Automated System for Detecting Jute Plant Disease Using Image Processing and Machine Learning Integrated with Mobile Application

Thesis Report

SCHOOL OF ENGINEERING AND COMPUTER SCIENCE
Department of Computer Science and Engineering
BRAC University

Supervisor
Dr Md. Haidar Ali

Conducted By

Faiza Nuzhat Joyee – 10101006

Nuzhat Ashraf Mahsa – 12101009

Zarreen Naowal Reza – 12101072

*All Authors have equal Contribution

Submitted on April 21, 2016

Declaration

This is to certify that this thesis report is submitted by Faiza Nuzhat Joyee (ID:10101006), Nuzhat Ashraf Mahsa (ID:12101009) and Zarreen Naowal Reza (ID:12101072) for the degree of Bachelor of Science in Computer Science and Engineering to the Department of Computer Science and Engineering, School of Engineering and Computer Science, BRAC University. We hereby declare that this thesis is based on the results found by ourselves and the materials of work found by other researchers are mentioned by reference. The contents of this thesis, neither in whole nor in part have been previously submitted to any other Institute or University for any degree.

_____
Signature of Supervisor
Dr MD Haider Ali

_____	_____	_____
Signature of Author	Signature of Author	Signature of Author
Faiza Nuzhat Joyee	Nuzhat Ashraf Mahsa	Zarreen Naowal Reza

Acknowledgement

It is a great honor for us to thank all for whom our thesis has been possible. We are very much grateful to my supervisor Dr. Md. Haidar Ali Professor and Chairperson, Department of Computer Science and Engineering, BRAC University for guiding us developing this system. It would have been really difficult to bring this work towards a completion without his guidance, enormous encouragement and continuous support. We also want to thank Dr. Md. Mahbubul Islam, Chief Scientific Officer and Dr. Saleh Mohammad Ashraful Haque, Senior Scientific Officer, Bangladesh Jute Research Institute (BJRI), for their assistance regarding validation of the approach for the developed system, for providing us with necessary data and suggestions for improving this system furthermore in future.

Abstract

In our research, we have implemented an automated system for disease detection of jute plants using image analysis and machine learning. We have also integrated this system with a mobile application for Android phones to serve the farmers where they get benefitted by identifying the diseases correctly and taking measures accordingly. At first, an image of the defected jute plant has been taken with the camera of the mobile phone. The farmers then just have to send the image to our server by selecting the options provided on the application. The image has been analyzed in the server where the plant's affected parts will be segmented using hue-based segmentation method, features for the texture analysis has been extracted using color co-occurrence methodology and comparing with our pre-defined database the disease is identified and classified using SVM classifier. At the final step, the classification result along with the necessary control measurement has been sent back to the user through the application on the phone.

Keywords: hue based segmentation, texture analysis, SVM classifier, mobile application for detecting Jute plant disease

Table of Contents

Chapter …..…….……..Page

List of Figures

Chapter One

Introduction

"Jute", the 'Golden Fibre' is considered as an important cash crop of many Asian countries, including Bangladesh. Like many other crops, jute plants get affected by various diseases every year. They cause a huge damage to the crop for which the farmers face a huge loss. For example, 60% to 70% jute plant may die in the field due to stem rot if no control measure is taken. For that reason we came up with the idea of helping the farmers to detect the diseases at the initial stage and we will also attempt to provide them with necessary control measures. We are attempting to develop a system that will detect the diseases of jute plants through image processing where Farmers will take snapshots of the leaves/stems of their disease affected crops and send them to the system server where the image will go through several levels of processing to detect and identify the disease. The result along with suggestions to improve the crop condition will be sent back to the farmer.

1.1 Motivation

Jute has always been a crop with major significance in the field of economy of Bangladesh. But in recent age for several reasons, the productions of this golden fiber have been hurdled. Fortunately, our government has stepped forward to revive the good times of this promising crop. On the other hand many researchers have been conducted for plant disease detection using image processing techniques. However, no such work has been done for jute plants' disease detection. If the diseases are not detected early and correctly, then the farmers have to incur huge losses. It is not always possible to correctly identify the diseases as it requires a lot of experience and knowledge. And at this recent

time as the production and exportation of jute is low the skilled workers, management experts are retiring from the jute sector. These are the cases from where we got the incentives to conduct our research on jute plant diseases and contribute a little by helping the farmers to detect the diseases at the initial stage and provide them with necessary control measures.

1.2 Thesis outline

Chapter-2 Contains the background study for development of the current system, which includes the detailed aspects of the targeted jute stem diseases followed by the literature review for related works in this field.

Chapter-3 presents the system design of the developed system. It includes various figures and diagrams that were used for designing and developing our system.

Chapter-4 describes the implementation details of the mobile application, the server side operation, including the analysis techniques used for image processing based disease recognition approach in every step.

Chapter-5 reviews the results of the research and provides a discussion on the project findings with comparisons.

Chapter-6 describes the overview and validation of our database.

Chapter-7 specifies the limitations and challenges we had faced during our work phase.

Chapter-8 describes our future development and updated system structure and our plan to launch it as real life application for rural mass usage. This part also includes conclusion of our project.

Chapter Two

Background Analysis

Bangladeshi jute is recognized as the golden fibre because of the excellent quality of it. For centuries, Bangladeshi Jute dominates the world market for its higher quality fibers. JUTE was the single most important export item of Bangladesh till the end of 1980s. Jute is one of the largest export oriented industries in Bangladesh and its contribution to the national economy is significant. According to the Bangladesh Jute Mills Corporation, major destinations of Bangladesh's jute goods export are Middle East, African Countries, European countries, South East Asia, Australia and USA.

Though it is an important Cash crop for increasing the economy of Bangladesh for some reasons the production of it has been hurdled. For which the exportation performance is also not in the static position where it should be. The falling price of raw jute is hindering our economy. In the below bar diagram [8], sourced from BJA, we can see how the export performance has decreased in recent times.

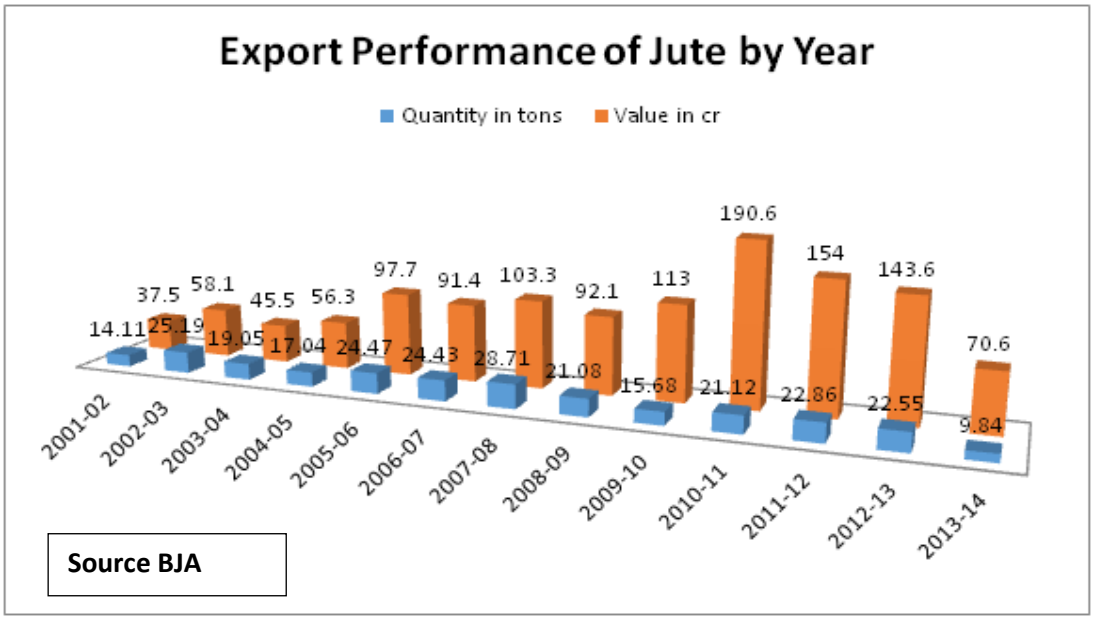

Fig: 2: Export Performance of jute

Jute is a unifying factor for workers and farmers both nationally and globally. But less cultivation, declining industries, less concentration from the government, wage dissatisfaction, etc have led the workers, experts and farmers to abandon this industry of ubiquitous possibilities. These are the words well known to everyone but it is time to think over what are the reasons behind this havoc. One of the very reasons that we have identified is the difficulties in identifying and curing diseases faced by Jute plant at different phases of production. This reason is way too difficult to solve as it requires experts of this field to identify any disease. Unidentified diseases lead to less production, less production leads to industry turmoil and lastly due to industry turmoil, the farmers, experts and workers cannot help abandon this industry.

2.1 Diseases

For this research, initially we have emphasized mainly on the seed-borne diseases of jute. Among the seed-borne diseases the most common diseases are caused due to fungal pathogens. With the collaboration of BJRI we have succeeded to enlist five diseases by which jute plants are being attacked and hampered the most. The diseases and their symptoms are described below: [Source: BJRI]

2.1.1 Stem rot (Macrophomina Phaseolina)

Stem rot is the most common and hazardous diseases of jute among all the seed-borne diseases. This disease alone damages around 5 lacs bales of jute fibers every year (Ahmad 1968). This disease is caused by *Macrophomina phaseolina* which is a kind of fungus.

Symptoms

Usually the symptom starts at the leaf region and gradually proceeds towards the petiole of the leaves. Leaves turn pale gray color in the mid rib and usually

turn black. The spot in the plant gradually grows in length by encircling the stem which internally rots and breaks the plant. As a result, the plants die. This disease initiates at the seedling stage when the height of the plant is around 6 to 8 inches and it stays till adult stage of jute plant. Brownish spots are noticeable on the leaves. These spots maybe seen from the lower level to the apex of the plant. Black dots are present at the brownish pretentious place. Eventually the precious consign break down resulting in the death of plant. Stem rot is disseminated by seed, soil and air. Warm and moist weather, excess nitrogen and low potash fertilizer in the soil are the favorable environment for this disease. Deshi and Tossa jute are mainly infected by this disease.

Fig: 2.1.1: Stem Rot

2.1.2 Die Back (Glomerella Cingulate)

Die back is another common disease that reduces the quality of jute fibre in an alarming manner. Basically the olitorius jute and kenaf plants are seen to be affected the most by die back disease. This disease is caused by *Glomerella cingulate*.

Symptoms

Die back affected plants begin to dry from the tip downwards at almost full-grown stage. This fungus is seed and air borne. The affected part becomes brownish and started drying from the tip and moves downwards at almost full growth stage. Inoculum enters through the wounds. The infected plants can hardly bear fruits. The whole plant dies off ultimately. This disease is disseminated by seed, soil and air. Warm and moist weather rises the contamination of this disease very quickly. Tossa jute and Kenaf are infected by this disease most frequently.

Fig: 2.1.2: Dieback

2.1.3 Black Band (Botryodiplodia Theobromae)

Black band is reportedly the most predominant seed-borne pathogen of jute (Baxter 1960). This disease is caused by *Diplodia corchori (Botryodiplodia theobromae).* This disease is also seed, soil, and air borne.

Symptoms

Initially, black spots are developed on the stem of the plant. The lesion first appears as small blackish brown patch, which is gradually enlarged and encircled the stem making a black band around. Then the number of black spots is increased with the increase of the time. The infected portion becomes slightly depressed while forming a band. The symptoms of this disease are usually

developed on the stem at 3-5 feet high above the ground level. The band that is being formed on the stem are larger in size and lesser in number than that of *Macrophomina phaseolina*. If one rubs his finger on the infected region of the stems, fingers will be black due to dark spores of the band. This disease is also disseminated by seed, soil and air. Deshi and Tossa jute are prone to be affected by this disease.

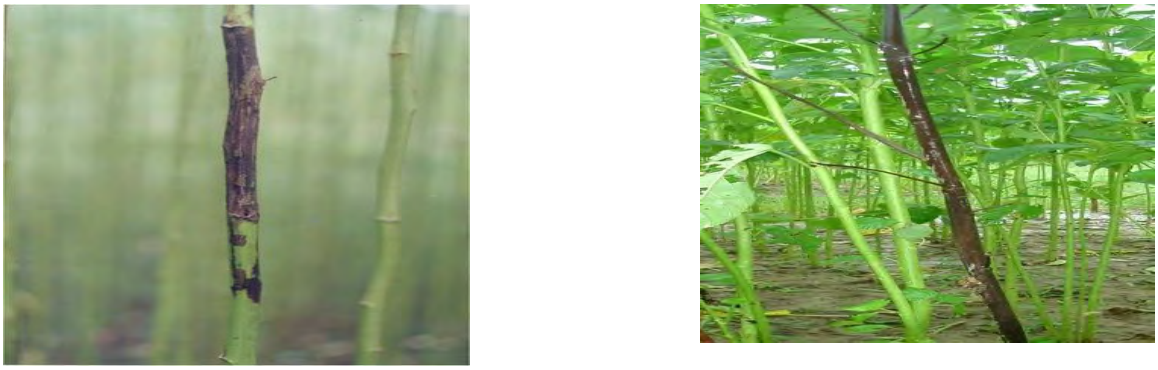

Fig: 2.1.3: Black band

2.1.4 Anthracnose (Colletotrichum Corchori)

Anthracnose is another very important disease of jute plants. *Colletotrichum corchori* is reportedly the causal organism of anthracnose [Ikata & Yoshida, (1940)]. Ahmed, (1966) also reported that anthracnose is a seed borne disease which causes seedling blight and pre emergence death of seedlings and consequent gaps in the field. In mature stages, the plant does not die but the disease very badly affects the fibre quality. In fact, the market value of the fibre is 30 to 50% reduced than that of healthy plants [Khan and Strange (1975)] due to get affected by Anthracnose. It mainly causes germination failure and rot of the seedlings leading the mature plants getting knotty in nature with adherent berks which resists retting and finally the production of low quality fibres resulting low income of the jute farmers. This disease is seen as more common in White jute plants (*Corchorus Capsularis*) than Tossa jute(*Corchorus Olitorius*).

Symptoms

In this disease, slightly depressed or sunken lesions are developed on the stem of the capsularis jute. The lesions are irregular in shape and brown to blackish in color. Several lesions coalesced with each other grows as a cankerous lesion on the stem. After a few days, the cankerous tissues are ruptured. In some cases, the stick may be visible through such lesions. Getting infected by this disease, plants usually do not die but it badly affects the fibre quality.

Fig: 2.1.4: Anthracnose

2.1.5 Soft Rot (Sclerotium Rolfsii)

Soft rot is caused by *Sclerotium rolfsii*. This disease can cause a huge decrease in yield of jute and many other crops like potatoes, tomatoes, corn, banana etc.

Symptoms

The disease appears first near the ground level of the plant. This disease spreads rapidly after rainfall as white cottony mycelia growth occurs at the collar region of jute crop. Sclerotia, resembling brown mustard seeds are formed on the stem. Foot of the plant is rotten due to the attack of this disease and the whole plant breaks down.

Fig: 2.1.5: Soft rot

2.2 Literature Review

The application of image analysis on various fields has become significantly important since past few years. At the very beginning the implementation of image analysis was limited to mainly medical and terrestrial images. Gradually, people started to apply this on agricultural purpose such as plant recognition, disease identification and management and so on.

In [1] Sariputra and Shirolkar has discussed about the basic steps to be taken to classify a disease by processing the image of the affected leaf. They have described the method that includes image acquisition, image pre-processing, segmentation of useful components, feature extraction and statistical analysis by using a spatial gray-level dependence method.

On the other hand, Lai and Leow in [2] has emphasized on different ways of texture analysis namely Statistical measures, Wold features and Gabor features. According to [2], Statistical measures are obtained by local statistical distribution of image intensity that measures coarseness, contrast and directionality as texture features. Similarly, the Wold model extracts the periodicity, randomness and directionality of a texture [2]. Periodicity and directionality are related respectively to the spatial frequency and orientation of

the texture, and randomness measures how uniform is the texture [2]. Beside these two, they conducted their research with Gabor features where the features are obtained by convolving an image with a set of Gabor filters.

In [3], the authors have discussed about leaf disease detection using image processing and neural network where they used Color Co-occurrence Methodology for extracting the features for texture analysis. For disease classification purpose, they have used multi-class SVM (Support Vector Machine) which is a binary classifier. In this classifier they used a winner-takes-all strategy, in which the classifier with the highest output function assigns the class [3].

Referring to the integration with mobile applications, Amos Gichamba and Ismail Ateya Lukandu in [3] described different implementations of mobile systems in agricultural purpose and presented a model for designing such applications. They also observed that the development of mobile solutions in the agriculture sector has not been yet done widely and showed how solutions can be created using mobile technology that will help in addressing some of the crop related problems faced by the farmers [20].

Another mobile-application based system has been presented in [5] by the authors named ‘Beetles’ which is designed to support farmers in rural area to detect crop diseases. ‘Beetle’ detects crop diseases from the image captured by a cell phone and detects the disease in real time using histogram and color information of the image [5].

Chapter Three

Methodology

3.1 System Design

We have built a system which can be operated by using a mobile application on Android phones. The farmers are the target user for this system. According to the information provided by BJRI, the jute crops get affected by several diseases within June to November. During this period, farmers need to be more cautious about their crops and sometimes they need to take immediate initiatives to save their crops from a certain disease. As mentioned earlier, for this research we have focused only on the seed-borne diseases that are caused by fungi. If a farmer wants to assess a disease-affected stem and need to be assured if the plant is affected by a certain disease or not, then he just has to use our mobile application and take a picture of the disease-affected stem. Then he will be given the option to send this image to our dedicated system server.

In our server, the image will be gone through several steps of analysis. Firstly, the image will be pre-processed to get prepared for further analysis. The process of pre-processing includes image resize, image enhancement, noise removal and filtering. After that, the image will be converted to a different color transformation, in our case we have converted our image to HSV (hue-saturation-value) color model as we are interested to conduct a hue-based segmentation method to extract the disease-affected portion of the plant.

In the next step, our aim is to extract the three individual channels from the HSV image and separate them as hue, saturation and value or intensity image. It should be mentioned that we want to use only the hue and saturation images for

the segmentation part as the intensity does not carry any color information in case of image segmentation based on color.

After the extraction of hue and saturation image, we want to mask the image by using thresholding. For this purpose, we have used the method described by Gonzalez and Woods at [6]. By using the thresholding, we are able to separate the affected stem from the background of the image.

Afterwards, we have applied the morphological analysis on the masked image in order to getting a more solid blob which will be used for further analysis. Morphological analysis includes erosion and dilation which fills up the scattered particles of the blob and gives it a smoother look.

In this level, we would be noticed that we have already detected more than one blob some of which are not useful to our analysis process. Thus, we need to get rid of the unwanted blobs. In order to getting the most useful blob we have applied an algorithm for detecting the largest connected component from the masked image. This algorithm gives us the biggest blob among the all which is the most useful segment for conducting our analysis.

Now as we have extracted our desired blob from the image, we will convert it back to RGB color space. This is the most important part of our analysis which we can call as feature extraction.

In our research, we have performed texture analysis for the classification of diseases. Texture analysis derives a general, efficient and compact quantitative description of textures so that various mathematical operations can be used to alter, compare and transform textures. To extract the features for texture analysis, at first we have converted the segmented RGB image to Grayscale image and then used the Color Co-occurrence methodology to calculate the feature values of the texture of the segment of the image. For this research, we

have calculated thirteen features in total including energy, contrast, homogeneity etc.

At final step, we passed the feature values to a binary classifier to compare the values with the values stored in our database and classify the particular disease. We have used the Support Vector Machine (SVM) for classifying multiple classes of the diseases.

After completing all the analysis, the result along with the respective control measurements are sent back to the Android application to show it to the user.

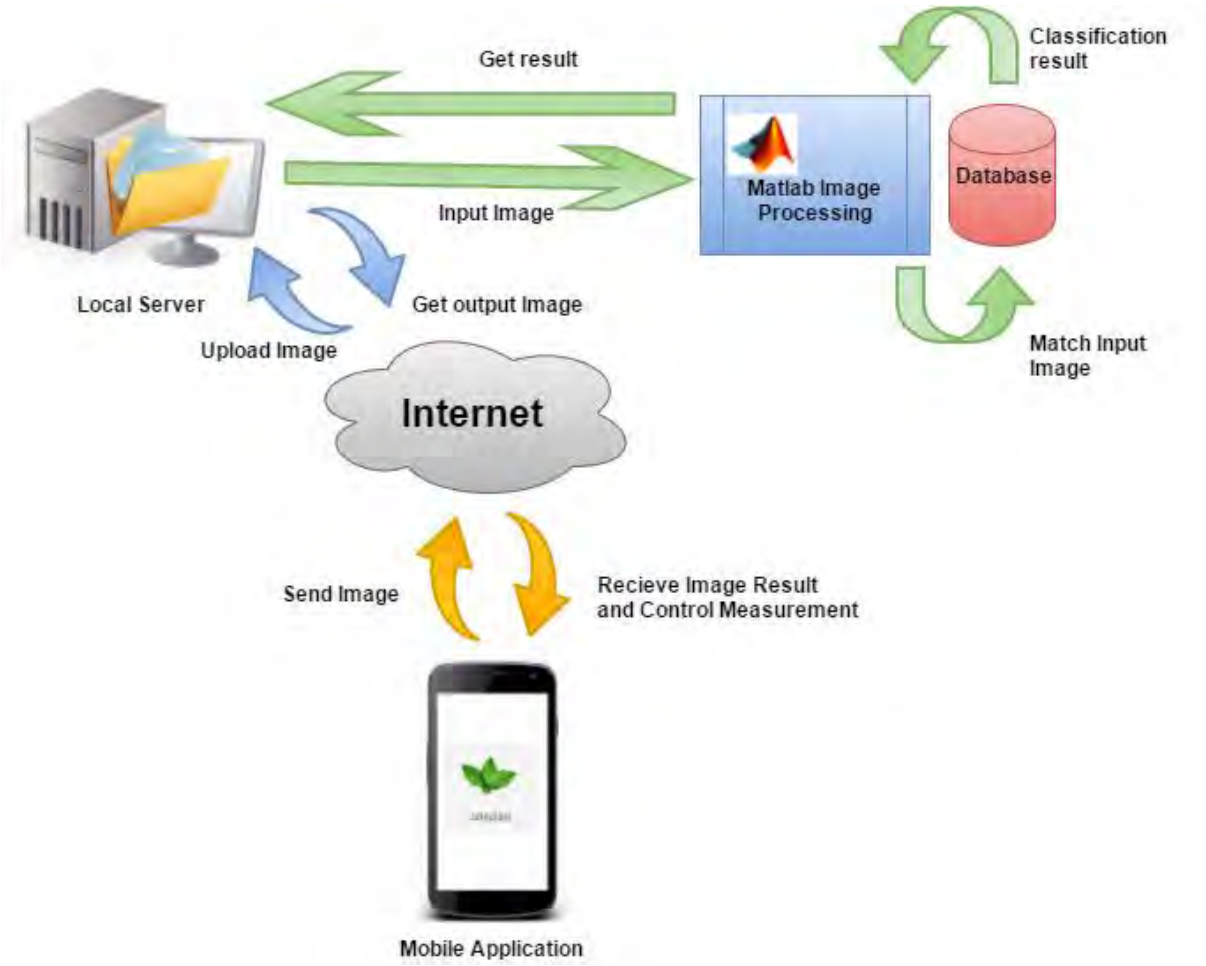

Fig: 3.1: System Overview

3.2 Diagrams

3.2.1 Flowcharts

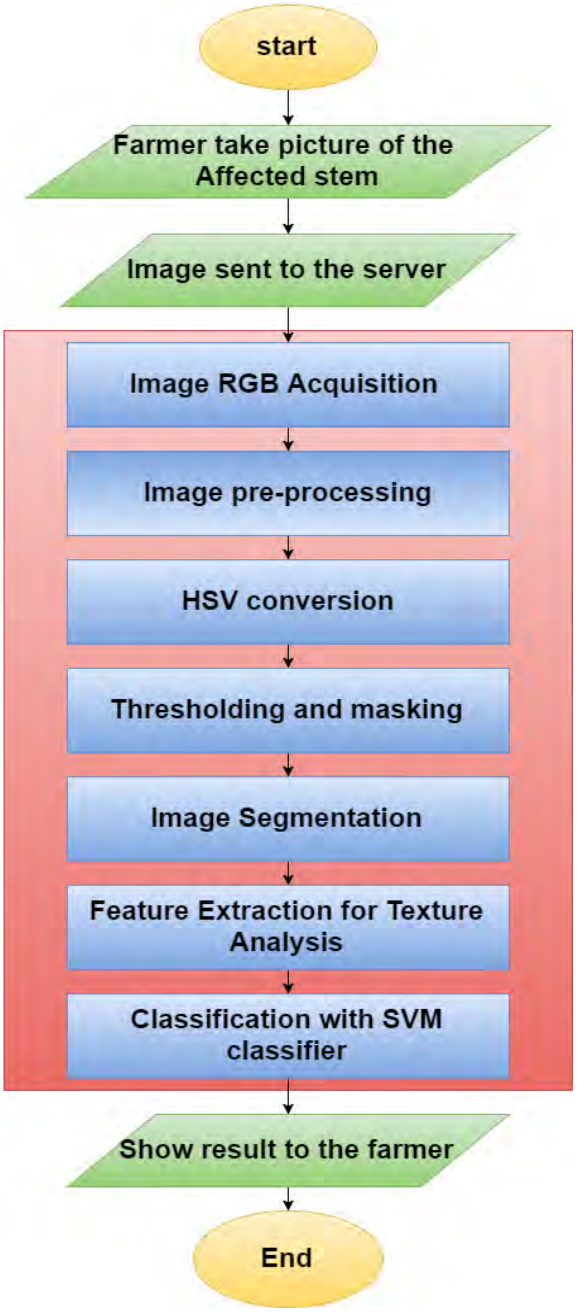

Fig: 3.2.1.1: Flowchart of overall system

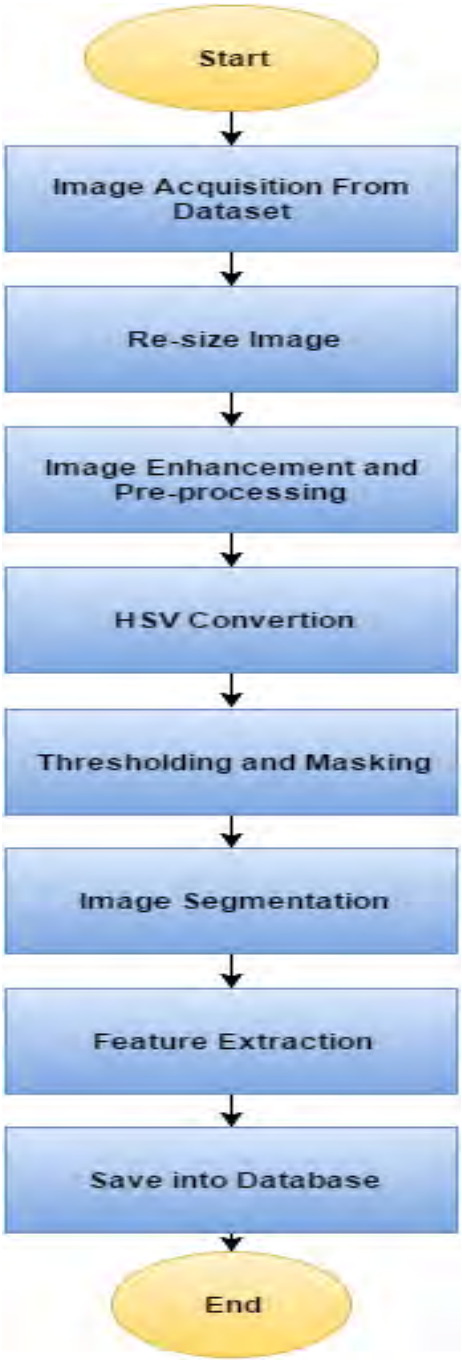

Fig: 3.2.1.2: Flowchart of database training

3.2.2 Activity Diagram

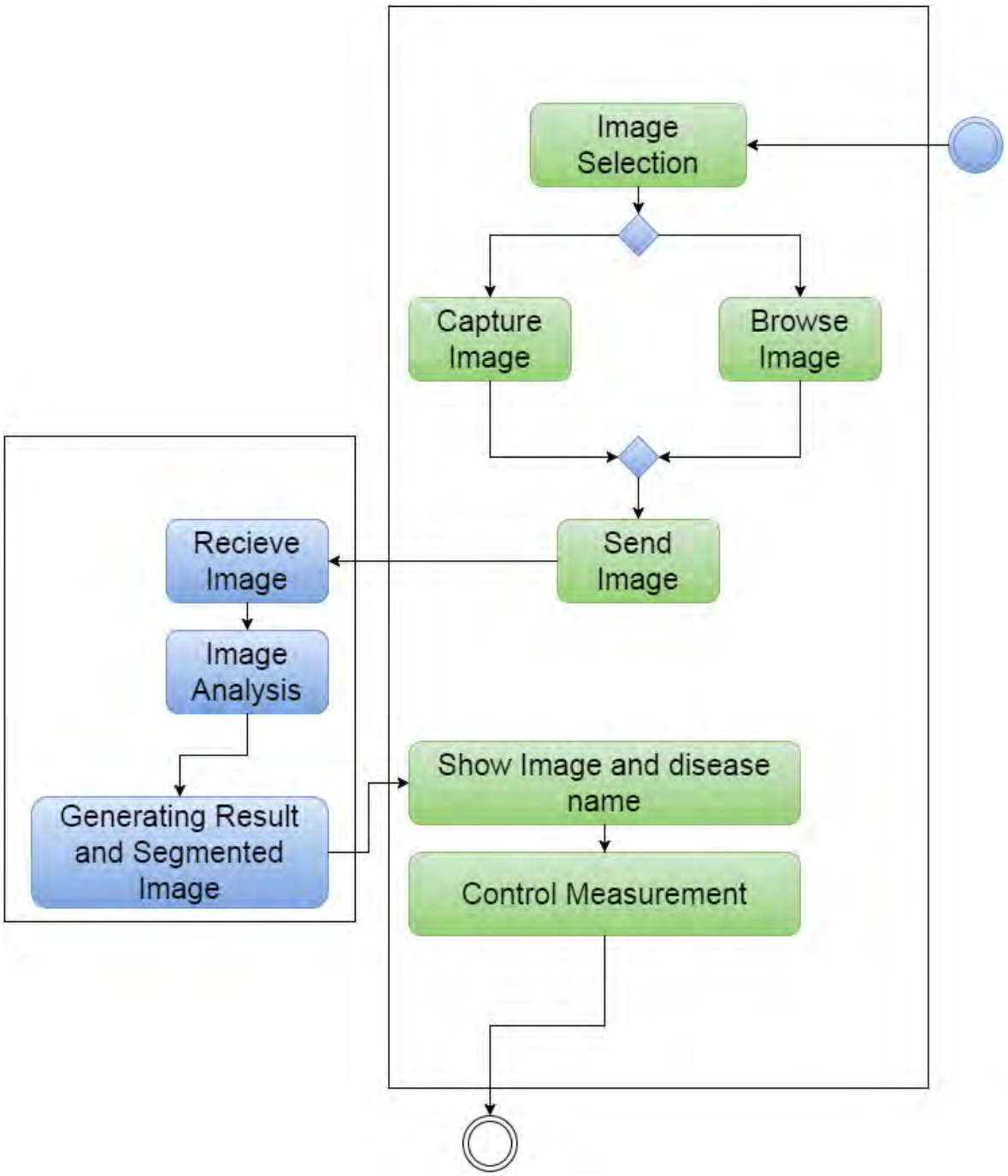

Fig: 3.2.2.1: Activity diagram

3.2.3 Use Case Diagram

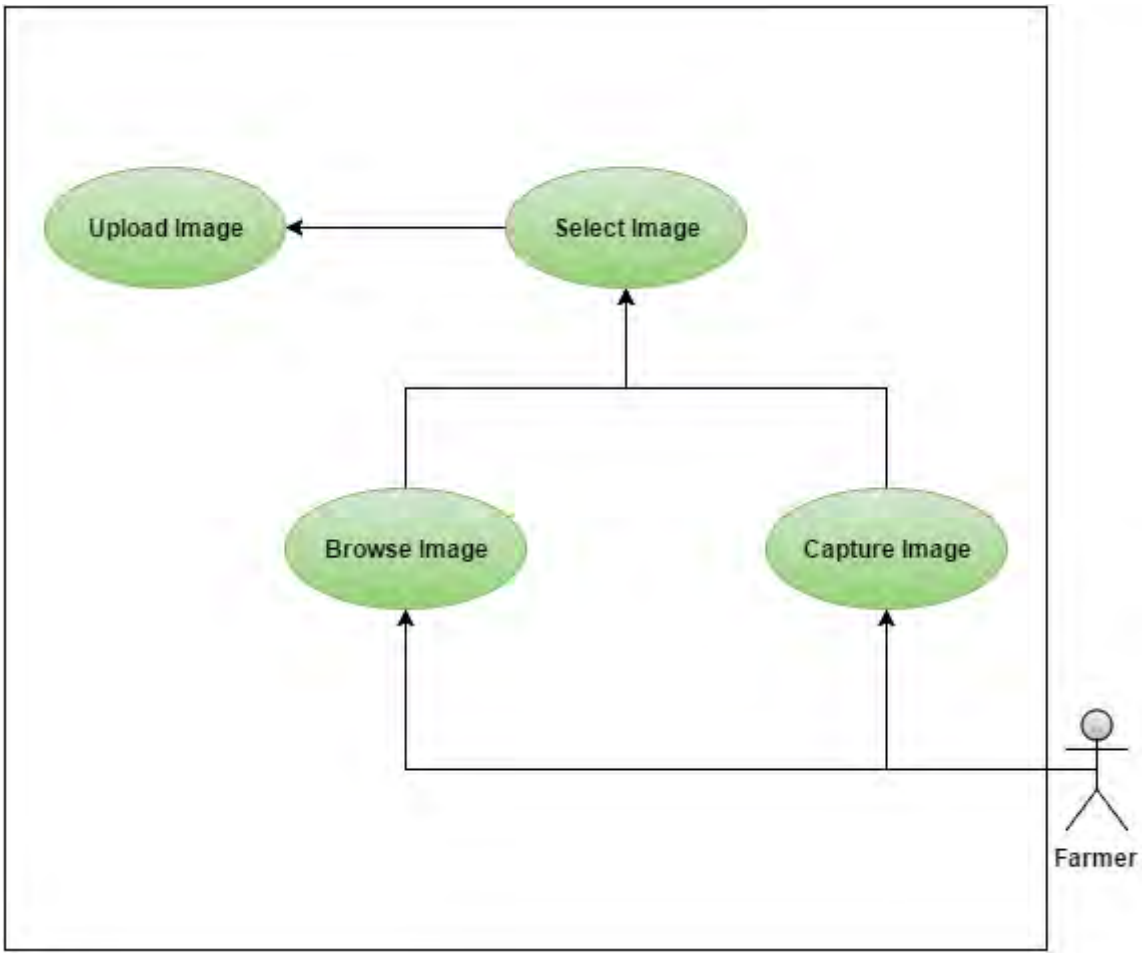

Fig: 3.2.3.1: Use Case diagram for uploading Image

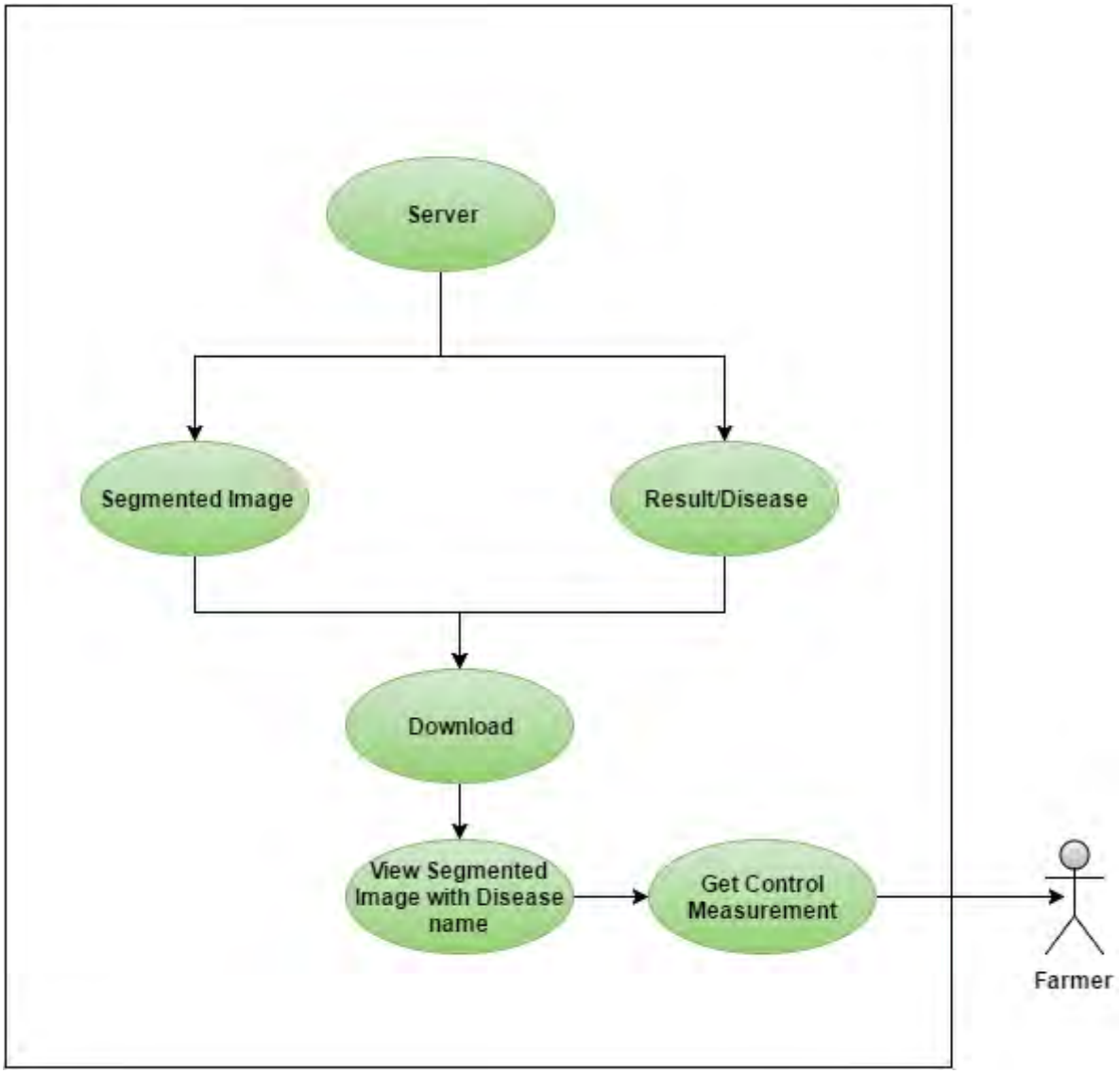

Fig: 3.2.3.2: Use Case diagram for Downloading Image

Chapter four

System Implementation

4.1 Mobile application development

The mobile application for our system “Jute plant disease detection” is targeted to be used by farmers or the people who are involved with the production process of jute and it will benefit them in a way that gives them solution within a short period of time with even an low price smart phone (android) resource without having to wait for the experts. The client mobile application has been developed for android phones with java programming language. Android studio with Android SDK tools (maximum API level 23 Support) was used in creating and implementing the application functionalities. This application supports all versions of androids including Android 4.4 "KitKat" is the single most widely used Android version in mobile markets.

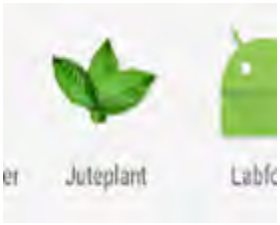

Fig: 4.1.1: Application Launcher Icon

Our mobile application has four major functionalities. They are 1) image capture/selection, 2) upload image to the server, 3) download the result from the server in form of segmented image with the disease name, 4) show the respective solution for the disease with the name of recommended fungicide.

1) Image capture/ selection

At the very first page after clicking the icon, the application shows a basic layout content_main.xml having one image view feature and two buttons. One button is for uploading the image to our local server and the second one is for downloading the result from the server.

Fig: 4.1.2: Interface of the Application

The MainActivity.java class starts when the app is started, OnCreate() method sets the layout view to content_main.xml. This class sends an intent ACTION_PICK to get the image from external storage of the phone. Then the bitmap image is set on the image view.

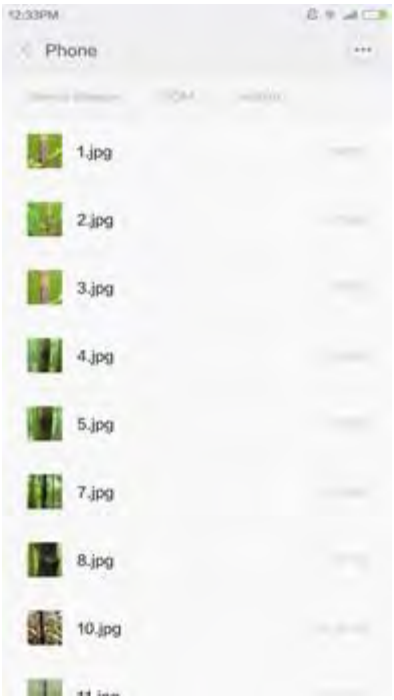

Fig 4.1.3: Choosing the picture from gallery

Fig 4.1.4: Selecting the image and showing it in image view

2) Upload image to the server

The upload image button for uploading the image is pressed in order to upload the image to server. Image uploading is done with the help of AsynchTask class. Asynchronous task class is a special class which allows the application to perform background work and publish the result to UI threads without interrupting the UI thread or handlers. This class is designed to be a helper class around a thread or handler which should ideally be used for short operations (a few seconds at most). Asynchronous programming helps the app stay responsive when it does work in the background. The downside of using synchronous programming on the UI thread to upload content to server is, the app will be blocked until the method returns. The app will not respond to user interaction which might turn the user frustrated. Thus the asynchronous class is the best choice for this kind of work so that the user interface will remain responsive while the operation is done in the background. After completion of the task the application will show a toast that image has been uploaded to server and if there is a problem it will show a error message in the toast. Our asynchronous task mainly overrides three methods that is preExecute(),postExecute() and doInBackground(). Firstly the bitmap image is converted to string using the built-in method Base64.encodeToString() method. It converts the bitmap image to Base 64 strings so that it could be sent to the server using HTTP web services. The bitmap image is firstly compressed into byte array output stream and then put into a byte array. Then this array is converted to base 64 string. The image in string form is put into a Hashmap data structure which will be sent as data using Http URL connection. The connection between the application and the server is maintained by the RequestHandler.java class. It establishes the connection using HttpUrlConnection class in java. Upload image class sends the Hashmap and server address to RequestHandler.java class.

The connection time out is set to 15 seconds and the request method type is POST. In the server side, the php checks if the request is POST type. If it is POST type, the data is then decoded from string to image and sent to Matlab for processing.

Fig 4.1.5: Uploading the image to server

3) Download the result from the server

Once the image processing is completed and the result is produced. The server sends back the result in form of string image and text string. After uploading is completed, the Download the result button is pressed and the layout is set to activity_view_image.xml and the ViewImage.java class starts executing. Here also we use asynchronous programming for the same reason as we need the task to operate in background. The input stream is decoded from string to bitmap format. The request method type is set as GET and the URL connection time out is set to 15 seconds. In the activity_view_image.xml layout contains two buttons, one image view and one text view attributes. The top button is for getting the segmented image which shows only the affected area of the stem and

the bottom button is for navigating the user to solution needed for the disease. The image view shows the segmented image and the text view updates with the name of the disease the plant is diagnosed with. Downloading and uploading image both activities are done with the help of asynchronous programming and the RequestHandler.java class. The connection between the app and the server is established by sendGetRequest() method in RequestHandler.java class and the input stream in decoded inside the ViewImage.java class inside asynchronous programming.

Fig 4.1.6: Showing the Segmented Image and disease name

4) Getting solution

The bottom button in activity_view_image.xml layout is pressed after downloading the segmented image and the analyzed disease name from the server in order to know about the necessary steps one can take to prevent the disease from spreading. After clicking the button, the user is directed to solution.xml layout where he or she can see the control measurement recommended by the specialist in Bangladesh Jute Research Institute (BJRI). The instructions are simple enough to understand and the occurrence period of

the disease is also mentioned in the solution. This activity is displayed by the Solution.java class which functions relatively in simple way than other classes of this application. It just receives the intent from ViewImage.java class with the disease name as string and search for the solution to show for the received disease name.

This application is designed in such way that the client does not require much technical knowledge to operate it. The instructions like the button names are pretty straight forward and will lead the user to the desired solution within short time. Our automated system is mostly combined of press and send techniques. The client sends the affected jute plant image to server by pressing the button and receives the solution within seconds with suggestions of controlling the spread of the disease. With time, there will more features in the app so that the rural people who have comparatively less knowledge and experience with technology will be able to use this mobile application and get help in order to cultivate good quality jute.

Fig: 4.1.7: Show Control Measurement According to the disease

4.1.1 Client and server interaction

This part of our system will be executed in the server as soon as the client sends the image via mobile application. When the mobile application sends data to the server, it is received using a PHP script using methods and the fetched images are stored in the assigned directory and our image processing will start on the assigned image automatically.

4.1.2 Shell Execution

For the purpose of our project, we created a local server in our laptop using "XAMPP" software. The server has PHP scripts that will automatically receive images and send to a particular directory and send the image from an assigned location in our laptop after segmenting the image. Previously, we created an 'exe' file of our MATLAB code and saved it to our local server. From our PHP script with an function called exec(), we can run this 'exe' file of MATLAB through shell execution. The exec() method in the PHP is called just after receiving and saving the image in the server, so the whole process is done automatically without any human interaction on our server side. It reduces the possibility of errors in a way that we can depend on machines here completely. The MATLAB 'exe' file contains every step of our image processing algorithms.

4.1.3 Parsing the result

Once the MATLAB code is executed through shell execution and the result is saved in the pre-assigned directory in our server, the PHP script fetches the data for the mobile application for request method type GET and sends it to our client. The result consists of two types of data. One is the segmented image which highlights the affected portion of the plant in Base 64 string form and the

other part of the result is the disease name the MATLAB code generates after analyzing the image in string form. Both of them are fetched by PHP script and sent over the internet to our mobile application.

4.2 Server Side Operations

The image received by the Android application is saved in the system server and the entire analysis procedure begins there. The algorithm that is implemented for the image analysis has been organized in several steps.

4.2.1 Image Preprocessing

Before proceeding towards image analysis the image should be processed in order to acquiring better result. Images taken from camera phones contains different factors which alters the result of the analysis. Our image preprocessing procedure is performed by following certain steps which are image resizing, image enhancement and noise removal.

4.2.1.1 Resize Image

To perform the classification, process the size of the input image must match the size of the images stored in the database. Thus, the input image must be resized to a fixed dimension at the very first stage.

4.2.1.2 Enhance the image

In this step the image intensity values or colormap has been adjusted so that 1% of the data is saturated at low and high intensities.

```
imadjust(I,stretchlim(I));
```

The above function adjusts the contrast of the image that returns a two-element vector of pixel values that specify lower and upper limits that can be used for

contrast stretching image I. By default, values in Low_High for this function specifies the bottom 1% and the top 1% of all pixel values of the image.

Original Image

Enhanced Image

Fig: 4.2.1.2.1: Enhancing the Image

4.2.1.3 Noise Removal

The uploaded images may contain noise. Noise can turn a simple thresholding problem into an unsolvable one. Thus it is very important to remove the noises from the image. In this case, a bilateral smoothing filter has been used for noise cancellation. The bilateral filter is a technique to smooth images while preserving edges where the intensity value at each pixel in an image is replaced by a weighted average of intensity values from nearby pixels [7].

4.2.2 Hue-Based Segmentation

In this paper, we have used the hue-based segmentation method to segment only the affected portion from the image. As we are conducting the entire process on stem diseases it is not possible to simply mask the green pixels from the image likewise the ones done in case of detecting leaf diseases and complete the segmentation. As a result, we have chosen hue-based segmentation method.

4.2.2.1 HSV Conversion

At first, the RGB spaced image has been converted to HSV color space. The reason behind that is we want to segment the image basically based on color and we also need to carry out the process on individual planes for thresholding. In HSV space, the color of the image is conveniently represented in the hue image and saturation is used as a masking image in order to isolate further regions of interest in the hue image [6].

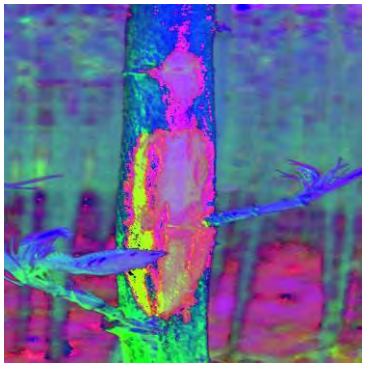

Fig: 4.2.2.1.1: HSV Converted Image

We have extracted the individual channels to separate the hue, saturation and intensity images.

Hue

Saturation

Value

Fig: 4.2.2.1.2: Hue, Saturation And Intensity Image

4.2.2.2 Thresholding

Thresholding holds a central position in the application of image segmentations [6]. Image thresholding is a simple, yet effective, way of partitioning an image into a foreground and background. This image analysis technique isolates objects by converting color images into binary images. This method is most effective in images with high levels of contrast. As we have already adjusted the contrast of the input image through preprocessing, we can use image thresholding in it and get better results.

In the thresholding method described in [6], we generate a binary mask by thresholding the saturation image with a threshold equal to ten percent of the maximum value in that image. Any pixel value greater than the threshold is set to 1 (white) and all others are set to 0 (black).

After that we have multiplied the binary saturation mask to the hue image. This product gives us our desired objects from the image separating them from the background.

Fig: 4.2.2.2.1: Masked Image

4.2.2.3 Extraction of the largest connected component

As we can see above, the masked image contains many unwanted blobs which are of no use for our analysis process. Thus, in order to remove them, we have

used an algorithm for extracting the largest connected component. In the algorithm we have used **bwlabel** and **regionprops** function to define the blobs and measure their area.

Bwlabel takes in a binary image. This binary image may contain a bunch of objects that are separated from each other. Pixels that belong to an object are denoted with 1 or true while those pixels that are the background are 0 or false.

Regionprops measures a variety of image quantities and features in a black and white image. In order to measure the area of the blobs we have used the following function:

```
regionprops (BinaryImage, 'area');
```

This function calculates the area of the blobs from the labeled image by looking into the local neighborhood for the pixels that are 1 and connected by a chain. After calculating those pixels, we can have the biggest blob with the greatest number of pixels connected with 1s. In this case, we have used 8-pixel neighborhoods where we need to look at the North, Northeast, East, Southeast, South, Southwest, West, and Northwest directions.

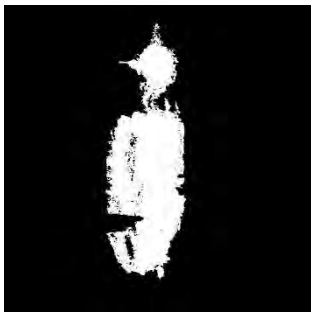

Fig: 4.2.2.3.1: Extraction of the Biggest Blob

4.2.2.4 Morphological Analysis

We need to apply some morphological analysis on the detected blob in order to getting a smoother and filled shape of it. For our analysis part, we have applied **dilation** and **erosion** on the blob.

Dilation block replaces each pixel with the local maximum of the neighborhood around the pixel. The block operates on a stream of binary intensity values. Theoretically, the dilation of *f* by a flat structuring element *b* at any location (x,y) is defined as the maximum value of the image in the window outlined by *b*^ when the origin of *b*^ is at (x,y) [6].

$$[f \oplus b](x,y) = \max_{(s,t) \in b} \{f(x-s, y-t)\} \quad \text{---------- (i)}$$

Similarly**, Erosion** of *f* by a flat structuring element b at any location (x,y) is defined as the minimum value of the image in the region coincident with b when the origin of b is at (x,y). In equation form, the erosion as (x,y) of an image f by a structuring element b is given by [6]

$$[f \ominus b](x,y) = \min_{(s,t) \in b} \{f(x+s, y+t)\} \quad \text{---------- (ii)}$$

Fig: 4.2.2.4.1: After Applying Morphological Analysis

4.2.2.5 RGB Conversion

For further analysis, process, we need to convert the segmented portion back to its original color. That's why we have converted the segmented image to RGB color space.

Fig: 4.2.2.5.1: Segmented RGB Image

4.2.3 Feature Extraction

As we are using texture analysis for the classification of diseases, we have to extract the features first for that. For that purpose, we have used the color co-

occurrence methodology which is developed through the GLCM (Grey-level Co-occurrence Matrices). The gray level co-occurrence methodology is a statistical way to describe shape by statistically sampling the way certain gray-levels occur in relation to other gray levels [3]. These matrices measure the probability that a pixel at one particular gray level will occur at a distinct distance and orientation from any pixel given that pixel has a second particular gray level. The GLCM's are represented by the function P(i, j, d, θ) where i represent the gray level of the location (x, y), and j represents the gray level of the pixel at a distance d from location (x, y) at an orientation angle of θ [3].

```
graycomatrix(segmented_image);
```

The function stated above is used to get the GLCM from the segmented grey image.

For this research, we have calculated thirteen feature values for each of the input images for performing texture analysis.

1. Contrast
2. Correlation
3. Energy
4. Homogeneity
5. Mean
6. Standard Deviation
7. Entropy
8. RMS (root mean square) contrast
9. Variance
10. Smoothness
11. Kurtosis
12. Skew ness
13. IDM (Image difference-measure)

The above features are calculated from the GLCM using their corresponding formulas.

4.2.4 Classification

After the extraction of all the necessary features, we have to compare them with our pre-calculated dataset stored in a .mat file. We have used the SVM (Support Vector Machine) classifier for classifying the disease. Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. Supervised learning involves analyzing a given set of labeled observations (the training set) so as to predict the labels of unlabeled future data (the test set). Specifically, the goal is to learn some function that describes the relationship between observations and their labels. More formally, a support vector machine constructs a hyper plane or set of hyper planes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyper plane that has the largest distance to the nearest training data point of any class (so-called functional margin), in general the larger the functional margin the lower the generalization error of the classifier [3].

Multiclass SVM aims to assign labels to instances by using support vector machines, where the labels are drawn from a finite set of several elements. The dominant approach for doing so is to reduce the single multiclass problem into multiple binary classification problems. Common methods for such reduction include: building binary classifiers which distinguish between (i) one of the labels and the rest (one-versus-all) or (ii) between every pair of classes (one-versus-one) [3].

In this case, the classification of new instances or input images for the one-versus-all case is done by a winner-takes-all strategy, in which the classifier

with the highest output function assigns the class and detects the particular disease.

4.2.5 Create Database and Train Classifier

In our experiments, two main files were generated, namely: (i) Training texture feature data, and (ii) Testing texture feature data. The two files had 150 rows each, representing 30 samples from each of the five classes of leaves. Each row had 13 columns representing the 13 texture features extracted for a particular sample image. Each row had a unique number (1, 2, 3, 4, 5) which represented the class (i.e., the disease) of the particular row of data. "1" represents Anthracnose,

"2" represents Black Band, "3" represents Die Back, "4" represents Soft Rot and "5" represents Stem Rot. Then, a software program was written in MATLAB that would take in .mat files representing the training and testing data, train the classifier using the "train files", and then use them to perform the classification task on the test data.

Consequently, a Matlab routine would load all the data files (training and testing data files) and match them to generate the result which identifies the particular disease.

0.1846	0.9492	0.7638	0.9747	0.0703	0.2016	1.5495	0.1020	0.0208	1	10.509
0.1669	0.9324	0.7970	0.9786	0.0537	0.1713	1.2721	0.0854	0.0159	1	13.797
0.2194	0.9442	0.7408	0.9717	0.0775	0.2095	1.6643	0.1080	0.0200	1	9.164
0.0103	0.7579	0.9696	0.9961	0.0048	0.0280	0.5215	0.0125	7.0000e-04	0.9997	96.767
0.0095	0.8332	0.9586	0.9958	0.0066	0.0320	0.7215	0.0189	9.0000e-04	0.9998	62.909
0.0180	0.7207	0.9924	0.9986	0.0026	0.0270	0.3989	0.0070	6.0000e-04	0.9995	888.038
0.0402	0.9218	0.8831	0.9919	0.0265	0.0778	2.0905	0.0630	0.0055	1	74.806
0.0729	0.9411	0.8645	0.9878	0.0325	0.1222	1.2934	0.0832	0.0120	1	22.912
0.0211	0.8502	0.9609	0.9951	0.0059	0.0431	0.3659	0.0117	0.0013	0.9998	107.365
0.0422	0.8689	0.9186	0.9905	0.0134	0.0654	0.7470	0.0382	0.0038	0.9999	50.723
0.0977	0.8963	0.8244	0.9815	0.0311	0.1057	1.0858	0.0668	0.0092	1	16.071
0.1344	0.9567	0.8153	0.9763	0.0553	0.1825	1.2534	0.0706	0.0072	1	13.323
0.1575	0.9685	0.7375	0.9688	0.0861	0.2300	1.7109	0.1024	0.0076	1	8.525
0.0862	0.8686	0.9165	0.9882	0.0163	0.0857	0.5147	0.0380	0.0058	0.9999	34.318
0.1286	0.9458	0.7929	0.9799	0.0510	0.1599	1.3335	0.1496	0.0241	1	12.731
0.1783	0.9348	0.7303	0.9668	0.0629	0.1736	1.8789	0.1124	0.0192	1	12.694
0.1562	0.9611	0.6433	0.9746	0.0933	0.2069	2.2702	0.2171	0.0412	1	6.284
0.1837	0.9161	0.8729	0.9847	0.0367	0.1488	0.7819	0.0801	0.0165	1	18.997
0.1511	0.9462	0.8071	0.9809	0.0544	0.1710	1.1845	0.1119	0.0215	1	11.111
0.0144	0.7769	0.9816	0.9975	0.0025	0.0285	0.1542	0.0098	8.0000e-04	0.9995	295.662
0.0017	0.6569	0.9953	7.0000e-04	0.0101	0.1096	0.0021	1.0000e-04	0.9982	0.9982	466.432
0.5154	0.9188	0.4708	0.9357	0.1620	0.2620	3.1688	0.2678	0.0567	1	3.048
0.1241	0.9557	0.8020	0.9807	0.0537	0.1703	1.2988	0.1103	0.0171	1	12.018

Fig: 4.2.5.1: Snapshot of the Knowledgebase

Chapter Five

Results Analysis

5.1 Result and accuracy

For our research, we have examined with two different cases for extracting the features for texture analysis. In case 1, we have calculated the features from the training images without applying the hue-based segmentation method on them. On the other hand, in case 2 we have calculated the features after the segmentation method.

From the observation on the both cases, we found that case 1 provides us 60 percent of accuracy in detecting the diseases which is not very much satisfactory. On the contrary, case 2 provides an accuracy of around 87 percent which is far better than case 1. Comparing both the accuracy results, we have come decided to proceed with the methods applied in case 2.

The comparison diagram between two cases is shown below:

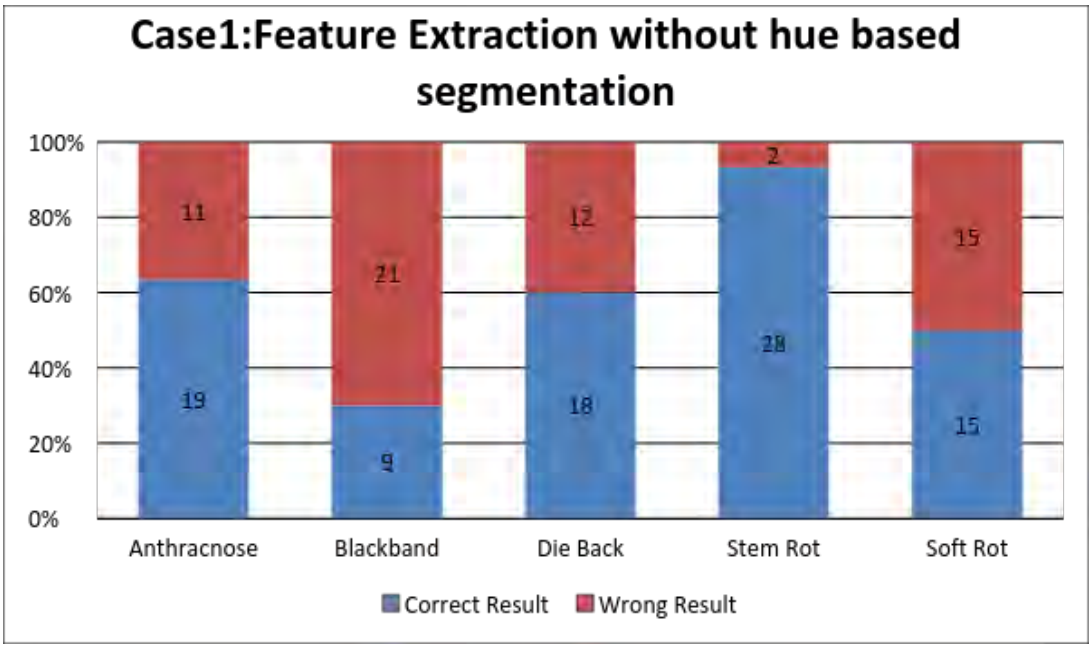

Fig: 5.1.1: Result for Case-1

In Case 1

Total Percentage of accuracy = (n x 100) / M

Where, n = no of correct Result

M = no of total Images for testing

Accuracy percentage = (91 x100) / 150 = 60% ---------- (iii)

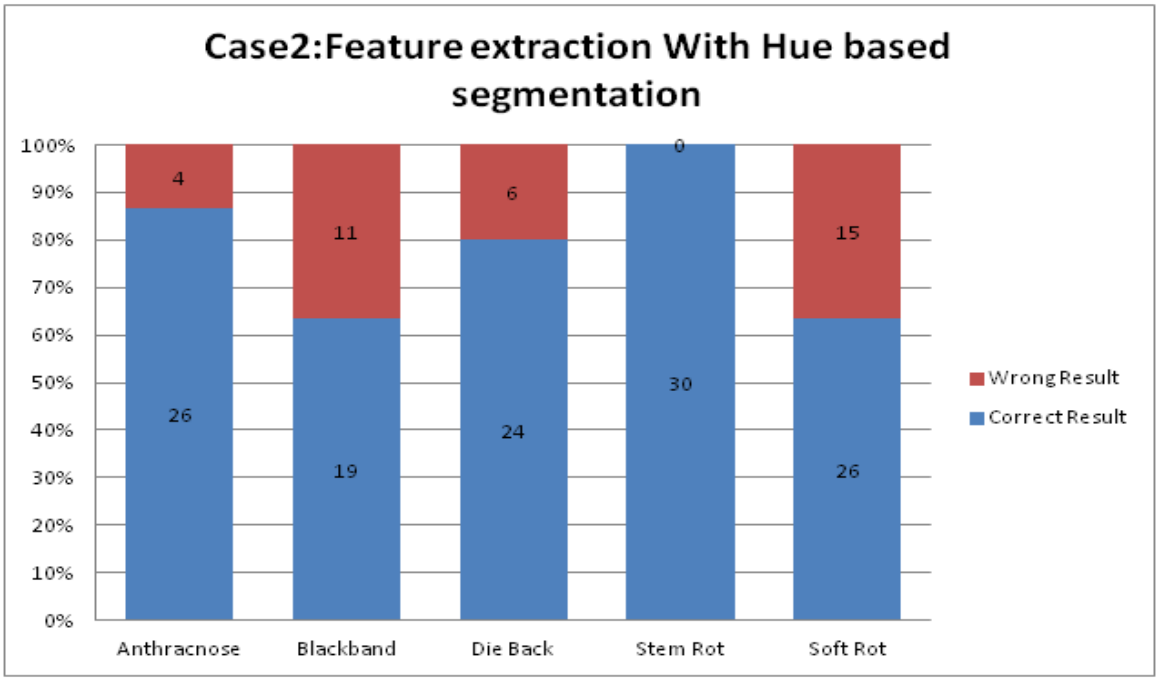

Fig: 5.1.2: Result for Case-2

In Case 2

Total Percentage of accuracy = (n x 100) / M

Where, n = no of correct Result

M = no of total Images for testing

Accuracy percentage = (129 x100) / 150 = 86% ---------- (iv)

Comparison between Case 1 and Case 2

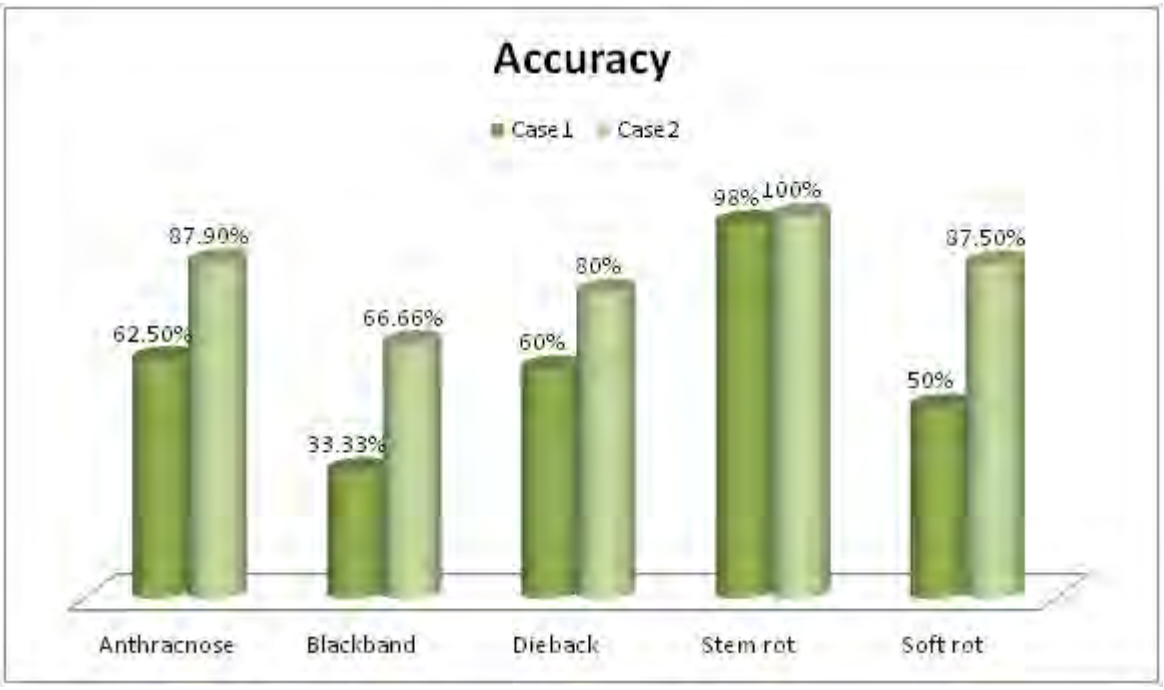

Fig: 5.1.3: Comparison between Two Cases

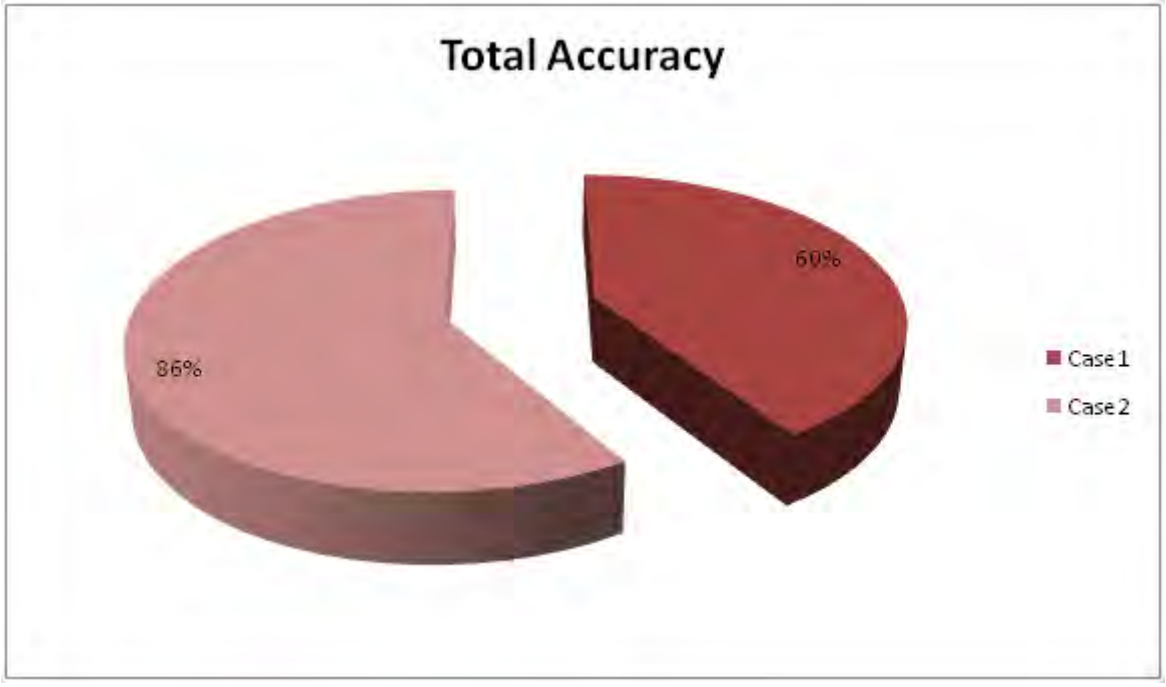

Fig: 5.1.4: Pie Chart for Total Accuracy

Chapter six

Database

The database of our system is not as elaborate as we wanted it to be but sufficiently satisfactory for the authenticity of our system. As we started our project from January month, we had almost three months for the full implementation of the system. But jute season in Bangladesh starts after April and for collecting the raw data from the field we would have to wait till June because around this time usually the jute plants are affected by diseases. After creating the image processing system with full mobile application integration, it was not possible for us to wait for another two months. So we decided to visit BJRI (Bangladesh Jute Research Institute) and collect raw data from their database of jute plants. We could manage sufficient amount of data and incorporate it into our matlab database. However, our algorithms worked well with the data set and we managed to achieve 67 percent of accuracy. So, if we add more data sets to our database our accuracy level will improve.

Crop Calendar of Bangladesh

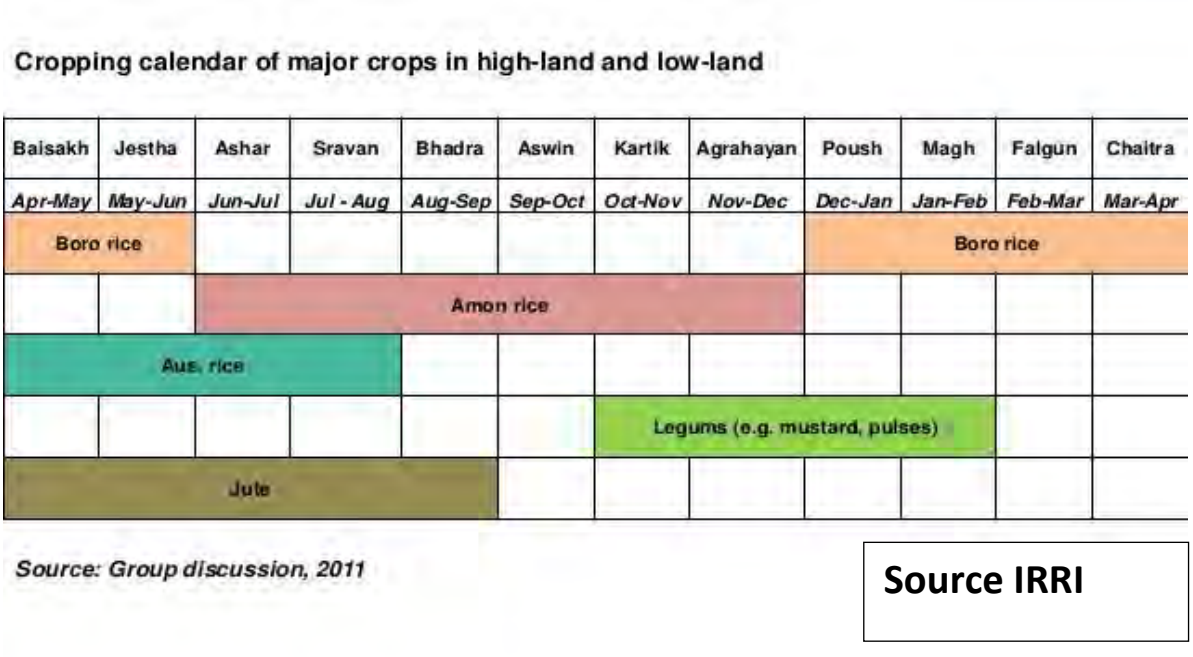

Fig: 6.1: Crop Calendar Of Bangladesh

Chapter seven

Limitations and Challenges

7.1 Limitations

We have considered some limitations to the system. One of them is regarding cancellation of the background for the image analysis of jute disease. If the image has a chaotic background and the actual affected jute plant is not prominent in the picture, it will be difficult to process the image. In this case, the analysis result might not show accuracy, if there is any unwanted background in the image. We have dealt with this situation by taking the image of the only disease affected area of the jute plant. Another point is, our mobile application has English language in the layout so it can be difficult for the rural farmers to understand it. So in order to actually launch our system for mass usage, it is essential that our mobile application has Bangla language layout or voice navigation in Bangla.

7.2 Challenges

Availability of data sets was one of the major challenges that we faced during our work period. As jute season starts in Bangladesh after April, we could not have as much as raw images as we wanted but we managed to get our hands on the resources of Bangladesh Jute Research Institute (BJRI) and the collected raw images helped us immensely with the variation of data which our project demanded. The Android development process has also faced some challenges such as our primary work plan was to develop a mobile application which will take raw image from mobile camera and process it. but jute cultivation was not available, so we did not include the camera activity instead we gave user the

options to select images from external storage of the phone that means user can select from previously captured images.

Chapter Eight

Conclusion and Future Work

8.1 Future Work

We have future plan to implement our system for real life application in the rural areas so that the people of our country get benefitted by our work and it will help them in jute cultivation immensely. We would like to introduce background cancellation technique in order to cancel out all the unwanted objects in the image. We want to develop Bangla language layout and Bangla voice navigation for our mobile application in near future. In addition, we also would like to upgrade our system a high level where it can detect all the possible jute diseases that our farmers face in the cultivation period.

8.2 Conclusion

Jute is a very important crop for the economy of Bangladesh. It has played a vital role in our economy in past years but not so much in the recent times as it lacks proper direction and patronization. Thankfully, recently awareness regarding jute cultivation is increasing and what other better way there is than introducing a service that can be used by anyone through their mobile phones. This simple system implementation will allow people with very little technological knowledge especially rural people to use and get effective help out of it. The barrier between the rural people and the technology will exist no more if we try to utilize the benefits of technology.

References

1. Sariputra, D., & Shirolkar, A. (n.d.). A Review Of Plant Leaf Disease Detection And Classification Based On Digital Image Processing Techniques. INTERNATIONAL JOURNAL OF TECHNOLOGY ENHANCEMENTS AND EMERGING ENGINEERING RESEARCH, VOL 4, ISSUE 2, ISSN 2347-4289.

2. Lai, S. Y., & Leow, W. K. (n.d.). Invariant texture matching for content-based image retrieval

3. S. Arivazhagan, R. Newlin Shebiah, S. Ananthi, S. Vishnu Varthini. 2013. Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. Agric Eng Int: CIGR Journal, 15(1): 211 - 217

4. Amos Gichamba, Ismail A. Lukandu "A Model for designing M-Agriculture Applications for Dairy Farming", The African Journal of Information Systems, Volume 4, Issue 4, 2012

5. Rahat Yasir, Nova Ahmed "Beetles: A Mobile Application to Detect Crop Disease for Farmers in Rural Area", Workshop on Human and Technology (WHAT), 8-10 March 2014, Khulna, Bangladesh

6. Gonzalez, R. C., & Woods, R. E. (2009). Digital image processing. UP, India: Dorling Kindersley.

7. Sylvain Paris, Pierre Kornprobst, Jack Tumblin, Fred Durand "A Gentle Introduction to Bilateral Filtering and its Applications".

8. JUTE and Bangladesh Jute Industry, Bangladeshi Jute Products Manufacturer and Exporter - Source of Best Jute Deals. (n.d.). Retrieved April 17, 2016, from http://www.bestjute.com/

9. Bangladesh Jute Spinners Association. (n.d.). Retrieved April 17, 2016, from http://juteyarn-bjsa.org/export.php