability of farmers assisted robots in real time detection of fungal and viral disease Mohammed Brahimi [14] proposed deep learning method to create classifier for detection of disease. Also proposed the occlusion concept to localize the disease regions & help to understand the disease. Author uses datasets which is published in good fellow, Bengio etc, further research is need to reduce the computation & size of deep models for small machine like mobiles H.Al-Hiary[15],proposed detection of plant diseases using

automation and classify its diseases. Here pixels are grouped on set of features into total k classes. When leaf has more than one disease then there are more clusters that cause disease. ANN is used to detection and classification of disease. Further research needs to increase the accuracy of detection. Yuanyuan Shao[16] discussed multi feature and genetic algorithm BP neural network. Otsu method was used for segmentation & extraction. Practically in real time tobacco disease can be identified through mobile client and server can make a diagnosis on diseases which

were uploaded by user. Here Otsu method was used to extract spot disease. Genetic algorithm can reduce training times and improve recognition accuracy. Further research needs other method describe tobacco disease feature and to improve accuracy. Vijai Singh[17] presented an algorithm for segmentation of plant leaf image. Author proposed image recognition and segmentation process. First, devices were used to capture image of different types and applied different segmentation method to process image. The author taken image of size m*n & every pixel has R,G,B components. Color co-occurrence method was used for feature extraction. Above experiments are done in MATLAB. Author demonstrates the results only for beans, leaf, lemon and banana leaf. Further research is needed for all types of leaves. Shanwen Zhang[18] proposed method for recognizing disease for cucumber leaves. Due to irregular shapes, complexity, shadows existing classifiers are not suitable for detection. From image of leaf, Author proposed a method using combined shape and color features. Author performed region.

CHAPTER 3

EXISTING METHOD

The traditional approach for detecting plant leaf diseases is based on **naked eye observation**, which requires experts to manually inspect plants. This method is **time-consuming, less accurate, and expensive**, especially in large agricultural fields. Farmers often struggle to recognize unfamiliar diseases, leading to delayed or incorrect treatment. Moreover, **manual monitoring** does not provide early-stage detection, allowing diseases to spread before intervention.

To address these limitations, various **image processing techniques** have been explored, such as **color thresholding, edge detection, and texture analysis**. However, these methods often struggle with **background noise, variations in lighting, and similarities in disease symptoms**, reducing accuracy. Some approaches incorporate **neural networks and fuzzy logic**, but they require large datasets and extensive training for reliable results.

CHAPTER 4

PROPOSED METHOD

CONVERTING RGB TO HSI: The RGB image is in the size of M-by-N-by-3,where the three dimensions account for three image planes(red, green, blue).if all the three components are equal then conversion is undefined. Generally the pixel range of RGB is [0,255] in his the pixel range is [0, 1].Conversion of pixel range can be done by calculating of the components; Hue, Saturation, Intensity.

k-means clustering algorithm: This algorithm is used to cluster/divide the object based on the feature of the leaf in to k number of groups. This is done by using the Euclidean distance metric. The algorithm of k means • Initialization: User should select the value of k. k means the number of clusters/groups, i.e. the image is divided in to k number of clusters. • Every pixel is assigned to its nearest centroid (k). • The position of centroid is changed by means of data values assigned to the group. The centroid moves to the centre of its assigned points. Out of these three clusters classification is done for only one cluster which has affected area.

SUPPORT VECTOR MACHINE (SVM): SVM is a statistical learning-based solver. Statistical is a mathematics of uncertainty.it aims at gaining knowledge, making decisions from a set of data simple classification problem is given in two dimensional input

space. The two types of pattern indicate the images of crescent-shaped and oval objects. We can draw a line separating the two classes and many such possibilities exist. because there is less error may be the optimal line of separation. A line becomes a plane if we have three attributes variables instead of two, and becomes a hyperplane if there are more than three attributes. s what is known as the optimal hyper plane (OH). Another name of OH is maximal margin hyperplane.

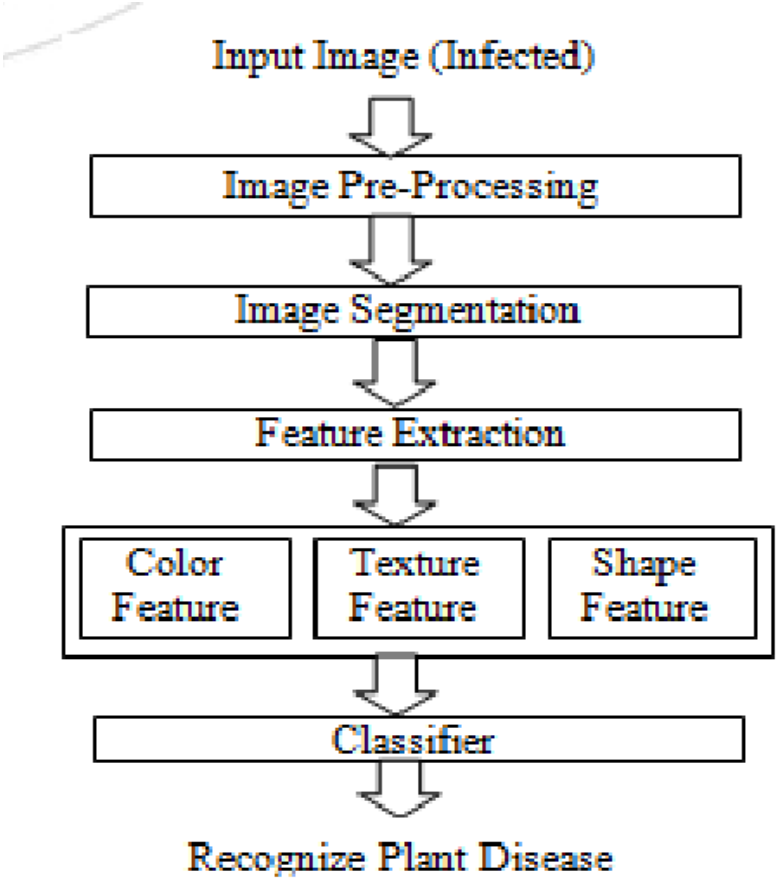


Fig 4. Block Diagram

A. Input Image: We used digital leaf images to identify diseases. The images were taken from different online sources. There are three common rose diseases that we used in this research, i.e., Black spot, Anthracnose and Rust. Fig. 1 shows the disease images in JPEG format.

B. Image Preprocessing: Image pre-process tasks are the initial stage before feature extraction. There are three steps of image preprocessing processing, i.e., image

cropping, image converting and image enhancement. The image is cropped on leaf diseases area, and then converted to gray levels. To enhance the image, we used Laplacian filter.

C. Image Segmentation: Image segmentation is one of the most important precursors for disease detection and has a crucial impact on the overall performance of the developed systems. The K-Means clustering technique is a well-known approach that has been applied to solve low-level image segmentation tasks. This clustering algorithm is convergent and its aim is to optimize the partitioning decisions based on a user-defined initial set of clusters. Paper proposed k-means segmentation method to segment target areas. The area affected by the disease is the target area.

D. Feature Extraction: Proposed method includes two features color texture and space features. These features are total 17 in numbers including 13 color features and 4 shape features. Shape features including area, perimeter, circularity and complexity were extracted from the binary segmentation images. Color features and texture features were extracted from the color segmentation image. The image analysis technique is done using Color Co-occurrence Matrix (CCM).

CHAPTER 5

HARDWARE REQUIREMENTS

1. Image Acquisition Device

- **Digital Camera:**
    - High-resolution camera for capturing detailed images of plant leaves.
    - The camera should be capable of capturing images under different lighting conditions or have controlled lighting to ensure consistency.
    - For field use, a portable and robust camera or smartphone camera may be suitable.
- **Scanner:**
    - For laboratory settings, a flatbed scanner can be used to capture leaf images with high precision.

2. Processing Unit

- **Computer:**
    - A desktop or laptop computer with a multi-core processor (e.g., Intel i5 or equivalent, or better) is essential.
    - The processor speed will significantly impact the processing time for image analysis, K-means clustering, and SVM classification.

- **Embedded System (for portable solutions):**
    - For real-time or in-field applications, an embedded system like a Raspberry Pi or NVIDIA Jetson Nano could be used.
    - These systems are smaller, consume less power, and can be integrated into portable devices.
    - They need to have sufficient processing power and memory to handle the image processing algorithms.

3. Memory (RAM)

- Sufficient RAM is crucial for storing images and intermediate processing data.
- At least 8GB of RAM is recommended for desktop applications.
- For high-resolution images or large datasets, 16GB or more would be beneficial.
- Embedded systems may have more limited RAM, so efficient memory management is important.

4. Storage

- **Hard Disk Drive (HDD) or Solid State Drive (SSD):**
    - A fast and large storage drive is needed to store the image dataset, program files, and processed data.
    - SSD is preferred for faster read/write speeds, which can improve the overall performance of the system.

- **Memory Card (for embedded systems):**
    - Embedded systems often use memory cards (e.g., SD cards) for storage.
    - The storage capacity should be adequate to hold the program and any temporary data.

5. Display

- A high-resolution monitor is needed for displaying the images and the results of the image processing.
- This is more important during the development and testing phase. For a final deployed system, a simple display or even no display might be needed (if the results are sent elsewhere).

6. Graphics Processing Unit (GPU)

- A GPU can significantly accelerate image processing and machine learning tasks.
- This is particularly beneficial for K-means clustering and SVM, especially with large datasets.
- NVIDIA GPUs with CUDA support are commonly used for accelerating these types of computations.

CHAPTER-6

SOFTWARE REQUIREMENTS

6.1. Introduction to MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include

- Math and computation
- Algorithm development
- Data acquisition
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non interactive language such as C or FORTRAN.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most uses of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M – files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

6.1.1 The MATLAB system

The MATLAB system consists of five main parts

- **Development Environment**:

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and command window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.

- **The MATLAB Mathematical Function Library**:

This is a vast collection of computational algorithms ranging from elementary functions, like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix Eigen values, Bessel

functions, and fast Fourier transforms.

- **The MATLAB Language**:

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create large and complex application programs.

- **Graphics:**

MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

- **The MATLAB Application Program Interface (API):**

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

Various toolboxes are there in MATLAB for computing recognition techniques, but we are using **IMAGE PROCESSING** toolbox.

6.1.2 GRAPHICAL USER INTERFACE (GUI)

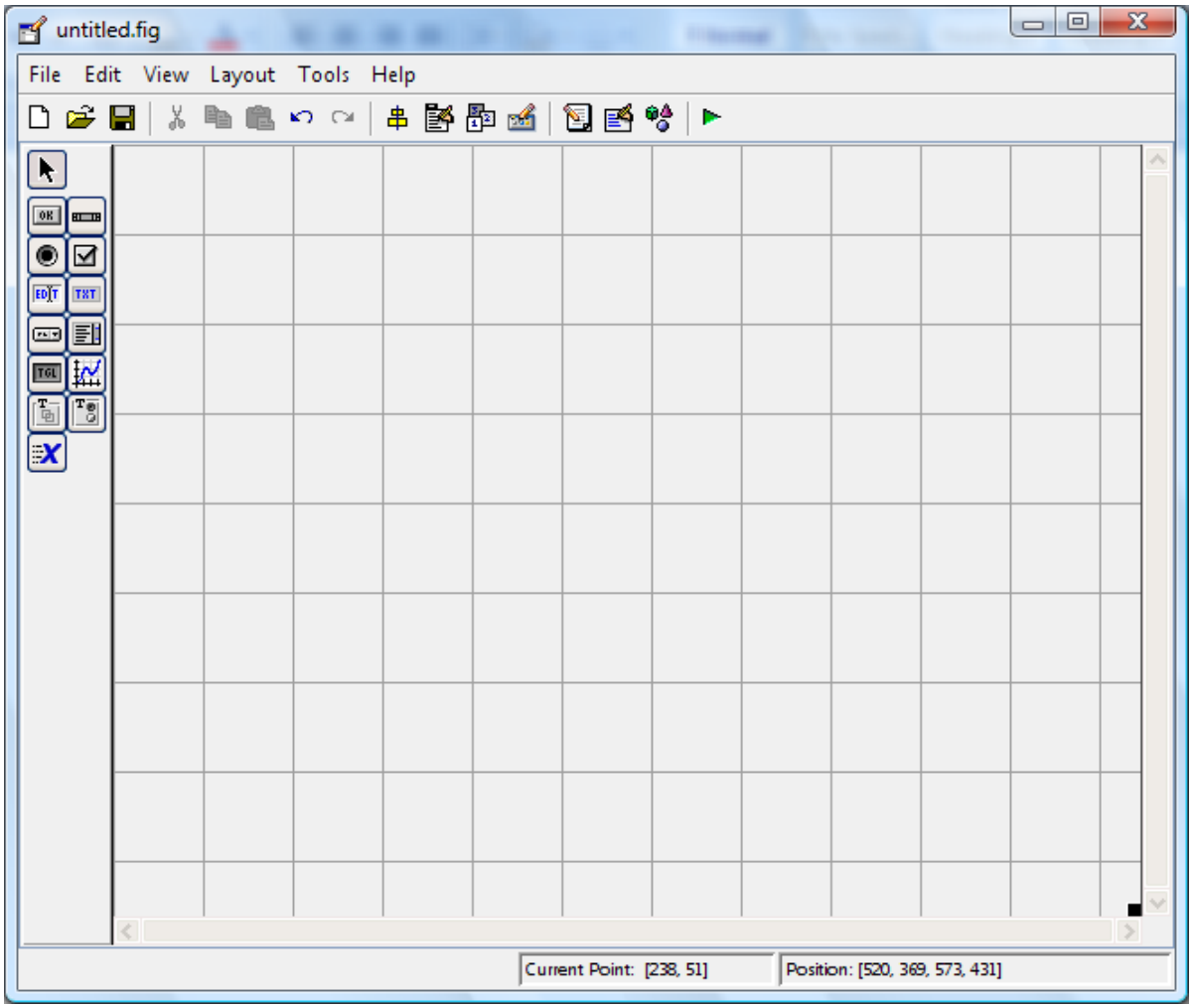
MATLAB's Graphical User Interface Development Environment (GUIDE) provides a rich set of tools for incorporating graphical user interfaces (GUIs) in M-functions. Using GUIDE, the processes of laying out a GUI (i.e., its buttons, pop-up menus, etc.)and programming the operation of the GUI are divided conveniently into two easily managed and relatively independent tasks. The resulting graphical M-function is composed of two identically named (ignoring extensions) files:

- A file with extension .fig, called a FIG-file that contains a complete graphical description of all the function's GUI objects or elements and their spatial arrangement. A FIG-file contains binary data that does not need to be parsed when he associated GUI-based M-function is executed.

- A file with extension .m, called a GUI M-file, which contains the code that controls the GUI operation. This file includes functions that are called when the GUI is launched and exited, and callback functions that are executed when a user interacts with GUI objects for example, when a button is pushed.

To launch GUIDE from the MATLAB command window, type
guide filename

Where filename is the name of an existing FIG-file on the current path. If filename is omitted,
GUIDE opens a new (i.e., blank) window.



A graphical user interface (GUI) is a graphical display in one or more windows containing controls, called components that enable a user to perform interactive tasks. The user of the GUI does not have to create a script or type commands at the command line to accomplish the tasks. Unlike coding programs to accomplish tasks, the user of a GUI need not understand the details of how the tasks are performed.

GUI components can include menus, toolbars, push buttons, radio buttons, list boxes, and sliders just to name a few. GUIs created using MATLAB tools can also perform any type of computation, read and write data files, communicate with other GUIs, and display data as tables or as plots.

6.1.3 Getting Started

If you are new to MATLAB, you should start by reading Manipulating Matrices. The most important things to learn are how to enter matrices, how to use the: (colon) operator, and how to invoke functions. After you master the basics, you should read the rest of the sections below and run the demos.

At the heart of MATLAB is a new language you must learn before you can fully exploit its power. You can learn the basics of MATLAB quickly, and mastery comes shortly after. You will be rewarded with high productivity, high-creativity computing power that will change the way you work.

Introduction - describes the components of the MATLAB system.

Development Environment - introduces the MATLAB development environment, including information about tools and the MATLAB desktop.

Manipulating Matrices - introduces how to use MATLAB to generate matrices and perform mathematical operations on matrices.

Graphics - introduces MATLAB graphic capabilities, including information about plotting data, annotating graphs, and working with images.

Programming with MATLAB - describes how to use the MATLAB language to create scripts and functions, and manipulate data structures, such as cell arrays and multidimensional arrays.

6.1.4 DEVELOPMENT ENVIRONMENT

Introduction

This chapter provides a brief introduction to starting and quitting MATLAB, and the tools and functions that help you to work with MATLAB variables and files. For more information about the topics covered here, see the corresponding topics under Development Environment in the MATLAB documentation, which is available online as well as in print.

Starting and Quitting MATLAB

Starting MATLAB

On a Microsoft Windows platform, to start MATLAB, double-click the MATLAB shortcut icon on your Windows desktop.On a UNIX platform, to start MATLAB, type matlab at the operating system prompt. After starting MATLAB, the MATLAB desktop opens - see MATLAB Desktop.

You can change the directory in which MATLAB starts, define startup options including running a script upon startup, and reduce startup time in some situations.

Quitting MATLAB

To end your MATLAB session, select Exit MATLAB from the File menu in the desktop, or type quit in the Command Window. To execute specified functions each time MATLAB quits, such as saving the workspace, you can create and run a finish.m script.

MATLAB Desktop

When you start MATLAB, the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications

associated with MATLAB.The first time MATLAB starts, the desktop appears as

shown in the following illustration, although your Launch Pad may contain different entries.

You can change the way your desktop looks by opening, closing, moving, and resizing the tools in it. You can also move tools outside of the desktop or return them back inside the desktop (docking). All the desktop tools provide common features such as context menus and keyboard shortcuts.

You can specify certain characteristics for the desktop tools by selecting Preferences from the File menu. For example, you can specify the font characteristics for Command Window text.

Desktop Tools

This section provides an introduction to MATLAB's desktop tools. You can also use MATLAB functions to perform most of the features found in the desktop tools. The tools are:

- Current Directory Browser
- Workspace Browser
- Array Editor
- Editor/Debugger
- Command Window
- Command History
- Launch Pad
- Help Browser

Command Window

Use the Command Window to enter variables and run functions and M-files.

Command History

Lines you enter in the Command Window are logged in the Command History

window. In the Command History, you can view previously used functions, and copy and execute selected lines. To save the input and output from a MATLAB session to a file, use the diary function.

Running External Programs

You can run external programs from the MATLAB Command Window. The exclamation point character! is a shell escape and indicates that the rest of the input line is a command to the operating system. This is useful for invoking utilities or running other programs without quitting MATLAB. On Linux, for example,!emacs magik.m invokes an editor called emacs for a file named magik.m. When you quit the external program, the operating system returns control to MATLAB.

Launch Pad

MATLAB's Launch Pad provides easy access to tools, demos, and documentation.

Help Browser

Use the Help browser to search and view documentation for all your Math Works products. The Help browser is a Web browser integrated into the MATLAB desktop that displays HTML documents.

To open the Help browser, click the help button in the toolbar, or type helpbrowser in the Command Window. The Help browser consists of two panes, the Help Navigator, which you use to find information, and the display pane,

where you view the information.

Help Navigator

Use to Help Navigator to find information. It includes:

Product filter - Set the filter to show documentation only for the products you specify.

Contents tab - View the titles and tables of contents of documentation for your products.

Index tab - Find specific index entries (selected keywords) in the MathWorks documentation for your products.

Search tab - Look for a specific phrase in the documentation. To get help for a specific function, set the Search type to Function Name.

Favorites tab - View a list of documents you previously designated as favorites.

Display Pane

After finding documentation using the Help Navigator, view it in the display pane. While viewing the documentation, you can:

Browse to other pages - Use the arrows at the tops and bottoms of the pages, or use the back and forward buttons in the toolbar.

Bookmark pages - Click the Add to Favorites button in the toolbar.

Print pages - Click the print button in the toolbar.

Find a term in the page - Type a term in the Find in page field in the toolbar and click Go.

Other features available in the display pane are: copying information, evaluating a selection, and viewing Web pages.

Current Directory Browser

MATLAB file operations use the current directory and the search path as reference points. Any file you want to run must either be in the current directory or on the search path.

Search Path

To determine how to execute functions you call, MATLAB uses a search path to find M-files and other MATLAB-related files, which are organized in directories on your file system. Any file you want to run in MATLAB must reside in the current directory or in a directory that is on the search path. By default, the files

supplied with MATLAB and MathWorks toolboxes are included in the search path.

Workspace Browser

The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces.

To view the workspace and information about each variable, use the Workspace browser, or use the functions who and whos.

To delete variables from the workspace, select the variable and select Delete from the Edit menu. Alternatively, use the clear function.

The workspace is not maintained after you end the MATLAB session. To save the workspace to a file that can be read during a later MATLAB session, select Save Workspace As from the File menu, or use the save function. This saves the workspace to a binary file called a MAT-file, which has a .mat

extension. There are options for saving to different formats. To read in a MAT-file, select Import Data from the File menu, or use the load function.

Array Editor

Double-click on a variable in the Workspace browser to see it in the Array Editor. Use the Array Editor to view and edit a visual representation of one- or two-dimensional numeric arrays, strings, and cell arrays of strings that are in the workspace.

Editor/Debugger

Use the Editor/Debugger to create and debug M-files, which are programs you write to runMATLAB functions. The Editor/Debugger provides a graphical user interface for basic text editing, as well as for M-file debugging.

You can use any text editor to create M-files, such as Emacs, and can use preferences (accessible from the desktop File menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as dbstop, which sets a breakpoint.

If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

6.1.5 MANIPULATING MATRICES

Entering Matrices

The best way for you to get started with MATLAB is to learn how to handle matrices. Start MATLAB and follow along with each example.

You can enter matrices into MATLAB in several different ways:

- Enter an explicit list of elements.

- Load matrices from external data files.
- Generate matrices using built-in functions.
- Create matrices with your own functions in M-files.

Start by entering Dürer's matrix as a list of its elements. You have only to follow a few basic conventions:

- Separate the elements of a row with blanks or commas.
- Use a semicolon, ; , to indicate the end of each row.
- Surround the entire list of elements with square brackets, [ ].

To enter Dürer's matrix, simply type in the Command Window

```
A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
```

MATLAB displays the matrix you just entered.

```
A =
    16     3     2    13
     5    10    11     8
     9     6     7    12
     4    15    14     1
```

This exactly matches the numbers in the engraving. Once you have entered the matrix, it is automatically remembered in the MATLAB workspace. You can refer to it simply as A.

Expressions

Like most other programming languages, MATLAB provides mathematical expressions, but unlike most programming languages, these expressions involve entire matrices. The building blocks of expressions are:

- Variables

- Numbers
- Operators
- Functions

Variables

MATLAB does not require any type declarations or dimension statements. When MATLAB encounters a new variable name, it automatically creates the variable and allocates the appropriate amount of storage. If the variable already exists, MATLAB changes its contents and, if necessary, allocates new storage. For example,

num_students = 25

Creates a 1-by-1 matrix named num_students and stores the value 25 in its single element.

Variable names consist of a letter, followed by any number of letters, digits, or underscores. MATLAB uses only the first 31 characters of a variable name. MATLAB is case sensitive; it distinguishes between uppercase and lowercase letters. A and a are not the same variable. To view the matrix assigned to any variable, simply enter the variable name.

Numbers

MATLAB uses conventional decimal notation, with an optional decimal point and leading plus or minus sign, for numbers. Scientific notation uses the letter e to specify a power-of-ten scale factor. Imaginary numbers use either i or j as a suffix. Some examples of legal numbers are

3	-99	0.0001
9.6397238	1.60210e-20	6.02252e23
1i	-3.14159j	3e5i

All numbers are stored internally using the long format specified by the IEEE floating-point standard. Floating-point numbers have a finite precision of roughly 16 significant decimal digits and a finite range of roughly 10-308 to 10+308.

Operators

Expressions use familiar arithmetic operators and precedence rules.

+	Addition
-	Subtraction
*	Multiplication
/	Division
\	Left division (described in "Matrices and Linear Algebra" in Using MATLAB)
^	Power
'	Complex conjugate transpose
( )	Specify evaluation order

Table-6.1.4(a) Operators in MATLAB

Functions

MATLAB provides a large number of standard elementary mathematical functions, including abs, sqrt, exp, and sin. Taking the square root or logarithm of a negative number is not an error; the appropriate complex result is produced automatically. MATLAB also provides many more advanced mathematical functions, including Bessel and gamma functions. Most of these functions accept complex arguments. For a list of the elementary mathematical functions, type help elfun, For a list of more advanced mathematical and matrix functions, type help specfun help elmat

Pi	3.14159265...
I	Imaginary unit, $\sqrt{-1}$
I	Same as i
Eps	Floating-point relative precision, 2^{-52}
Realmin	Smallest floating-point number, 2^{-1022}
Realmax	Largest floating-point number, $(2-\varepsilon)2^{1023}$
Inf	Infinity
NaN	Not-a-number

Table-6.1.4(b) Constants in MATLAB

Some of the functions, like sqrt and sin, are built-in. They are part of the MATLAB core so they are very efficient, but the computational details are not readily accessible. Other functions, like gamma and sinh, are implemented in M-files. You can see the code and even modify it if you want. Several special functions provide values of useful constants.

CHAPTER-7

RESULTS

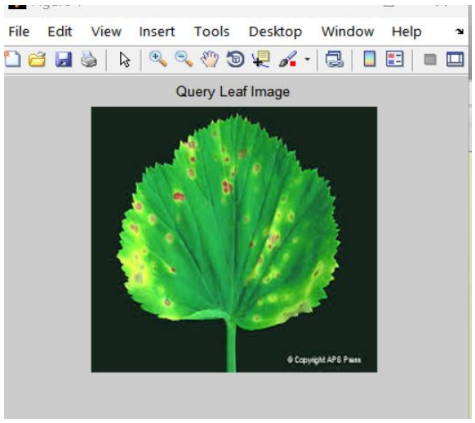


Fig.7(a) Input image

This is an input image used in a leaf disease detection system. It shows a green leaf with yellowish and reddish spots, which are likely symptoms of a plant disease. The image is processed using image processing techniques such as color analysis, feature extraction, and classification using machine learning models (e.g., CNN, SVM).

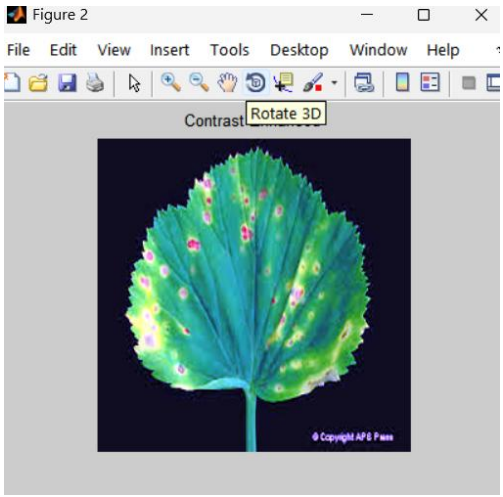


Fig. 7(b) Enhanced Image

This image represents a **contrast-enhanced version** of the input leaf image used for disease detection. The enhancement is likely performed to improve feature visibility, making disease symptoms more distinguishable for analysis.

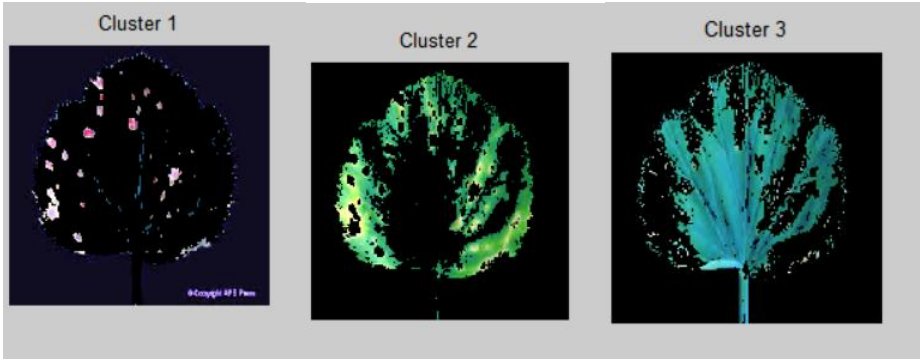


Fig.7(c) clustering Images

This image represents **clustering-based segmentation** of the leaf image, typically used for disease detection. The clusters help in identifying different regions of the leaf based on color, texture, or intensity variations.

Fig. 7(d) Leaf health level

This **warning dialog** displays the **chlorophyll content level** detected in the leaf, which is between **50-60%**.

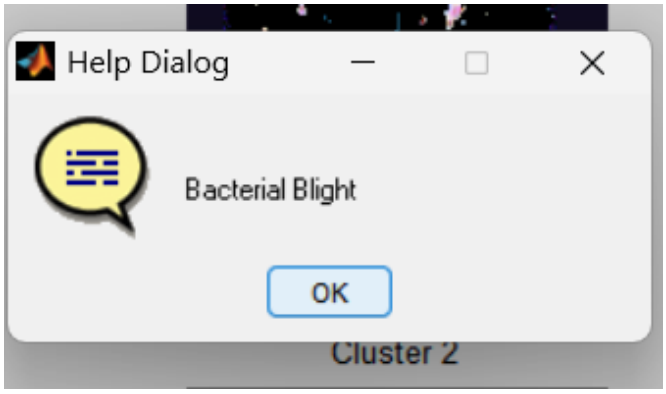


Fig.7(e). Disease name

This **Help Dialog** indicates that the detected disease in the leaf is **Bacterial Blight**.

```
ans =

Accuracy of leaf disease is: 98.3871%
```

Fig.7(f). Accuracy of leaf

This result shows that the **leaf disease detection model achieved an accuracy of 98.3871%**.

```
ans =

leaf problem Area is: 25.1669%
```

Fig.7(g). Leaf problem Area

The output "leaf problem Area is: 25.1669%" indicates the result of some kind of image analysis or processing, likely related to identifying and measuring the area of a "leaf problem" within an image.

CHAPTER-8

CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION:

Leaf disease detection using image processing plays a crucial role in modern agriculture by enabling early and accurate identification of plant infections. The traditional manual inspection methods are time-consuming, less accurate, and impractical for large-scale farming. The proposed approach leverages **K-Means clustering for segmentation** and **Support Vector Machine (SVM) for classification**, significantly improving disease detection efficiency and accuracy.

By applying **image preprocessing, segmentation, and feature extraction**, the system successfully identifies diseased areas and classifies them with high precision. K-Means clustering efficiently segments affected leaf regions, while SVM accurately classifies diseases based on extracted features. This automated method reduces human dependency, enhances productivity, and provides a cost-effective solution for farmers. Future improvements can include **deep learning models** and **real-time mobile applications** for even better accuracy and usability in agricultural settings.

8.2 FUTURE SCOPE:

The method reported in the thesis can be used to design a soya bean expert system for farmers for the early detection of plant foliar infection, infection grading and getting the appropriate cure remotely. Through the thesis work, we have tried to highlight the problems associated with the cultivation of soybean and causes of low yield loss in the developing countries like India. It has been taken-up six soya plant foliar diseases, namely; Rust, Bacterial Blight, Sudden Death Syndrome, Brown Spot, Downy Mildew, and Frog Eye, which are mainly responsible for significant yield loss; it has been proposed a fully automatic method for identification and classification by different digital image processing techniques and also to classify the disease severity level using five classes. It has been derived and development various new parameters and indices like DSI, IPR, DLP, which are subsequently used for disease level prediction.

REFERENCE

[1] Hong-ning Li, JieFeng, Wei-ping Yang, Xiang-sheng Wu, Ze-dong Li, Wei Liu "Spectrum-based Method for Quantitatively Detecting Diseases on Cucumber Leaf" 2011 IEEE.

[2] Hashim H.; Haron M.A.; Osman F.N; Al Junid, S.A.M "Classification of Rubber Tree Leaf Disease Using Spectrometer "2010 IEEE.

[3] Min Zhang,QinggangMeng "Citrus canker detection based on leaf images analysis"2010 IEEE.

[4] Dheeb Al Bashish, Malik Braik, and Sulieman Bani-Ahmad "A Framework for Detection and Classification of Plant Leaf and Stem Diseases" 2010 IEEE.

[5] Kholis Majid, YeniHerdiyeni, AunuRauf "I-PEDIA: Mobile Application for Paddy Disease Identification using Fuzzy Entropy and Probabilistic Neural Network" IEEE ICACSIS 2013.

[6] Mrs. Jayme Garcia ArnalBarbedo‖ Digital image processing techniques for detecting, quantifying and classifying plant diseases‖(SPRINGER PLUS-2013.

[7] SmitaNaikwadi, NiketAmoda, "advances in image processing for detection of plant diseases" International journal of application or innovation in engineering & management, PP: 168-175, November 2013.