

Palestine Polytechnic University



College of Information Technology and Computer
Engineering

Information Technology Department

Using Computer Vision for Plant Disease Detection

A case study for cucumber downy mildew disease

Project Team

Nermeen Alami and Shahd Ewawi

Project supervisors

Dr. Hashem Tamimi and Dr. Rami Arafeh

This project is presented to the department of information technology at college of Information
Technology and Computer Engineering, for partial fulfillment Bachelor of information
Technology degree requirements

2013

Abstract

Plants diseases cause significant damage and economic losses in crops worldwide. We propose a computerized plants diseases detector as an applied research project that targets crop production especially in large farms. The project is based on image processing techniques that can detect the infected plants using color and texture information appearing on plant leaves. As a sample we chose cucumber plant because disease symptoms appear on its leaves quickly. The proposed system monitors crops and provides an early warning when the crop is infected with a certain foliar disease.

We rely on capturing the image and segmenting the leaves using color-based segmentation, then detecting if parts of the leaves have given colors and textures that indicate an infection. Once an infection is detected, the severity is measures, stored and compared for monitoring the plants while appropriate action for disease control.

الخلاصة

هذا المشروع عبارة عن مشروع بحثي يستهدف خدمة قطاع الإنتاج النباتي خاصة في حالة وجود مساحات شاسعة من المحاصيل وما يصاحب ذلك من مشكلة مراقبة المحصول بالطرق التقليدية أو العين المجردة. يعتمد هذا المشروع بشكل أساسي على استخدام تقنيات معالجة الصور لتحديد النباتات المصابة بناءً على التغيرات اللونية الظاهرة على سطح أوراق النبات. سيتم اختيار نبات الخيار كعينة مبدئية لتطبيق النظام وذلك لعدة أسباب أهمها سرعة ظهور الأعراض على الأوراق بالإضافة لكبر حجم الورقة وهذا ما سوف يسهل عملية تعقب الأوراق باستخدام الكاميرا وسهولة تمييز أعراض المرض عليها. هذا النظام يقوم بمراقبة المحصول وتزويد المزارع بتنبيهه عند حدوث إصابة في الحقل . سيتم تطبيق ذلك عن طريق النقاط صور لنبات معين ومن ثم تحليل هذه الصورة واستخلاص خصائص معينة منها قد تم تحديدها مسبقاً ومن ثم تحليل هذه الخصائص لمعرفة نوع المرض وشدته وبالتالي تحديد كيفية التعامل معه ومكافحته. باستخدام هذا النظام يمكن تفادي خسائر اقتصادية فادحة وذلك عن طريق الكشف المبكر عن الأمراض وهذا يساعد المزارع في اتخاذ الإجراءات المناسبة وعلاج المحصول في الوقت المناسب.

Acknowledgment

This graduation project has been supported by the Deanship of Graduate Studies and Scientific Research through "Distinguished Graduation Project Fund" – Palestine Polytechnic University.

We owe thanks to Dr.Hashim Tamimi and Dr.Rami Arafeh, the supervisors of this project for guiding and correcting our work with attention and care.

Table of Contents

Title	Page
Abstract	I
Table of Contents	III
Acknowledgement	V
List Of Figures	VI
List Of Tables	VII
CHAPTER ONE: INTRODUCTION	
Problem Statement	2
Project Importance	2
Project Objectives	3
Methodology	3
Project Block Diagram	3
Expected Results	4
CHAPTER TWO: Literature Review	
Plant diseases detection	6
infectious disease	6
Downy mildew disease	6
Available methods to detect plant diseases	7
ELISA	7
Magnetic reader	8
Profiling of volatile organic compounds	9
Visible and infrared stereoscopy	9
WinFOLIA system	9
CHAPTER THREE: Background	
Image Processing	11
Image Filtering	12
Mean Filtering	12
Median Filtering	13
Gaussian filtering	14
Converting the color of the image from (RGB) to (HSV)	15
What is RGB model	15
What is HSV model	16

Image Segmentation	17
Region Based Image Segmentation	17
Connected components labeling	18
CHAPTER FOUR: Requirements and specification	
Data collection	20
Collecting images of a cucumber leaf	20
Determining the constraints of the project	21
DataBase	21
User Interface	22
System Environment	25
System Requirements	26
User Interface	28
System environment	31
Using Qt to implement the project	32
CHAPTER FIVE: Implementation	
System Simulation	32
System Real time implementation	34
System Installation	35
System testing	35
CHAPTER SIX: Conclusion	
Results	42
Recommendations	42
Future Developments	42
Conclusion	42
References	43

List of Figures

Number	Title	Page
Figure1.1	Block diagram of the system	3
Figure2.1	Cucumber plant infected by Downy mildew disease	6
Figure2.2	The Magnetic reader	7
Figure2.3	Leaf image acquisition using WinFOLIA system	9
Figure 3.1	An image with salt and paper noise, image after denoising process using mean filter.	13
Figure 3.2	An example shows how median filter works.	13
Figure 3.3	An image with salt and paper noise, image after denoising process using median filter.	14
Figure 3.4	An image with salt and paper noise, image after denoising process using Gaussian filter.	14
Figure 3.5	RGB color model	15
Figure 3.6	HSV color cone	16
Figure 3.7	Image is in RGB model, image converted to HSV model	16
Figure 3.8	Image before applying segmentation process, b: image after applying segmentation process	17
Figure 4.1	Samples of infected cucumber leaves	20
Figure 4.2	New test interface	23
Figure 4.3	Get chart interface	24
Figure 4.4	System environment	26
Figure 4.5	Describes and summarized the methodology that is used in this system.	28
Figure 5.1	Image in RGB color space, image a converted to HSV	33
Figure 5.2	Image with salt and paper noise, image after denoising	33

	process using median filter	
Figure 5.3	Image with salt after segmentation	34
Figure 5.4	Infected regions	34
Figure 5.5	Capturing an image inside the system.	36
Figure 5.6	An image before isolation, an image after isolation.	37
Figure 5.7	An image before filtering, an image after filtering	37
Figure 5.8	Figure 5.8 a: an image in RGB color space, b: an image in HSV color space.	38
Figure 5.9	Shows infected regions of a leaf.	39
Figure 5.10	shows connected component labeling	40

List of Tables

Number`	Title	Page
Table 4.1	Test table	22
Table 4.2	Detail the contents of New Test interface	23
Table 4.3	Detail the contents of get chart interface	25

Chapter One

Introduction

Contents:

- 1.1 Problem statement
- 1.2 Project importance
- 1.3 Project objectives
- 1.4 Methodology
- 1.5 Project block diagram
- 1.6 Expected results

Overview

Technology is influencing many aspects of our lives, where the direction and the nature of progress is influenced by the accelerated growth of scientific discoveries and technological innovations, and the employment of this knowledge in improving all sciences including biotechnological science.

Plants diseases have been controversial because of their negative impact on agricultural industry. Plants diseases cause significant damage and economic losses in crops worldwide. So, we propose and experimentally evaluate a software solution for automatic detection and classifications of plant diseases by automatically detect the symptoms of diseases as soon as they appear on plant leaves.

In this chapter, we will present the problem statement, project objectives and project importance. We will also explain the methodology to be followed in building the system.

1.1. Problem statement:

The naked eye observation of experts is the main approach adopted in practice for detection and identification of plant diseases. However, this requires continuous monitoring of the experts, which might be prohibitively expensive in large farms. Further, in some developing countries, farmers may have to go long distances to contact experts, this makes consulting experts too expensive and time consuming.

Therefore we propose an automatic system for plant disease detection based on image processing, where the computer can take the role of the expert.

1.2. Project importance:

The system importance can be summarized in the following aspects:

1. A better description of the relationship between the environmental conditions and disease level, which could be useful for disease management.
2. High accuracy in the process of analyzing the results, which is one of the basic requirements of any experiment especially in biological systems.
3. Facilitate the processes of identifying and treatment of plant diseases.

4. Enrich the culture of farmers about the amounts and method of using pesticides, after better determining the plant disease identity.

1.3. Project Objectives:

We seek to develop a computerized system that determines whether a certain plant is infected with a disease or not, determine the plant disease identity, and the severity of it. This can be performed by capturing an image of a certain plant leaf, then extracting a predefined feature from the captured image and analyzing these features. Based on image processing techniques we can know the type of the disease we are dealing with and the severity of it.

1.4. Methodology:

This system is based on the process of determining the symptoms of the disease, where the disease appears on the form of speckles on the upper or the lower surface of the plant leaf or a combination of both. Plant disease identity is determined based on the color of the speckles, where the color of the speckles varies from one disease to another. The severity of the disease is determined based on the area covered by the speckles of the total plant area.

1.5. System block Diagram:

First, a special digital camera will automatically capture images of leaves. The system will process and enhance the image. Then the system will extract predefined features from the image. Figure 1.1 illustrates the main steps performed in this project.

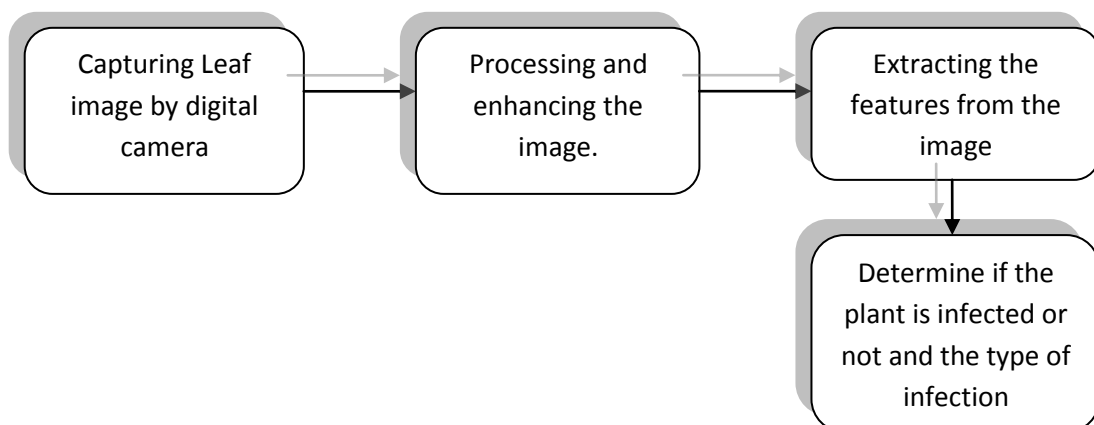


Figure 1.1 Block diagram of the system.

1.6. Expected Results

The expected results of the project are:

1. A computer based method that can read a captured image of a plant leaf and give a statement and numbers that describes this leaf from the image with minimum user intervention.
2. The project can read a sample of plant leaves and classify them as infected of a disease or not, type of infection and what is the severity of infection.

Chapter Two

Literature Review

Contents:

2.1 Plant diseases detection

2.1.1 Infectious disease

Downy mildew disease

2.2 Available methods to detect plant diseases

1. ELISA
2. Magnetic reader
3. Profiling of volatile organic compounds
4. Visible and infrared stereoscopy
5. WinFOLIA system

Overview

This chapter reviews the previous and related works in the field of detecting plants diseases. In addition, explaining what do we mean by image processing.

2.1. Plants diseases detection

Plants diseases detection is identifying the disease based on the plant reaction toward the causal agent. Often plants diseases are caused by a parasite, which locates plants diseases under the department of infectious diseases (Olsen, 2011).

1. Infectious diseases

Infectious disease characterized by the presence of the causal agent in or on the plant. Sometimes with the naked eye or by a magnifying glass we can identify the causal agent, but microscopic examination is needed for detailed identification of particular causal agents. If the causal agent does not exist on the surface of the plant it is necessary to consider additional symptoms. Usually the causal agent can be found at the edges of the infected tissue, vascular tissue, at the base of the plant or the roots. Here is one of the most common infectious diseases (Becker, & Miller, 2009):

Downy mildew

Appear in the form of pale yellow spots on the upper surface of the leaves. As the infection progresses the leaf color becomes dark gray or brown, corresponding growth on the bottom surface of fluffy white or gray color is a Tripods germ bags of fungus that come out of the mouths of the leaf. Fungi that cause this disease are mandatory snooping and specialized.

These fungi atmosphere spread with high humidity and temperatures medium which tend to be cold. Infection occurs through stomata. As the injury progresses the spots color turns to brown or reddish-brown. Severe injuries lead to stunting of plants defoliation and poor fruit production. Figure 2.1 show a cucumber plant leaf infected with downy mildew.



Figure 2.1 Cucumber plant infected by downy mildew disease (Holmes, 2008).

2.2. Available methods to detect plants diseases

Plants diseases detection can be done by several direct or indirect methods (Sankaran et al., 2010). Direct methods include Serological methods (such as ELISA method) and Molecular methods (such as the magnetic reader). Indirect methods include plant properties based disease detection (such as Visible and infrared spectroscopy) and Biomarker-based diseases detection (such as Profiling of plant volatile organic compounds) (Sankaran et al., 2010). Here are some of the available methods to detect plants diseases:

1. ELISA

In the past, in order to detect plants diseases farmers used to apply the ELISA method. ELISA stands for Enzyme-linked Immunosorbent Assays. In this method the farmer need to collect plants samples and sends them to laboratories. The ELISA method is a traditional method to detect plants diseases; it depends on antibody antigen reaction. Still, the test is sensitive, good for large scale testing. This test is expensive and requires two weeks to show results, by then the disease has spread (Sankaran et al., 2010).

2. Magnetic reader

Currently, plants diseases can be detected by several methods e.g. for sensitive detection of plants diseases magnetic reader can be useful. Magnetic reader was invented by scientists at the Peter Gruenberg institute (Schroper et al., 2011). This device contains excitation coils and detection coils arranged in pairs.

Excitation coils generates high and low frequency magnetic field. The detection coils measures this field. The result is displayed on a screen, where the user views the conclusions about the concentration of particles in the magnetic field. Researchers use this method to track pathogens. In this method, what is tracked not the virus itself but magnetic particles attached to it. Magnetic Particle equipped with antibiotics to attached pathogens. Researchers use a way works in a way similar to ELISA method, where the plant extracts is placed in a tiny tube filled with a polymer matrix. Then the plant extracts pass the tube and the virus particles are trapped in the matrix. Then the particles that are not useful are removed and the concentration of particles is measured. This method is very sensitive but farmers face difficulty in dealing with large number of analytical steps, it takes large detection time and it is not user friendly (Schroper et al., 2011). Figure 2.2 shows the magnetic reader.



Figure 2.2 The magnetic reader (Schroper, & Schillberg, 2011).

3. Profiling of plant volatile organic compounds

An organic compound released by plants depends on several factors, including temperature, humidity, soil, light and fertilizer. These factors directly or indirectly affect the volatile organic compounds and thus affect the relationship between plants and other organisms such as pathogens. In Profiling of plant volatile organic compounds method the focus is on monitoring the plants volatile organic compounds profile. Volatile organic compounds released by plants changes when the plant is infected by a disease. The system contains gas sensors that are sensitive to specific ranges of organic compounds. The cost of this technique depends on the desired accuracy for volatile organic compounds profiling. The accuracy of this system is unknown (Sankaran et al., 2010).

4. Visible and infrared spectroscopy

This method relies on the analysis of the visible and infrared rays. These rays provide information on the stress level in plants. The disease can be identified based on the wavebands, where each disease has a specific waveband. The higher the visible symptoms, the better are the accuracy of this technique.

This method in combination with other methods as molecular detection can be effective in rapid detection of plant diseases. This technique was used in detecting fire blight disease in the asymptomatic pear plants. The technique was unable to determine the infected plants from healthy ones; this is due to the small size of leaf scan area (Sankaran et al., 2010).

5. WinFOLIA system

For fast detection of plants diseases WinFOLIA system can be used (Regent Instruments Inc., 2012). This system depends on analyzing images of plants that do morphological measurements on plants leaves. In this system there are two ways to acquire images, either by using a desktop scanner or by using digital cameras.

When using a desktop scanner the images that are captured are free of illumination. It can be used in the field while leaves are attached to the plant. For digital camera a base can be used to hold it. The WinFOLIA (figure 2.4) system is made of a plastic background which is transparent and anti-reflective. It can be placed on soil or held in hands. WinFOLIA measures the leaf area, length and width in a simple manner (in the

horizontal and vertical directions), perimeter, holes area and some other measurements. It quantifies the area per color. It can classify a leaf as diseased if the diseased color group has the largest area or is larger than a percentage that you specify (Regent Instruments Inc., 2012). Since WinFOLIA cannot determine if a certain plant is diseased unless diseased color group has the largest area, it is too late to cure the diseased plant. Figure 2.3 shows the process of leaf acquisition using WinFOLIA system.

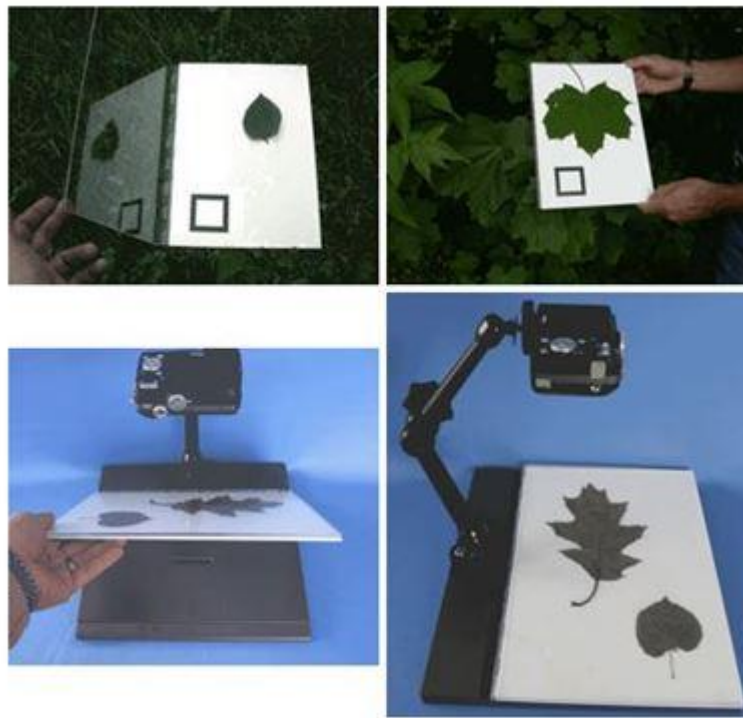


Figure 2.4 Leaf image acquisition using WinFOLIA system (Regent Instruments Inc., 2012).

Chapter Three

Background

Contents:

- 3.1 Image Processing
 - 3.1.1 Image Filtering
 - 3.1.2 What is Image Filtering?
 - 3.1.3 Mean Filtering
 - 3.1.4 Median Filtering
- 3.2 Converting the color of the image from Red, Green, Blue (RGB) to Hue, Saturation, Value (HSV)
 - 3.2.1 What is RGB model
 - 3.2.2 What is HSV model
- 3.3 Image Segmentation
 - 3.3.1 Region Based Image Segmentation
- 3.4 Connected Components Labeling

Overview

The field of digital image processing refers to processing digital images by means of a digital computer (Gonzalez et al., 2007). This is done for the purpose of creating an enhanced image that is more useful to satisfy human observer, or to perform some of the interpretation and recognition tasks usually performed by computer. Image processing has shown remarkable progress in the past decade in both terms of theoretical development and applications (Gonzalez et al., 2007). It is used in many areas, including astronomy, medicine, industrial robotics and many other fields.

There are many theories for image processing, but in this section, the theories that are needed to develop this project will be introduced.

2.1. Image Processing

The field of digital image processing refers to processing digital images by means of a digital computer (Gonzalez et al., 2007).

The image passes through several stages:

- Image acquisition: the image can be captured using a digital camera or a scanner.
- Pre-processing: such as filtering noise or converting the image to binary.
- Segmentation: isolating an object in the image from the background.
- Features extraction.
- Classification: based on predefined type.
- Image understanding.

Digital Image

An image may be defined as a two-dimensional function $f(x, y)$, where x and y are spatial coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity of the level of the image at that point. When x , y , and f are all finite, discrete quantities, we call an image a digital image (Gonzalez et al., 2007). Digital image quality depends on the number of pixels in the image. The greater the number of pixels

the better the quality. If the digital image was enlarged to a certain limit, the image becomes distorted.

Digital images are divided into:

- Binary image: this image contains black and white colors and each pixel can be either black or white.
- Grayscale image: contains black, white and grayscale. The intensity of the image can be represented by numbers from 0 to 255. When representing an image in the computer the pixels arranged in rows and columns.
- Color image: the image supports colors by allocating three matrices in each pixel to determine the intensity of the three key colors (Red, Green, and Blue) (Gonzalez et al., 2007).

The following is an image processing technique for detecting plants diseases:

3.1. Image filtering

Sometimes, images cannot be used directly due to having poor contrast or random differences in intensity or in illumination. Therefore image filters are applied.

3.1.1. What is image filtering?

Filtering can be defined as modifying the values of pixel intensity based on neighborhood to make certain image characteristics revealed, Such as enhancement by improving contrast and smoothing. In our case, we will use filtering for denoising leaves images.

The type of the filter mainly depends on the problem. There are several types of filters such as mean, gaussian and median filters. In this section we will explain them.

3.1.2. Mean filter

Mean filter replaces each pixel with an average of its neighborhood. The process of calculating the average removes some of the higher frequency pixels, while keeping the low frequency pixels unchanged. The efficiency of this filter depends on the characteristics of the input image. If the high frequency pixels represent noise that decreases the image quality, the mean filter will improve the image. However, the use

of this filter may remove useful image details, and reduce contrast (Jain et al., 1995). Figure 3.1 shows the denoising process using mean filter.

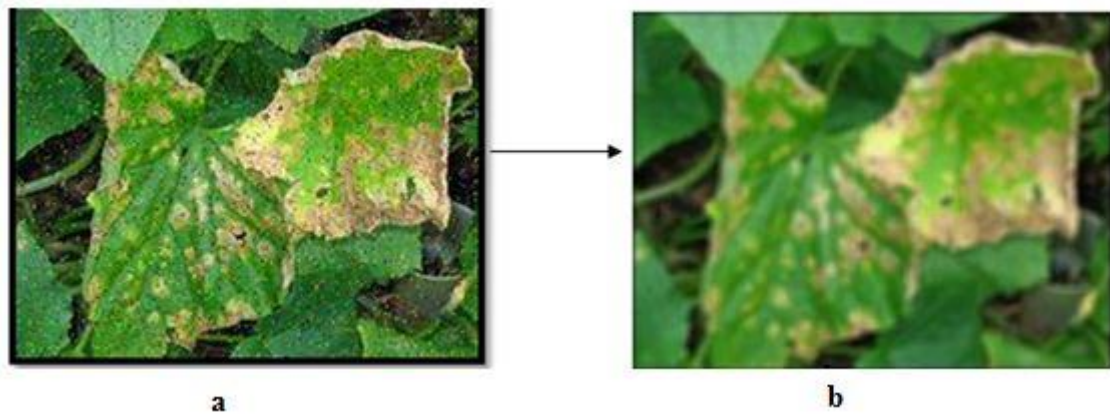


Figure 3.1 a: image with salt and paper noise, b: image after denoising process using mean filter

3.1.3. Median Filter:

The problem with mean filter is that they tend to remove useful image details from an image. An alternative approach is use the median filter. The idea of this filter depends on taking a window (for example a 3 x 3) the window slides along the image. Then we compute the median of the pixels in each window by sorting the window values into ascending order and then selecting the center value. The median value of the pixels in the window becomes the value of the pixel being processed.

For example, figure 3.1 shows an example of how the median filter works. The values in a window are 7, 12, 6, 10, 44, 11, 9, 13 and 8 the pixel that will be processed is 44. We sort the values in an ascending order (6, 7, 8, 9, 10, 11, 12, 13, 44). The value of the pixel being processed becomes 10. The median filter is considered effective because it does not depend on values that are different from values in the neighborhood (Jain et al., 1995). Figure 3.2 shows an example of how median filter works

7	12	6
10	44	11
9	13	8

→

7	12	6
10	10	11
9	13	8

Figure 3.2: an example shows how median filter works



Figure 3.3 a: image with salt and paper noise, b: the image after denoising process using median filter.

3.1.4. Gaussian Filter

The Gaussian smoothing filter is a very good filter for removing noise drawn from a normal distribution. The one dimensional Gaussian function is (Jain et al., 1995):

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

Gaussian filter removes noise by replacing each image pixel with a weighted average of the pixel neighbors such that the average given to a neighbor decreases with distance from the central pixel (Jain et al., 1995). Figure 3.4 shows the results of denoising process using Gaussian filter.

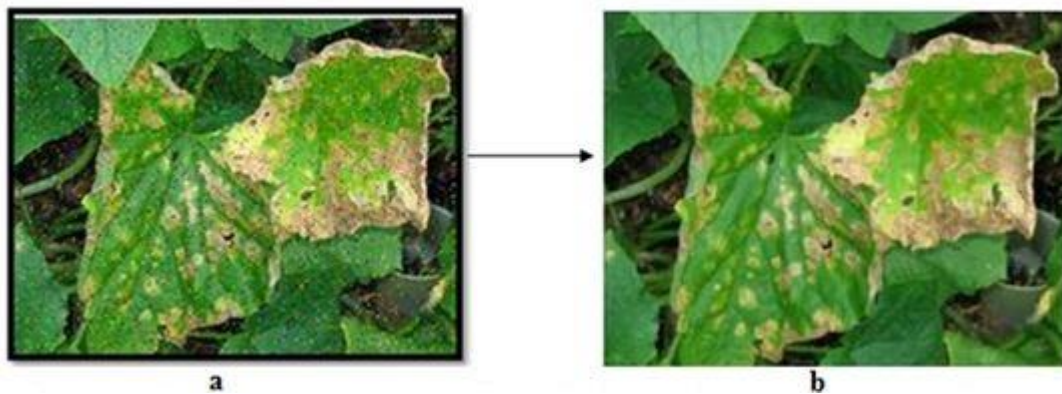


Figure 3.4 a: image with salt and paper noise, b: the image after denoising process using Gaussian filter.

As we can see from the previous figures Gaussian and median filter have better results than mean filter in removing salt and paper noise.

3.2. Converting the color of the image from Red, Green, Blue (RGB) to Hue, Saturation, Value (HSV)

In this project, the main image processing technique that will be used is converting the color of the image from Red, Green, and Blue (RGB) to Hue, Saturation, and Value (HSV).

Color vision can be processed using RGB color space or HSV color space. RGB color space describes color in terms of the amount of red, green, and blue present. HSV color space describes colors in terms of the Hue, Saturation, and Value.

3.2.1. RGB color model:

RGB color model: an RGB color image is an $M * N * 3$ array of color pixels, where each color pixel is a triplet corresponding to the red, green and blue components of an RGB image at specific spatial location (Gonzalez et al., 2007) All the colors can lie in a cube extending from the origin (black) as illustrates in figure 3.5.

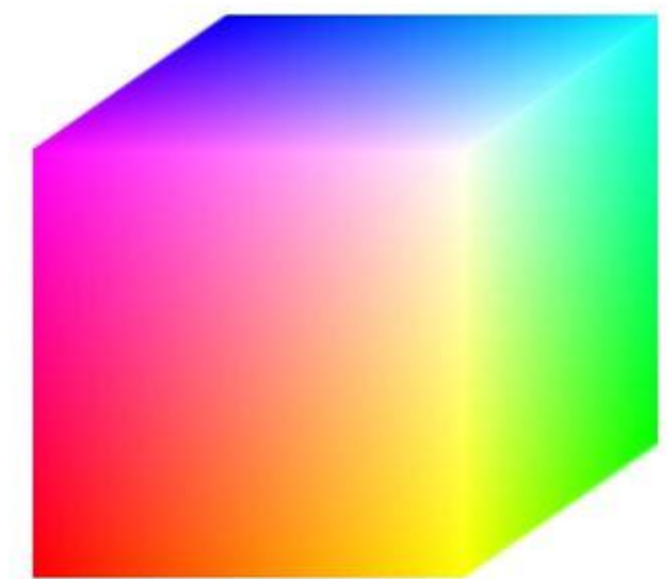


Figure 3.5 RGB color space (Schouten, 2003).

All the colors you can make lie in a cube extending from the origin (black)

3.2.2. HSV color model:

HSV color model: HSV is one of several color systems used by people to select colors from a color wheel or palette (Gonzalez et al., 2007). Hue is expressed as a number from 0 to 360 degrees representing hues of red (starts at 0), yellow (starts at 60), green (starts at 120), cyan (starts at 180), blue (starts at 240), and magenta (starts at 300). Saturation is the amount of gray (0% to 100%) in the color. Value (or Brightness) works in conjunction with saturation and describes the brightness or intensity of the color from 0% to 100%. All the colors can lie in a cone as illustrates in figure 3.6.

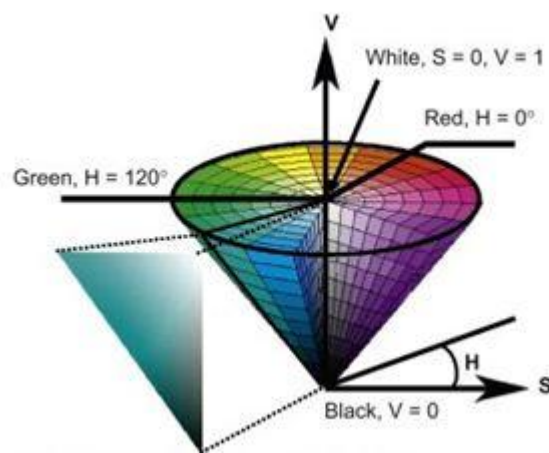


Figure 3.6 HSV color cone (Lai, & Chien-Yin, 2010).

Unlike RGB, HSV separates image intensity, from the color information. This is very useful in many applications. The other advantage of HSV is that each of its attributes corresponds directly to the basic color concepts, which makes it conceptually simple. An example of an image in RGB and HSV models is shown in figure 3.7.

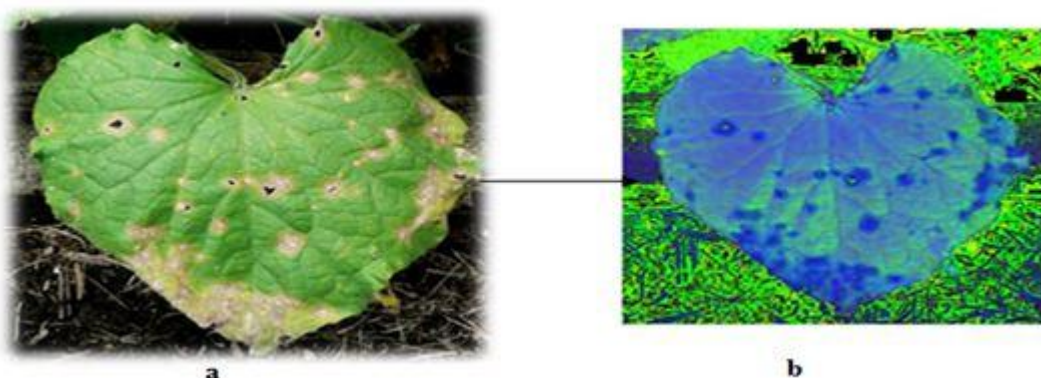


Figure 3.7 a: image is in RGB color space, b: image converted to HSV model.

3.3. Image Segmentation

The objective of segmentation is to partition an image into regions. Segmentation algorithms depends on one of two properties of values, it may depend on discontinuity which means partitioning an image based on sharp changes in intensity (such as edges). The second property is similarity which means partitioning an image into regions that are similar according to a set of predefined criteria (such as color) (Gonzalez et al., 2007).

3.3.1. Region-based image segmentation

A Region is a group of connected pixels that share similar properties. The goal of Region-based segmentation is to find coherent regions in the images, which contains pixels that share some similar property to make it more meaningful and easier to analyze.

First, we start from one seed pixel located inside region. Then, we should define a similarity measure for all pixels (such as color) in the image. We add adjacent pixel to the region of the seed pixel if the similarity measure of the seed pixel and the adjacent pixel is greater than a predefined threshold. We evaluate the other neighbors of the seed pixel as above; we can now consider the adjacent pixel as a new seed. Finally, we continue until all pixels in the currently investigated neighborhood satisfy the inclusion criteria (Jain et al., 1995). Figure 3.8 shows the results of segmentation process

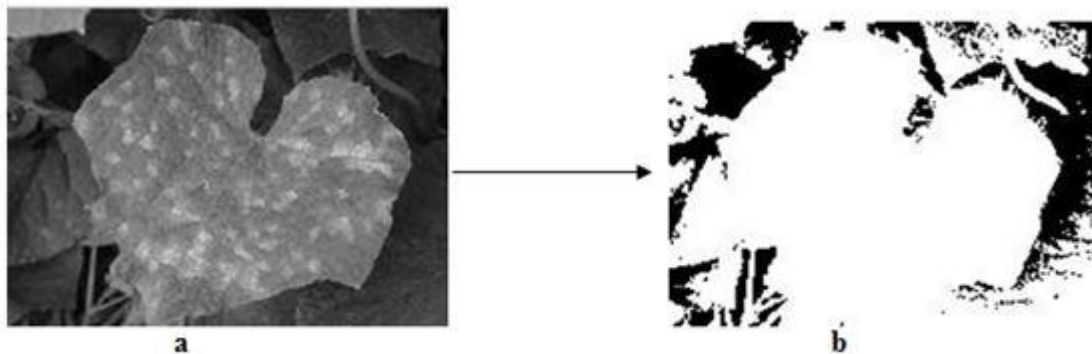


Figure 3.8 a: image before applying segmentation process, b: image after applying segmentation

3.4. Connected Components Labeling

Connected components labeling is a relatively simple grouping algorithm that has been used to isolate, measure and identify potential object regions in an image. It is widely used in applications where an image often consists of objects against a contrasting background. Such images may be binarized yielding data that retains useful shape and size information of the objects under observation. The labeling operation assigns a unique name or number to all 1-pixels that belong to the same connected component of the image. As a result of the labeling, individual components can be extracted from the image programmatically and therefore is available for further processing and analysis (Jankowski, 2009).

Connected components labeling is used in our system in order to improve accuracy. This algorithm is used to determine the number of pixels in each region to remove regions with large number and small number of pixels. We cannot consider regions with large number of pixels as infected because disease symptoms appear as regions with medium number of pixels. As for regions with small number of pixels, it is also cannot be considered as infected areas, usually it is considered as noise.

Chapter Four

Requirements and Specifications

Contents:

4.1 Data Collection

Collecting images of a cucumber leaf

Determining the constraints of the project

4.2 Database

4.3 User Interface

4.4 System Environment

4.5 System Requirements

4.6 Software specification

Overview

This chapter covers the preparation steps that must be followed before starting implementation, the tools that were used to implement this project and the methodology that is used in this project to detect plants diseases.

4.1. Data Collection

The preparation steps include collecting images of a cucumber leaf and studying them and then defining some constraints on these images to make project successful.

Collecting images of a cucumber leaf

Because cucumber plant is taken as a case study in this project, we start collecting images for a cucumber leaf infected with Downy mildew disease as well as images of leaves not infected at all. Cucumber plant was chosen due to the large size of the cucumber leaf which makes it easy to track. These images are obtained from a greenhouse in Hebron city, by a digital camera with high resolution. Figure 4.1 shows samples of an infected cucumber plant leaves in late stages.

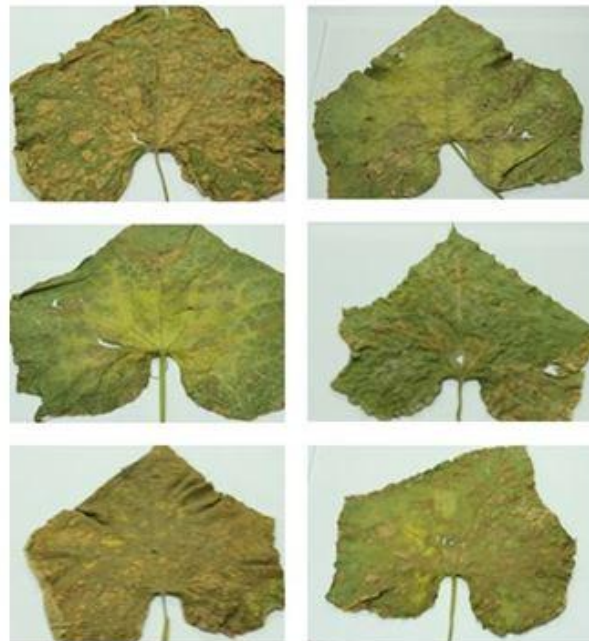


Figure 4.1 samples of infected cucumber leaves

Determining the constraints on the data

Plants diseases detector system has some constraints; in this section we will list some of them.

1. The leaves are digitized using a digital camera with very high resolution at least (640 * 450). The camera automatically takes snapshots of the plant leaves.
2. The digital camera is connected to a standalone device which sends the snapshots to a computer.
3. The image should be a complete leaf of plant not leaflet.
4. The images are captured in RGB format and the leaves are isolate from background.

4.2. Database:

- In our system we have only one table shown to store the results of the tests on each leaf, see Table (4.1), in which attribute column has attribute names, description column represent a brief explanation of the attribute, type represent the data type that will be used for each attribute, size represent the size of the attribute and notes column contains a brief notes about each attribute. Table 4.1 shows the attributes of the test table.

Table (4.1): Test table.

Attribute	Description	Type	Size	otes
<u>Image_id</u>	The ID of the image(Serial number)	Number	6	PK
Time	Time of capturing image automatically by camera (Automatic by system)	Time	6	Not null
Image	the path of the captured image	String	100	Not null
Degree	The severity of infection	Number	6	Not null
Comments	the user comments about the infection	String	200	

4.3. User Interface

In this part the interfaces are going to be displayed and explained.

- New Test interface. When the user clicks on Take shot button an image will be captured in order to be tested. Save button will save results in database when it is clicked. Exit button is used to close the screen. Using this interface the user can see the results of the system as shown in figure 4.2.

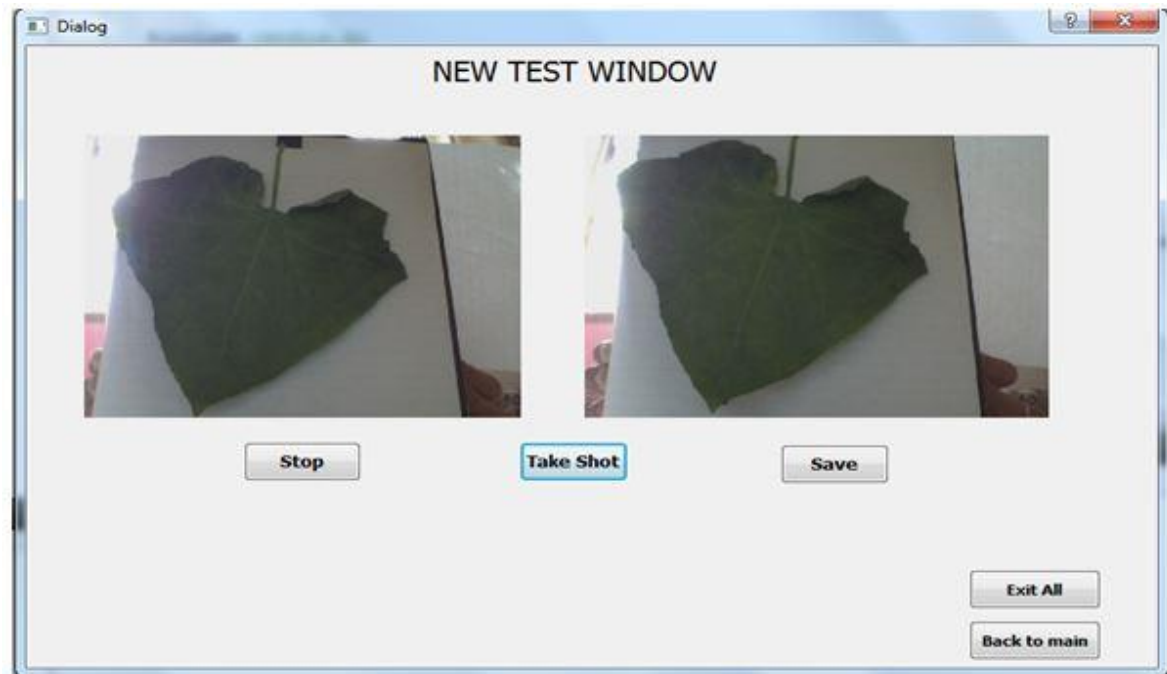


Figure 4.2 New test interface

- Detail the contents of New Test interface: this table explains the toolboxes used in the interface. Name the toolbox column shows the name in the interface, the type column displays the type of the toolbox, data type receipt column shows the value received after the execution of toolbox action and description column describe the function of the toolbox. Table (4.2) has details of the content of new test interface.

Table (4.2): Detail the contents of New Test interface

Name the Toolbox	Type	Data Type Receipt	Description
Save results	Button	Null	This button used to save results in the database
Take shot	Button	Null	Used to capture an image
Exit All	Button	Null	Used to close the screen

- **Chart Interface**

By this interface user can get a graph that shows the change in disease severity during system life, figure 4.3 shows get chart interface.

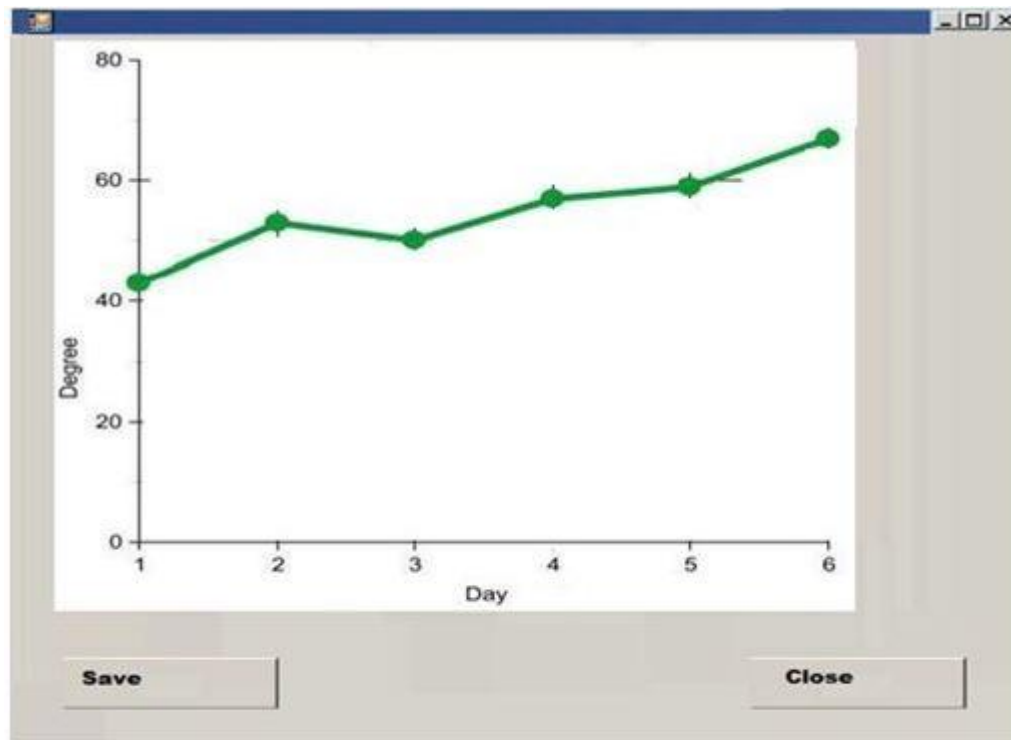


Figure 4. 3 Get chart interface

- Detail the contents of get chart interface: this table explains the toolboxes used in the interface. Name the toolbox column shows the name in the interface, the type column displays the type of the toolbox, data type receipt column shows the value received after the execution of toolbox action and description column describe the function of the toolbox. Table (4.3) detail the contents of get chart interface

Table (4.3): Detail the contents of get chart interface

Name the Toolbox	Type	Data Type Receipt	Description
Chart	Image	Null	It represents the relationship between the infection severity and time in days.
Save	Button	Null	This button used to save the results as picture
Exit	Button	Null	Used to close the screen

4.4. System Environment

The system requires special instruments, in order to maintain good detection quality. The project environment must contain the following instruments:

1. A high resolution digital camera at least (640 * 450). The camera takes snapshots of the plants leaves. The camera is connected by a cable to the Digital Video Recorder (DVR).
2. DVR sends images captured by the digital camera to a server, which is connected to a computer.
3. The computer is the environment, where the system is installed. Figure 4.4 illustrates the system environment.

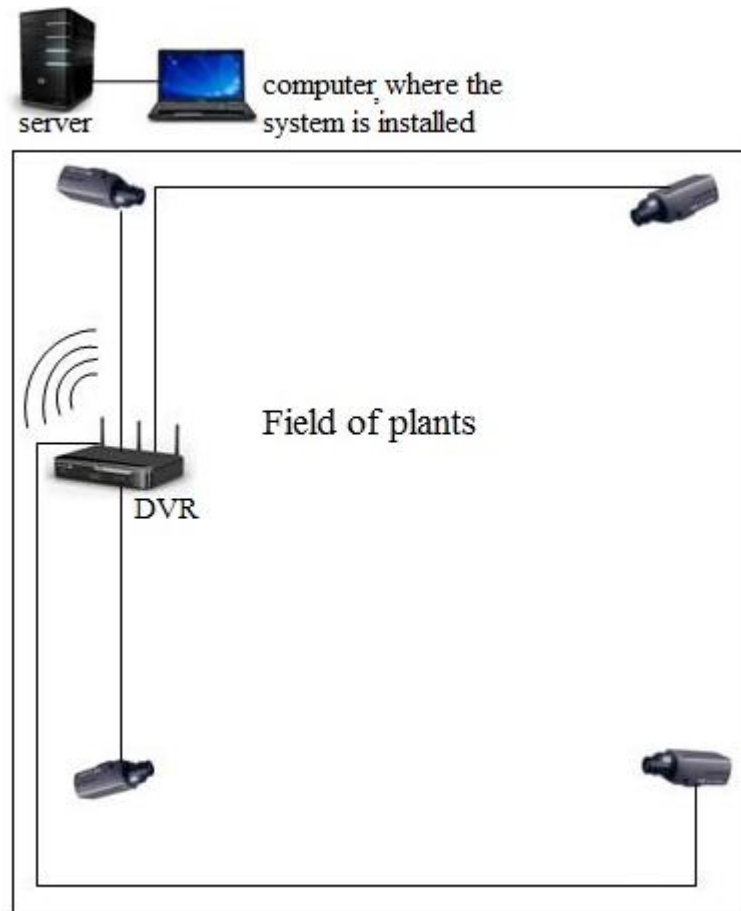


Figure 4.7 System Environment

4.5. System requirements

Hardware

Different hardware tools were used during the development of the project:

1. Digital camera to capture images every pre-defined period of time.
2. A computer where the system is installed.
3. Server where infection information about cucumber field are stored if we have wide fields.

4. DVR (digital video recorder) to transfer images captured by camera to the system where it can be processed.

4.6. Software Specifications

Different software were used during the development of the project

1- Matlab

We used Matlab for simulation process because of the following reasons:

- a. Matlab is an easy environment for simulation.
- b. It includes many image processing ready functions in the image processing toolbox.
- c. The programming language of Matlab is similar to C language which can help in the code translation.

2- C/C++

We use C/C++ for the real time implementation because C/C++ is very fast in execution and because we have excellent experience in it.

3- OpenCV

For image processing with C/C++ we use OpenCV library because it has many image processing functions and it can be easily integrated with C/C++.

4- QT for GUI

QT was proposed to be used in implementing the interface of this project. For the following reasons:

- a) QT is commonly used for interface design worldwide
- b) QT can be easily integrated with C/C++ and openCV
- c) The interface can be designed with drag/drop procedure and then a script can be produced from the designed interface.

Software design

The plant disease detection works as following: first, a digital camera will capture images of leaves every pre-defined period of time. The system will enhance the image using a filter in order to remove unwanted pixels (noise) from the leaf image. Then the enhanced image will be converted from RGB color space to HSV color space to perform using this color model. The system will then apply segmentation on the image for to isolate the leaf from the background of the image. In order to extract infected regions from the leaf image we apply another segmentation based on the color of these regions, where the values of pixels that comprise the infected regions are located within specific range. If the system finds pixels that hold a value located within the specified range and the region was of a given size, then the leaf will be classified as infected leaf, otherwise the leaf will be classified as not infected. Figure 4.5 shows these steps.

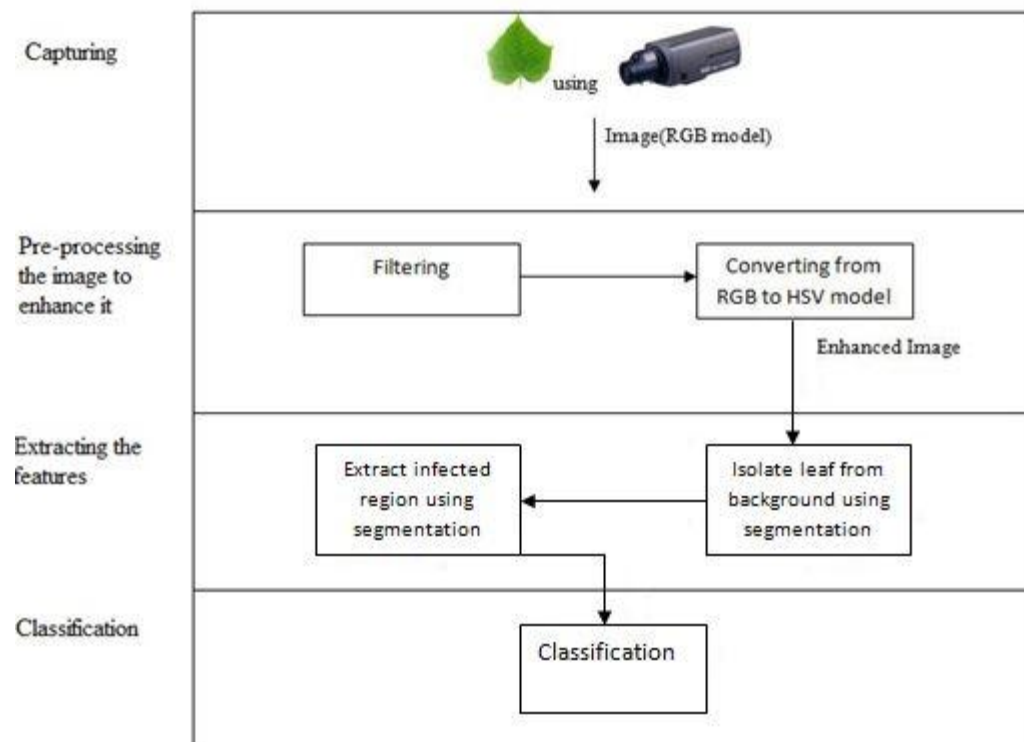


Figure 4.5 describes and summarized the methodology that is used in this system.

Processing the image

Before being able to detect infections, some processing is needed on the image. These processes include:

1. Reading the leaf image (RGB model): the model should be entered into the system in RGB model.
2. Converting image from RGB to HSV image: RGB input images should be converted to HSV color model. HSV separates image intensity, from the color information, which is useful in this project to facilitate the process of detecting the infected areas of the plant leaf and then determining the type of infection.
3. Using filtering to reduce the noise: As mentioned in chapter three there is many types of filters such as mean and median filters. Image filtering allows you to apply various effects on photos, which will be useful in enhancing images to be clearer to apply segmentation algorithms.
4. Applying segmentation on enhanced images: As mentioned in chapter three, segmentation is determining the boundaries of a region in an image, here the leaf of the plant should be isolated from background, this done using segmentation algorithms.
5. Another Segmentation will also be used to extract infected regions of the leaf image.
6. Connected components labeling is used in applied in order to improve accuracy. This algorithm is used to determine the number of pixels in each region to remove regions with large number and small number of pixels.

Pseudo-code of plants diseases detection

Repeat until test ends

 Capture image x1;

 Apply filtering to x1 producing x2;

 Apply segmentation to x2 using parameters [p1] producing x3;

 Join x3 with x2 producing x4;

 Apply segmentation to x4 using parameters [p2] producing x5;

 Apply connected components to x5 producing a set of components [c1, c2, ..., cn];

 For each component c1....cn

 If $T1 < \text{size}(cn) < T2$

 Label ci as infected;

 Else

 Label ci as not infected;

 endif

 Endfor

 For each ci when ci is infected component

 Count=count +1 ;

 End for

 Infected severity=count/total image size of x3

End repeat

Chapter Five

Implementation

Contents:

- 5.1 System simulation
- 5.2 System real time implementation
- 5.3 System installation
- 5.4 System Testing

Overview

This chapter talks about implementation phase which is divided into two sub phases simulation and real time programming in order improve accuracy and efficiency.

This chapter covers important phases of system development phases which are system programming, installation and testing, where the software tools and equipment that are used in developing the system will be identified, this will be done by analyzing the following:

- Software tools that are required to develop the system.
- System programming.
- System installation.

In this chapter, a set of algorithms were developed to apply them in the project, these algorithms are converted into implementation code. Some of these algorithms will be discussed in this chapter.

5.1 System Simulation

Simulation part was an important step in system development process. We used MATLAB program to simulate the system, which was a great choice because Matlab provides a comprehensive and flexible environment for image acquisition, analysis, processing, visualization, and algorithm development (Németh , 2013). We applied simulation to test the system using large number of samples and after we were sure of the results we then move to real time programming.

First, we used `imread` to read the image. Then image was converted from RGB to HSV color space using `rgb2hsv` function, where this function takes an RGB image as parameter and returns an HSV image. Then the hue, saturation and value images were extracted individually using `hsvImage(:, :, 1)`, where 1 represents H image, 2 represents S image and 3 represents V image. Then we applied each color band's particular thresholds to the color band. For example, `hueMask = (hImage >= hueThresholdLow) & (hImage <= hueThresholdHigh)`, where `hueThresholdLow` and `hueThresholdHigh` are pre-defined values. Masks were combined together using `uint8 (hueMask & saturationMask & valueMask)`.

The combined image is a black and white image. Finally we check the whole image for black pixels, if all of the image pixels are black then the leaf I infected, otherwise the leaf is not infected.

Results of MATLAB simulation

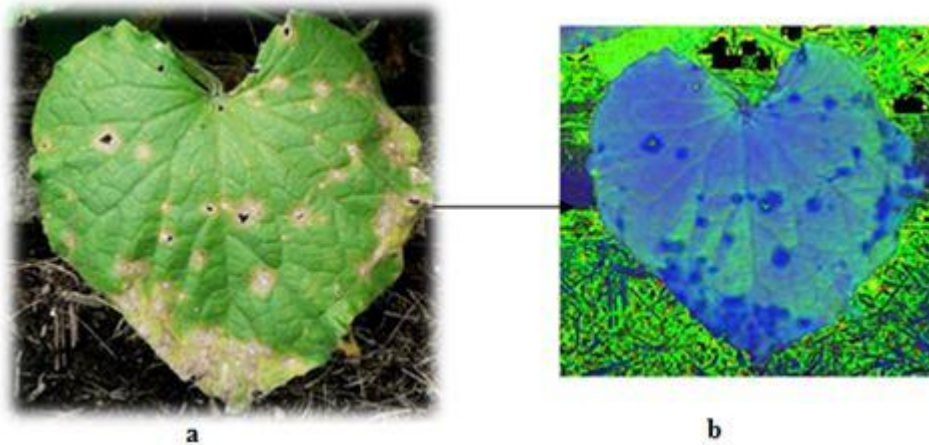


Figure 5.1 a: Image in RGB color space, b: Image a converted to HSV

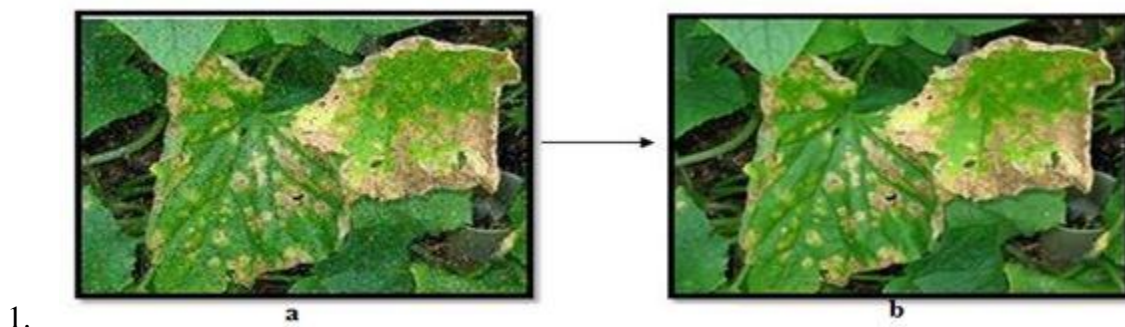


Figure 5.2 a: Image with salt and paper noise, b:Image after denoising process using median filter

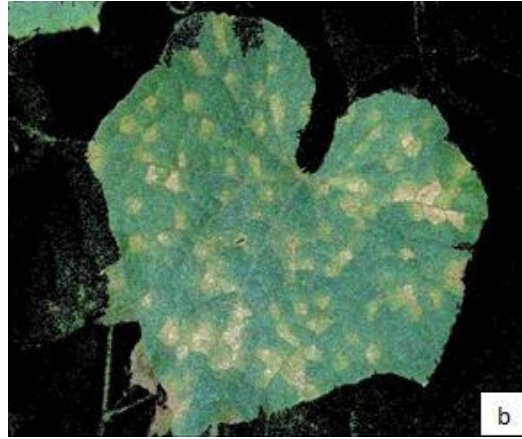


Figure 5.3 b Image with salt after segmentation

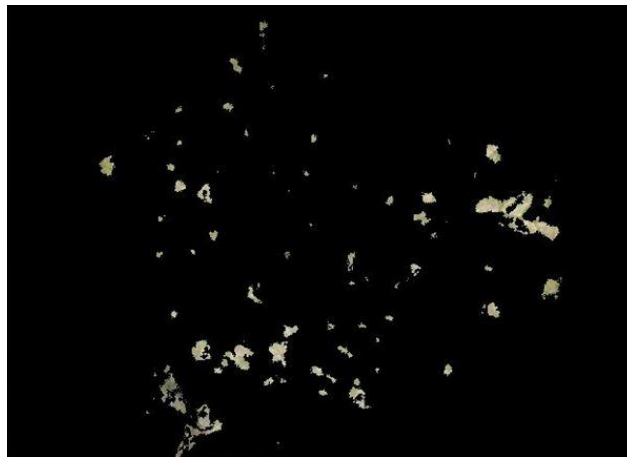


Figure 5.4 Infected regions

5.2 Real Time implementation

This is the second phase of implementation process in which the project was programmed using Qt program and OPENCV library.

First, imread was used to read the image. Then medianBlur function were used to filter the image aand remove unwanred pixels from it. The image was converted from RGB to HSV color space using cvtColor. For segmentation the function findContours where used to find connected regions based on high and low values of hue, saturation and value masks. The cvThreshold function were used to convert the image to black and white image. Finally we check the whole

image for black pixels, if all of the image pixels are black then the leaf is infected, otherwise the leaf is not infected.

5.3 System installation:

After the successful completion of preparation phase of writing algorithms and choosing tools needed in the project, creation of the database and user input, output and processing units and writing the code for each one of them. The system is ready to run and execute the recommended tasks, display the results to the user, getting data from user, saving into database and implement the needed operations.

To run the system, the following are needed:

- Compatibility between the recommended camera and computer, in which camera can work in the right way.
- Download the system to the new environment.
- Maintenance of the camera-system settings.
- After finishing the previous three steps you can run the system.

5.4 System Testing

In this phase system modules will be tested, in addition the whole system will be tested using black box testing, where testing process is done through making sure that the system achieves its desired requirements and needs and works as expected.

Testing operations

The process of testing the system consists of testing system modules and testing the whole system.

- **Testing system modules:**

In this type of testing every module of system modules will be tested separately through entering inputs and confirming outputs.

The first test is capturing images from the camera test. The team has been confirmed that the process of capturing images from the camera is done correctly (figure 5.5 Illustrates capturing an image using the camera inside the system).

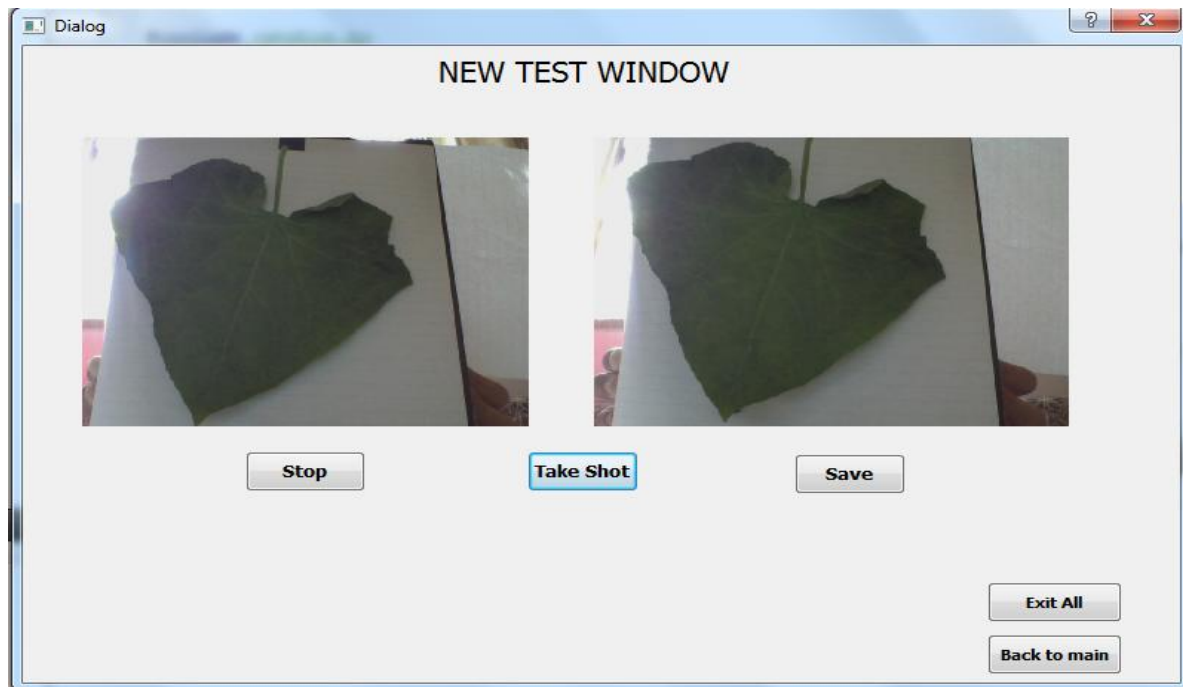


Figure 5.5 Capturing an image inside the system.

Testing the process of isolating the leaf image from the background. The process of isolating the leaf image from the background is done correctly by using segmentation method. (Figure 5.6 shows an image before and after isolation).

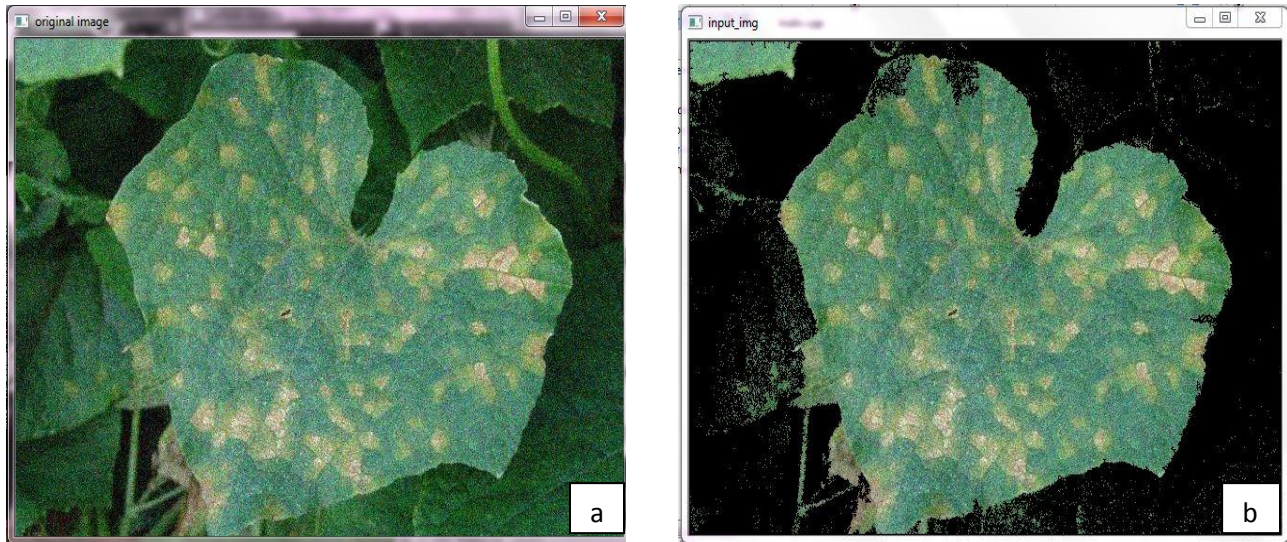


Figure 5.6 a: Image before isolation, b: Image after isolation.

Testing the process of enhancing the image. The process of enhancing the image by removing noise is done correctly by using median filter. The type of noise we are dealing with is salt and paper noise therefore we chose median filter for being the most effective filter for dealing with this type of noise. (Figure 5.7 shows an image before and after enhancement).

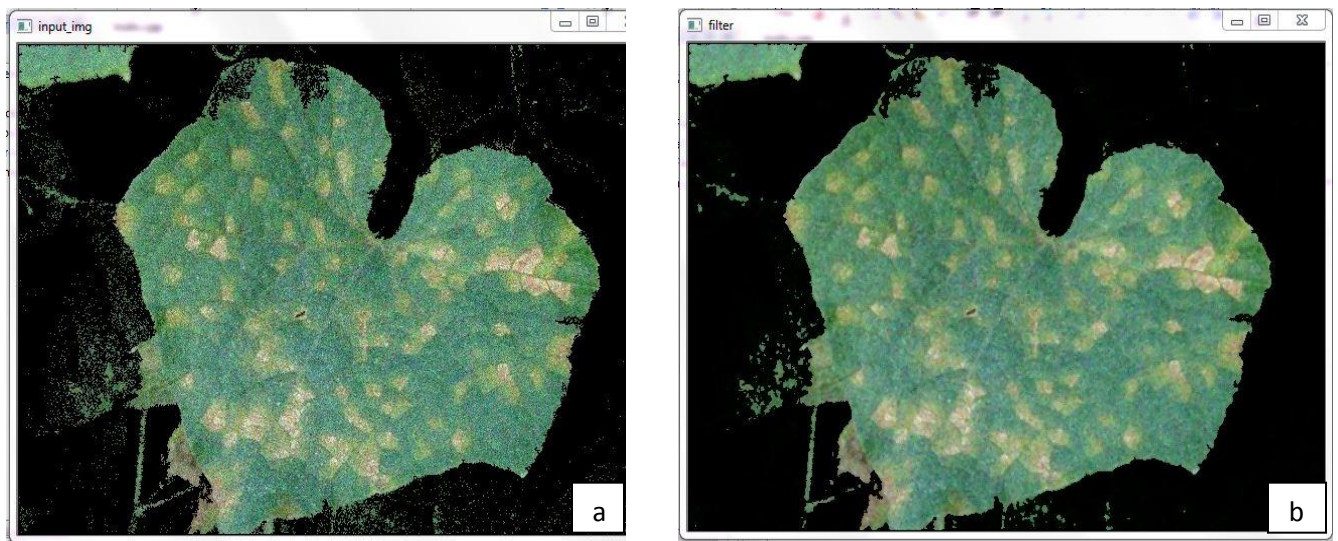


Figure 5.7 a: Image before filtering, b: Image after filtering.

Testing the process of converting the image from RGB color space to HSV color space. (Figure 5.8 shows an image before and after converting to HSV color space).

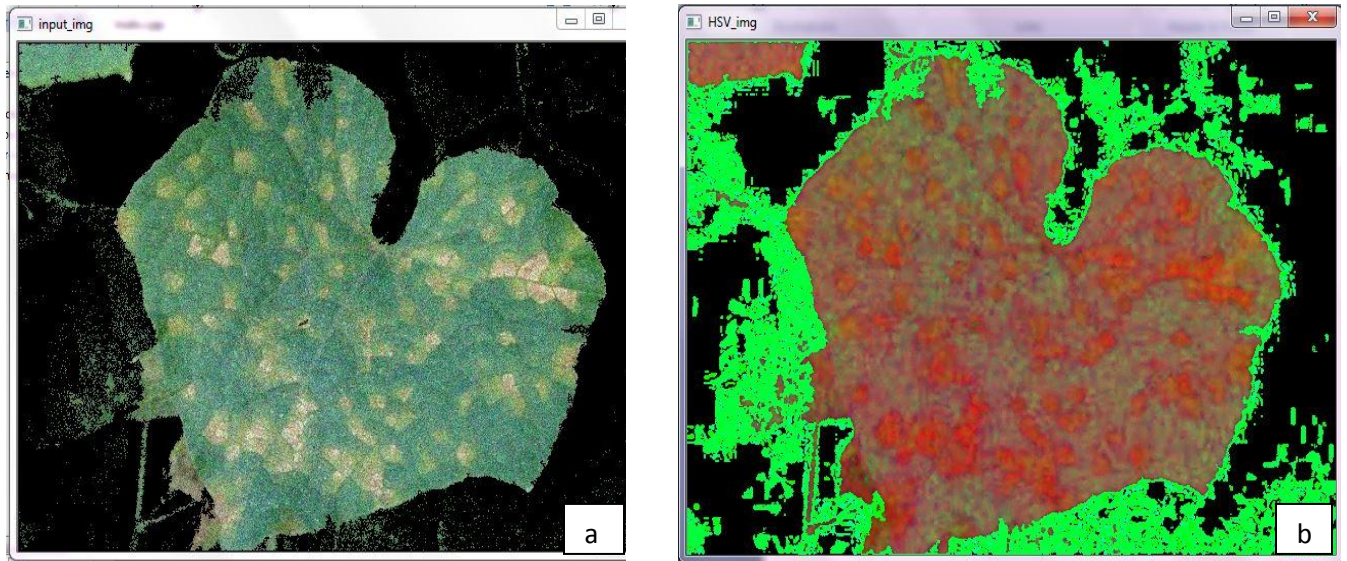


Figure 5.8 a: Image in RGB color space, b: Image in HSV color space.

Testing the process of extracting the infected regions (Figure 5.9 Illustrates that the process of extracting the infected regions is successfully done).

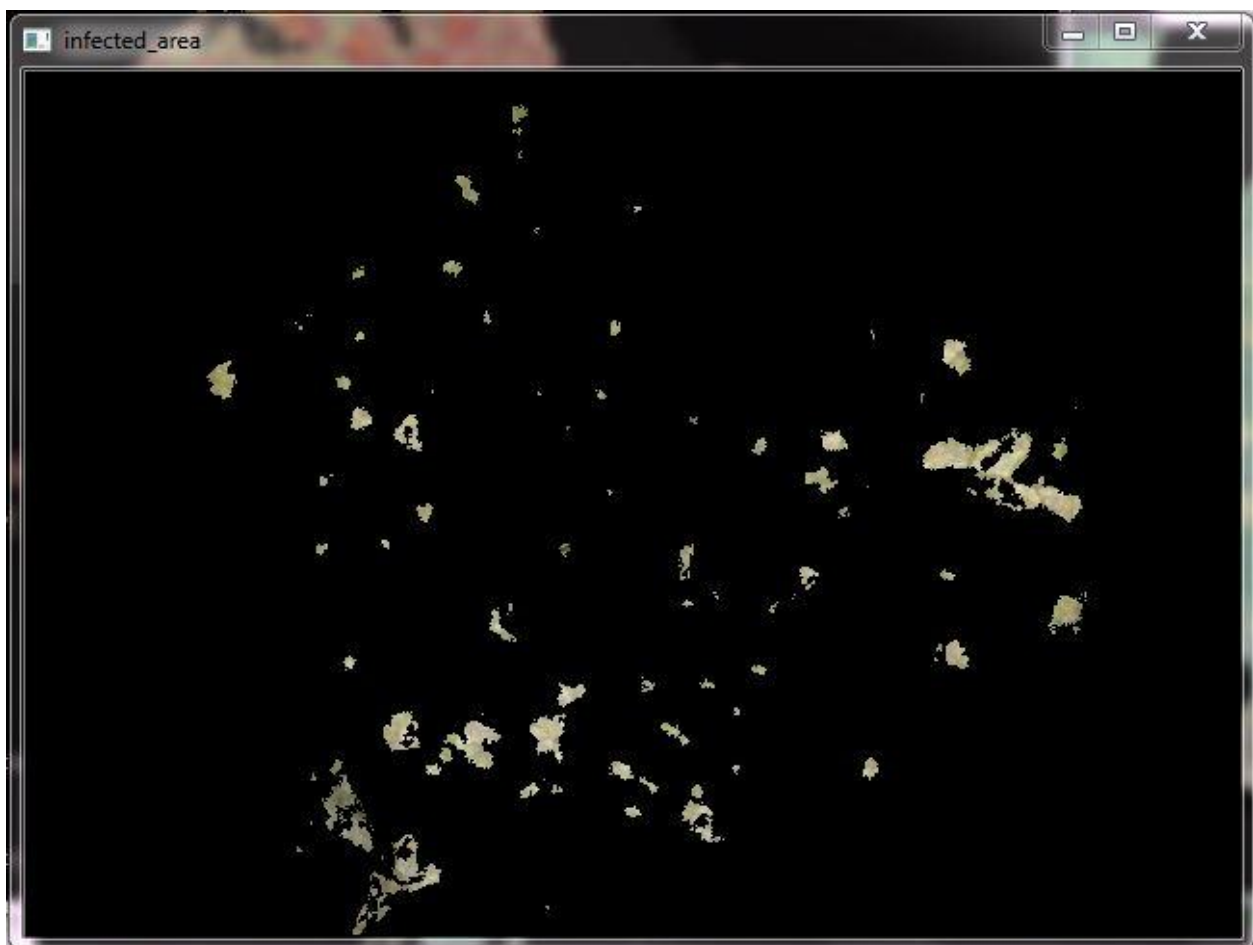


Figure 5.9 Infected regions of a leaf.

Testing the process of labeling connected components (Figure 5.10 illustrates that the process of labeling connected components is successfully done).

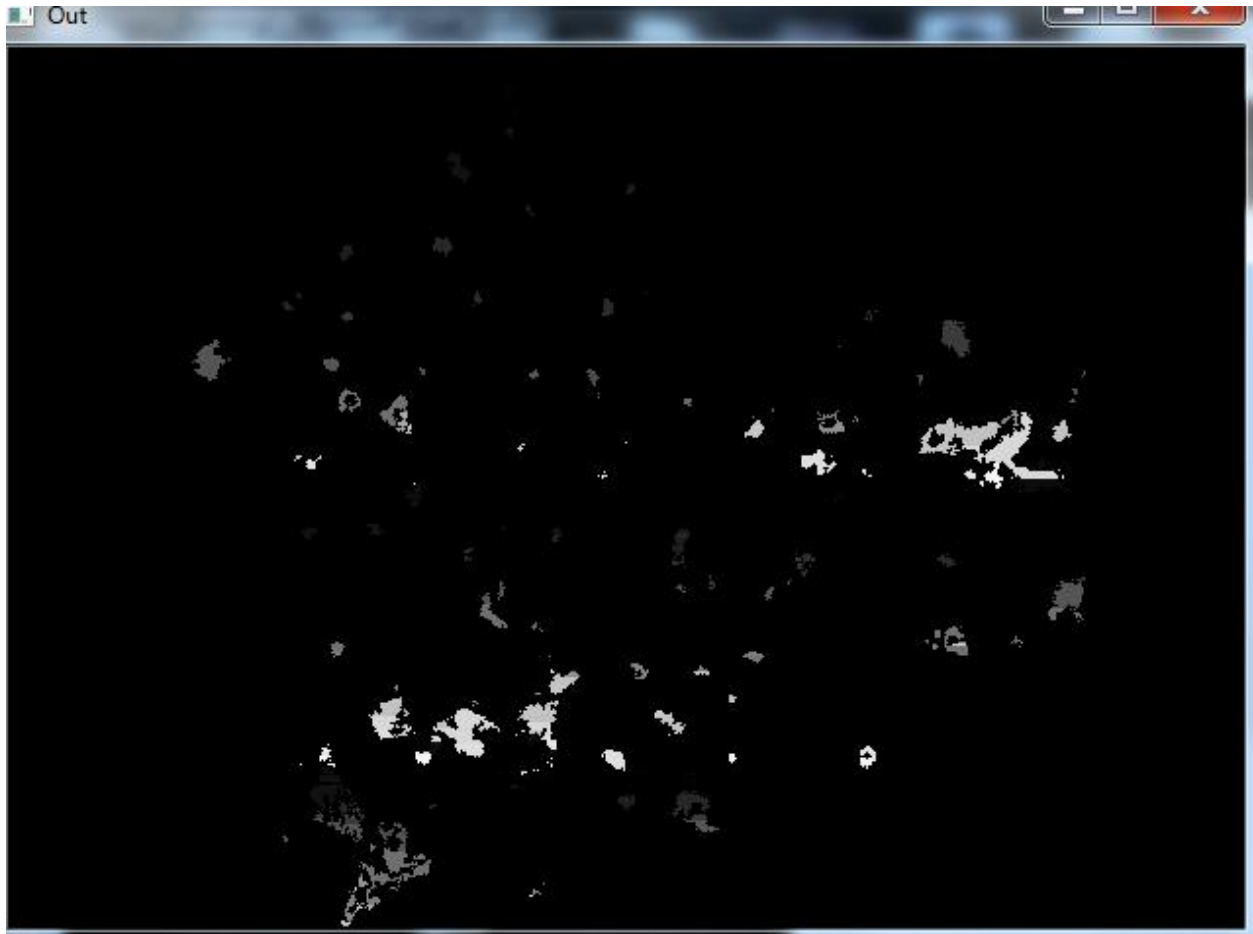


Figure 5.10 connected component labeling.

- **testing the whole system:**

After testing each of system modules separately, we tested the whole system, where all of the system modules are integrated together. Testing process is done through testing the validity of the results. In this approach, the system is tested on 100 samples from which 50 samples are infected with downy mildew disease, 50 samples are not infected. The results show that we were able to detect 50 infected images as infected i.e. 100% and 45 of the not infected images as not infected i.e. 90%.

Chapter six

Conclusion

Contents:

- 6.1 Results
- 6.2 Recommendations
- 6.3 Future Developments
- 6.4 Conclusion
- 6.5 References

Overview

This chapter will illustrate all results that are achieved after finishing development phase, recommendations and references that were used during the development phase.

6.1 Results

After finishing development phase the following results has been achieved:

- The team was able to develop the system, where images were converted to readable data to extract results from it.
- Decreasing time and increasing accuracy in plants experiments.
- Increasing the accuracy of results if a camera with high resolution were used in capturing images.

6.2 Recommendations

- Using a camera with a very high resolution.
- Adjust lighting properly.

6.3 Future developments

- Using sensors to check external effects and conditions such as humidity and temperature, in order to predict the occurrence of diseases before occurring.

6.4 Conclusion

A computer based method that can read a captured image of a plant leaf and give a statement and numbers that describes infection of this leaf from the image with minimum user intervention is innovated. The project can analyze a cucumber leaf then decide whether it is infected or not in addition to the degree of disease severity.

6.5 References

1. Athitsos, Vassilis. (2012). Nearest-neighbor retrieval and classification .
URL <http://vlm1.uta.edu>.
2. Averre, Charles. (2009). powdery mildew . URL
<http://www.insectimages.org/browse/detail.cfm?imgnum=1563076>.
3. Becker, Ron, & Miller, Sally A. (2009). Managing Downy Mildew in Organic And Conventional Vine Crops. The Ohio State University extension, 3127, 1-2.
4. Bradski, Gary, & Kaehler, Adrian. (2008). Learning OpenCV. USA: O'Reilly Media, Inc.
5. C. Gonzales and E. Woods, Digital Image Processing, Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2004.
6. Gonzalez, Rafael C., & Woods, Richard E.(2007) . Digital Image Processing.3rd Edition.New Jersey, USA: Prentice Hall.
7. COOKE, B.M and JONES, D. GARETH et al. (2006). The Epidemiology of Plant Diseases. 2nd Edition. Dordrecht, The Netherlands: Springer.
8. Holmes, Gerald. (2008). downy mildew . URL
<http://www.forestryimages.org/browse/detail.cfm?imgnum=5377181>
9. Jain, Ramesh, & Kasturi, Rangachar et al. (1995). Machine Vision. New York, USA: McGraw-Hill.
10. Lai, Chien-Yin et al.(2010). Computational models and experimental investigations of effects of balance and symmetry on the aesthetics of text-overlaid images.International Journal of Human-Computer Studies, 68.
11. Leung, K. Ming. (2007). Naive Bayesian Classifier. POLYTECHNIC UNIVERSITY Department of Computer Science / Finance and Risk Engineering.
12. Mariusz Jankowski. (2009). Connected components labeling - algorithms in Mathematica, Java, C++ and C#. USA: University of Southern Maine.

13. Németh, Szilárd. (2013). Image Processing with MATLAB – Hands-on workshop. URL http://gamaxlabsol.com/imageprocessing_ljubljana/
14. Olsen, Mary W. (2011). Plant disease management. Arizona Cooperative Extension, 1033, 1.
15. Regent Instruments Inc. (2012). WinFOLIA A Versatile Leaf Area Meter. Retrieved from <http://www.regent.qc.ca/products/fofia/WinFOLIA>.
16. Riley, Melissa B and Williamson, Margaret R et al. (2002). Plant Disease Diagnosis. URL <http://www.apsnet.org/edcenter/intropp/topics/Pages/PlantDiseaseDiagnosis.aspx>
17. Sankaran, Sindhuja, & Mishra, Ashish et al. (2010). Computers and Electronics in Agriculture. ELSEVIER, 72, 1-13.
18. Schroper, Florian, & Schillberg Stefan.(2011). Magnetic Immunodetection for Mobile and Rapid Identification of Plant Pathogens. Fraunhofer-institute for Molecular Biology and Applied Ecology.
19. Schouten, Theo. (2003). Color models. URL <http://www.cs.ru.nl/~ths/rt2/col/h2/2fundENG.html>.
20. Turunen , Tuukka .(2013). Qt |The Power of a Complete Development Framework. URL <http://qt.digia.com/Product/>.
21. Vick, Paul. (2005). The Microsoft® Visual Basic® Language Specification. Version 8.0. Microsoft Corporation.

APNDEX

Some of the important codes used in developing the system:

1. Camera capture code

```
void newtest::captureCam()
{
    if(camStarted)
    {
        cvReleaseCapture(&camera);
        ui->TakeShotButton->setEnabled(false);
        ui->SaveShotButton->setEnabled(false);
        ui->startCambutton->setText(tr("Start"));
        killTimer(cameraTimerId);
        camStarted = false;
    }
    else
    {
        camera = cvCreateCameraCapture(0);
        cvSetCaptureProperty( camera, CV_CAP_PROP_FRAME_WIDTH, 320);
        cvSetCaptureProperty( camera, CV_CAP_PROP_FRAME_HEIGHT, 240);

        IplImage * image = cvQueryFrame(camera);

        if(!camera || !image)
        {
            QMessageBox::critical(this, tr("Error"), tr("can't open cam"), QMessageBox::Ok);
            return;
        }

        ui->TakeShotButton->setEnabled(true);
        ui->SaveShotButton->setEnabled(true);
        ui->startCambutton->setText(tr("Stop"));
        cameraTimerId = startTimer(20);
        camStarted = true;
    }
}
```

2. Connected components labeling

```
Public Class ConnectedComponenet
Public Sub New()
End Sub
Public Function count_connected_componenet(ByVal img As Bitmap) As Integer
Dim i As Integer
Dim j As Integer
Dim c(img.Width, img.Height) As Integer
Dim lable As Integer = 1
Dim eq(2, 200000) As Integer
Dim count As Integer = 0
Dim temp1 As Integer
Dim temp2 As Integer
Dim tf As Boolean
Dim white As Color
Dim black As Color
Dim n As Integer = 0
white = Color.White
black = Color.Black
Dim o As Bitmap
o = img
Dim hi As Integer = img.Height
Dim hj As Integer = img.Width
For j = 0 To img.Height - 1
For i = 0 To img.Width - 1
If img.GetPixel(i, j).R <> 0 Then '-----> if the the pixle is black
c(i, j) = 0
Else
If i = 0 And j = 0 Then '-----> if it the top left corner
c(i, j) = lable
lable += 1
ElseIf i = 0 Then '-----> if it the top line
If img.GetPixel(i, j - 1).R <> 255 Then
c(i, j) = c(i, j - 1)
Else
c(i, j) = lable
lable += 1
End If
ElseIf j = 0 Then
```



```
int N,id;
N=label;

int* arr = new int [N]; '-----> create a dynamic array
For j = 0 To img.Height - 1
For i = 0 To img.Width - 1
Dim id =img.GetPixel(i, j).R
If (id >0)
arr[id]++;
cImg = cvCreateImage(cvSize(width, height), IPL_DEPTH_8U, 1)
For j = 0 To img.Height - 1
For i = 0 To img.Width - 1
id= id =img.GetPixel(i, j).R
if (arr[id]==0)
PIXEL(cImg,i,j)=0
else
PIXEL(cImg,i,j)=255
delete [] arr
```

3. Segmentation code

```
void hsvSeg(const cv::Mat& input_image, const cv::Mat& hsv, cv::Mat
&output1, cv::Mat &output2, int low_h, int low_s, int low_v, int
high_h, int high_s, int high_v)
{
    hsv.copyTo(output1);
    Mat bw;
    inRange(output1, Scalar(low_h, low_s, low_v), Scalar(high_h,
high_s, high_v), bw);
    vector<vector<Point>> contours;
    findContours(bw.clone(), contours, CV_RETR_EXTERNAL,
CV_CHAIN_APPROX_SIMPLE);
    Mat dst = Mat::zeros(input_image.size(), input_image.type());
    drawContours(dst, contours, -1, Scalar::all(255), CV_FILLED);
    dst.copyTo(output2);
    output2 &= input_image ;
}
```

