Palestine Polytechnic University

College of Administrative Sciences & Informatics
Information Technology Department

Features Extraction of Plant Compound Leaf

Project Team

Asma Ishaq Idais

Duaa Wail Abu Maizer

Haneen Msalam Tartory

Project Supervisors

Dr. Hashem Tamimi & Dr. Rami Arafeh

This project is presented to the Department of Information Technology at College of Administrative Sciences & Informatics, for partial fulfillment Bachelor of Information Technology degree requirements.

May, 2010

Abstract

Extraction features from plant leaf represents a challenging problem in image processing. Many researchers extracted features from plant leaf to help in plant classification and also in early diagnosis of certain plant diseases. Many previous researches emphasized on extracting features from simple leaf using image processing techniques.

This project aims at proposing a methodology to extract features from a compound leaf using image processing, which is used for clustering plants that have compound leaves into similarity groups; this will help to know the plants that have similarity features. The experimental results show that the proposed methodology of extracting features from a compound leaf can be used to cluster the plant compound leaf into similarity groups using hierarchical clustering method.

Dedication

To our parents for their support

To our brothers and sisters

To our teachers for their advices

To our friends

Asma Idais

Duaa Abu Maizer

Haneen Tartory

Acknowledgement

As we write the last words of this project we greatly appreciate the project's supervisors Dr. Hashem Tamimi & Dr. Rami Arafeh for their support and time they spent with us in order for this project to succeed.

Special thanks to Dr. Mohammed Aldasht the Head of Information Technology Department.

Special thanks to Dr. Yaqoub Ashhab the Head of the Biotechnology Training and Research Unit in Palestine Polytechnic University.

Special thanks to Eng.Hani Salah for his interest and support.
Special thanks to Eng. Suzan Sultan for her support.

We thank our families for continued support, encouragement and patience from the first step till the end.

Asma Idais
Duaa Abu Maizer
Haneen Tartory

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter One

Introduction

Contents:

1.1 Overview

Plants represent an important component in human life; they are source of food, source of energy, used in industry and as construction material and also used for their medicinal values. Recent advances in computer technology open new avenues in many life sciences including plant biology (Botany). Computer recognition of plant types and discrimination between plants by computer feature extraction are gaining more attention among scientists.

Many plants carry significant information for humans, features of plant can be extracted from different parts of the plant such as fruits, flowers, roots or plant leaves, and it could have many implications for example in plant taxonomy or in the early diagnosis of certain plant diseases.

Recently, some studies focused on extracting certain features from the plant depending on its leaves which is considered as an obvious feature using image processing techniques. Nevertheless, the previous works in literature showed that there are many works on a simple leaf of different plant species; on the other hand, few studies have focused on using a compound leaf, which is considered to be more complicated.

This project aims at proposing a methodology to extract features from compound leaves using image processing techniques. In this project, the main features of a compound leaf are proposed, and as a case study, tomato leaf is suggested because it appears in many types depending on the variety of fruit shape and size.

These extracted features are used in this project to identify points of matching among the leaves, and to cluster plants that have compound leaves into similarity groups based on features; this will help the interested researchers in the plant field to know the similarity plants that have the similar features, also the clustering can be used to reduce the size of large numbers of plant's types to help in classification. These two methods also help to verify the importance of features extracted in this project.

The proposed methodology uses digital images of a compound leaf, then using image processing to pre-process the images of the leaf to isolate the main leaf from other parts of leaf, and then to extract morphological features from plant leaf image. After that the hierarchal clustering is used to do the clustering process.

The expected results of this project will be a methodology that can read digital image of compound leaf and give a statement and numbers that describe this leaf in a good way with minimum intervention from user, and the project can read a sample of plant compound leaf and cluster them into similarity groups depending on the proposed features using hierarchical clustering.

1.2 Project objectives

The project aims at achieving two objectives:

- The main objective of this project is developing a methodology that can extract features from plant compound leaf, using image processing techniques.
- The other objective of this project is finding similarity in groups of plants that have compound leaves.

1.3 Project block diagram

The main steps of this project are illustrated in Figure 1.1.

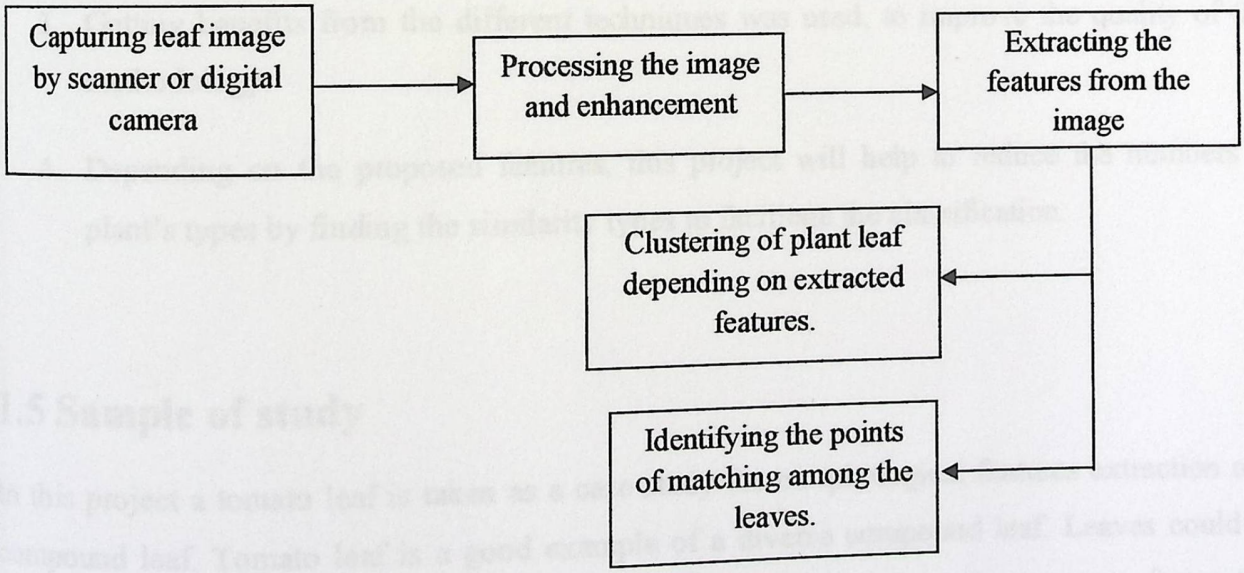


Figure 1.1 Block diagram of plant leaf recognition

1.4 Project benefits

Before mentioning the benefits of this project, a summary about some drawbacks in previous works that related to this project is presented, some of these drawbacks including:

1. Most previous works focused on extracting morphological features from simple leaf.
2. In previous works, researchers depended on complicated computations and analysis to describe the morphological features of plant leaf.
3. In many previous researches, the process of extracting the features depended on high user intervention.

This project aims at overcoming these drawbacks by developing a methodology contains many benefits including:

1. Developing a method that can extract morphological features from a compound leaf, which is considered more difficult than simple leaf.
2. Making the process of extracting the features automatically with minimum user intervention.
3. Getting benefits from the different techniques was used, to improve the quality of this methodology.
4. Depending on the proposed features, this project will help to reduce the numbers of plant's types by finding the similarity types to facilitate the classification.

1.5 Sample of study

In this project a tomato leaf is taken as a case study for morphological features extraction of a compound leaf. Tomato leaf is a good example of a diverse compound leaf. Leaves could be obtained either from leaf scanned images, from internet sources, or from tomato farms that representing varieties grown in Palestine.

1.6 Expected results

The expected results of this project are:

1. A computer based method that can read a scanned image of a compound leaf and give a statements and numbers that describes this leaf from the image with minimum intervention of user
2. The project can read a sample of plant compound leaves and cluster them into similarity groups depending on the proposed features

Chapter Two

Background

Contents:

2.1 Overview

Plant leaf carries significant information for human, such information could be used to classify plants or to the early diagnosis of certain plant diseases. Certain features of plant leaf should be extracted and identified in order to end up with useful information.

To extract these features, they should be identified carefully in order to know their variations and the important features that should be studied. In order that in this chapter a good background of plant leaf features will be introduced under the section morphological features of plant leaf.

In order to extract the morphological features of plant leaf there are some techniques in image processing will be implemented. In the second section of this chapter the image processing techniques are needed in this project will be explained. Some of them are: converting from RGB color to binary color, morphological operation on binary image such as erosion, dilation, opening, and closing, and connected component technique will be explained.

Final section of this chapter will cover the cluster analysis concept, algorithms and hierarchical clustering methods.

2.2 Features of the plant leaf

This section introduces features as a general, the general morphological features of plant leaf, the difference between simple and compound leaf, and the main characteristics of tomato leaf.

2.2.1 What are features?

Features are defined as points represent images in multidimensional feature space which are used to compute the similarity between the images. The images are similar to each other when the images are close to each other in high dimensional space.
Feature extraction can be defined as the process of mapping image from its original space. Some of the methods to extract features are better than other methods in accuracy and time of computation process (Tamimi, 2006, p.22) [1].

Some approaches of features extraction can produce the following features (Tamimi, 2006, p.23) [1]:

1. Uniqueness: images are different to awareness of human have the different features.
2. Invariance: images are similar to awareness of human have the similar features.
3. Stability: the degree of similarity between two images that leads to small degree of similarity in matching features.
4. Efficiency: the feature should be with small size without losing characteristics.
5. Ease of implementation: the feature extraction should be with efficient consecutively.

Local and Global Features

There are two ways for extracting features from an image (Tamimi, 2006, p.24) [1]:

1. Global Feature: one feature is extracted from whole image.
   Advantages are compact representation of an image and standard classification algorithms and the implementation for detecting and extracting less complicated than local features.
   Disadvantages are sensitive to occlusion and require segmentation.
   Examples of global features are histogram and Principal Component Analysis (PCA).

2. Local Feature: Set of features is extracted from the image that describes localized image regions and these regions called patches. Descriptors are a set of features that describe small image region and they are computed around interest points. Local features do not lend themselves easily to standard classification techniques.
   Advantages occlusions are no need for segmentation, robust to occlusion.
   Disadvantage is the images are represented by different size sets of feature vector.
   Examples of local features are Scale Invariant Feature Transform (SIFT) and (PCA) based local features.

2.2.2 General morphological features of the plant leaf

In biology, the study of the size, shape, and structure of organisms in relation to some principle or generalization [2], is called morphology.

When the plant studied using computer technology emphasis should follow on morphological features of plant leaf, these morphological features can be used to classify plant and to study the plant leaf and interaction with environment.

Before talking about the morphological features, the main parts of leaf should be explained to make features obviously.

The main parts for a leaf are: leaf blade (lamina), stipules and petiole. The blade is the flat part of the leaf, forms the green portion of the leaf. The petiole is the leaf stalk. Some leaves do not have petioles so the base of the leaf is attached to the stem. Stipules are leaf like parts at the base of the leaf [2]. Other parts of leaf are axil which is an angle between petiole and the stem, and axillary bud which is a bud present in the axil (Figure 2.1).

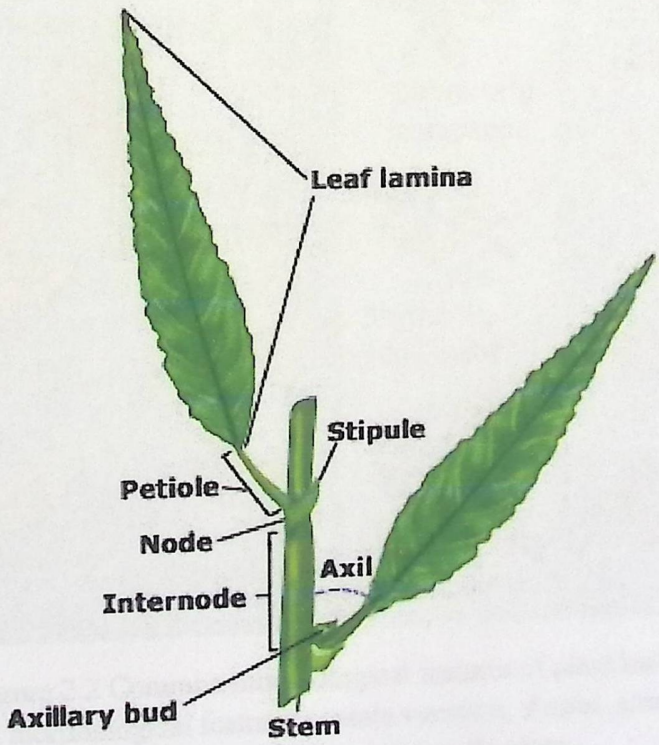


Figure 2.1 Parts of leaf [3]
Where the main parts of leaf are: leaf blade, stipules and petiole.

General morphological features in plant leaf that can be studied are [3]:

- Venation which is the arrangement of veins in a leaf, such as pinnate, parallel and palmate.
- Shapes of leaf, such as linear, ovate, reniform and obovate.
- Arrangement of leaf maybe simple, or palmately compound, or pinnately compound or bipinnately compound.
- Margin of leaf which is the edge of a leaf, such as entire, dentate.
- Arrangement on the stem is how the stem of plant arranges, such as alternate, opposite, whorled. (See Figure 2.2 for more illustration).

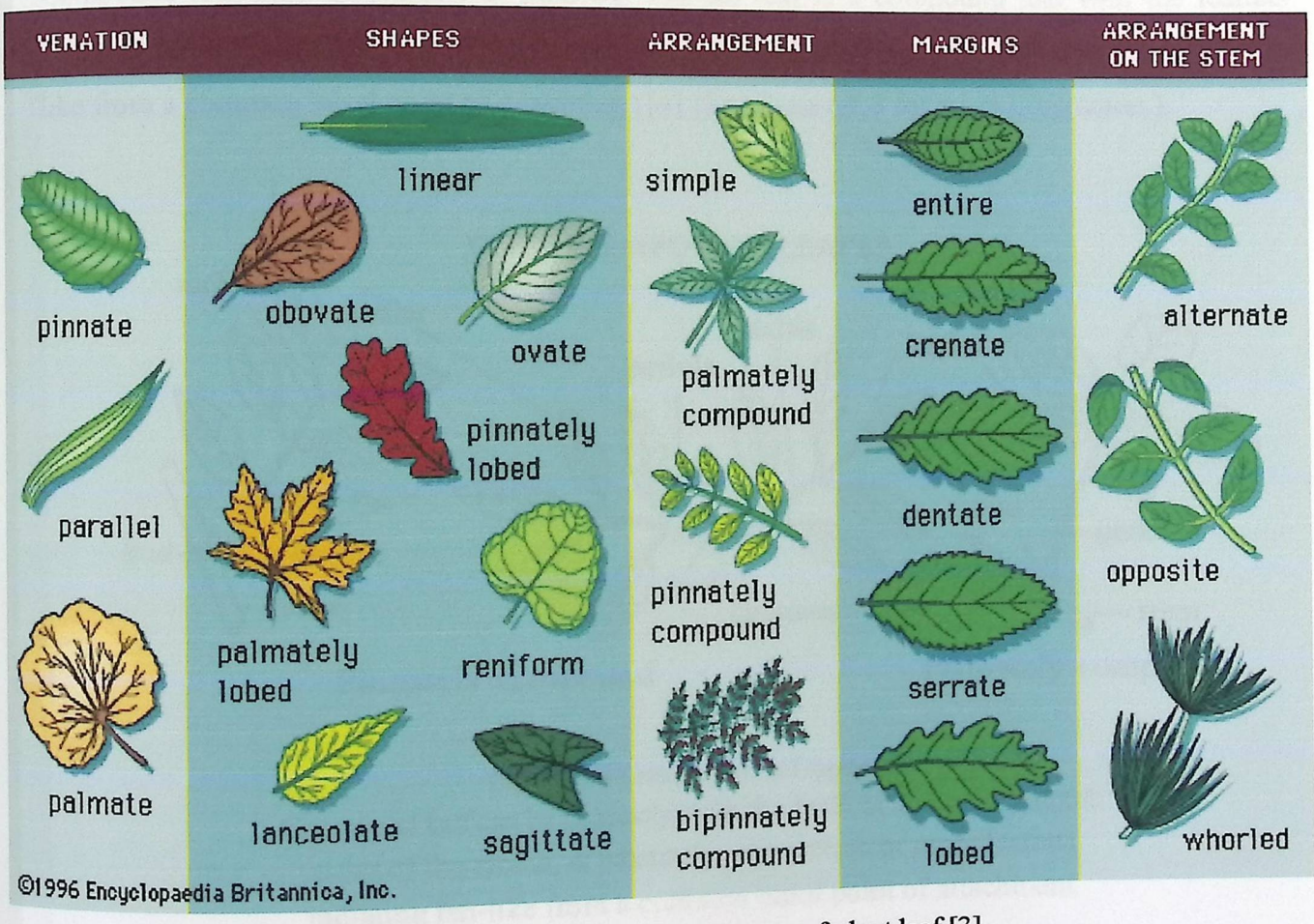


Figure 2.2 Common morphological features of plant leaf [3]
General morphological features contain venation, shapes, arrangement, margins and arrangement on the stem.

2.2.3 Simple and Compound leaf

Because this project focuses on extracting features from a compound leaf rather than simple leaf, the differences between these two types should be explained.

Plants have two main types of leaves; simple and compound leaf. A simple leaf is a leaf in which the blade is appears like one unit. On the other hand, compound leaf is one leaf that has many blades and it appears like many simple leaves called leaflets.

To distinguish between simple and compound leaves one should look for the band on axil. A leaf that is connected to band on axil is a simple leaf, whereas a leaflet that has no bud at its base is part of a compound leaf (Figure 2.3).

Compound leaf may be pinnate or palmate. Pinnate leaf is a compound leaf with the leaflets arranged on both sides of the rachis, whereas palmate leaf is three or more leaflets radiating fan-like from a common basal point of attachment [4] (See Figure 2.3 for more illustration.)

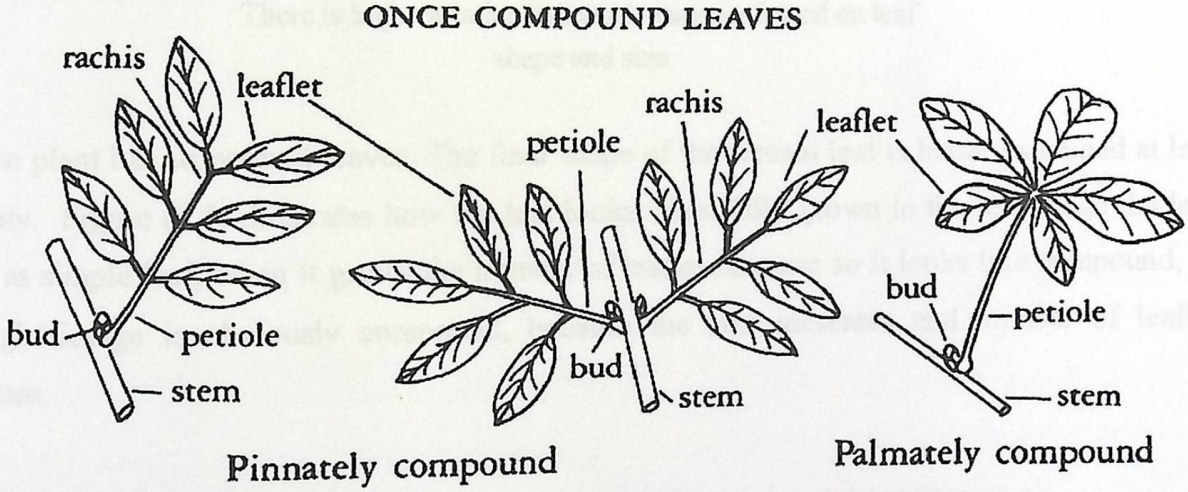


Figure 2.3 Compound leaf types [5]
Compound leaf can be pinnately; with the leaflets arranged on both sides of the rachis, or palmately; where three or more leaflets radiating fan-like from a common basal point of attachment.

2.2.4 Characteristics of tomato leaf

Tomato is one of the most common vegetables spread in the world. One of the most important nature characteristic of tomato is the variation. There are more than 5000 tomato varieties that differ in fruit shape and size [6]. These variations are also reflected on size and shape of the tomato leaf. Figure 2.4, illustrates some variation of tomato leaf.

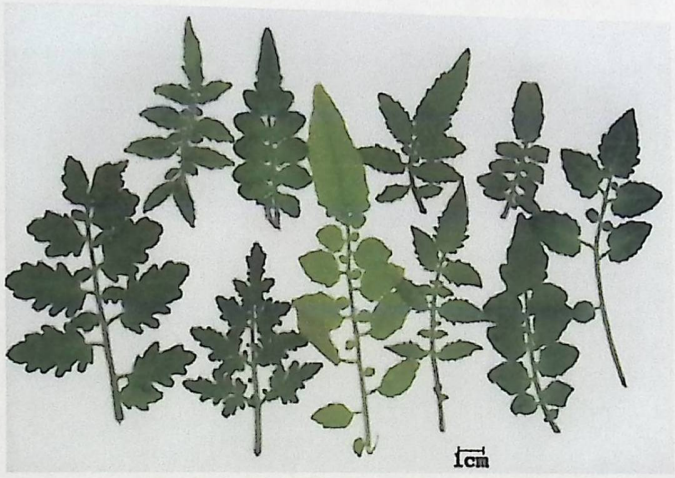


Figure 2.4 Tomato leaf variations [7]
There is high variation in tomato varieties based on leaf shape and size.

Tomato plant has compound leaves. The final shape of the tomato leaf is better examined at leaf maturity. Figure (2.5) illustrates how the leaf looks when fully grown in the left image the leaf looks as simple leaf, when it grows the number of leaflet increase so it looks like compound, in the right image is obviously compound, because the size increases and number of leaflet increases.

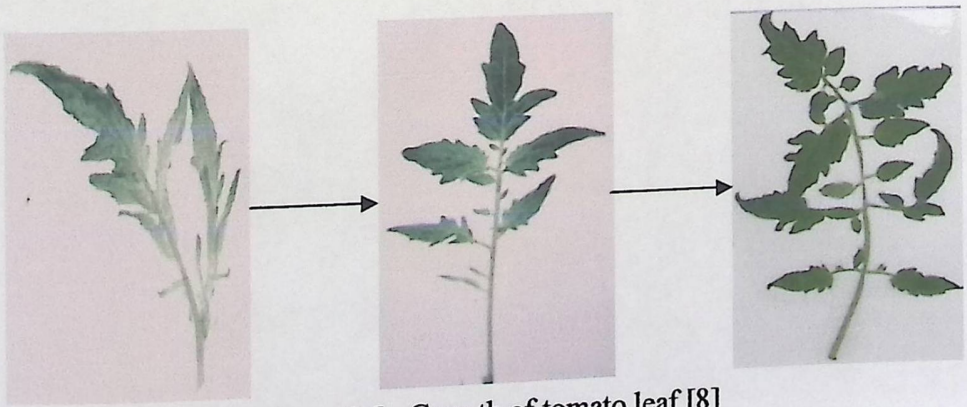


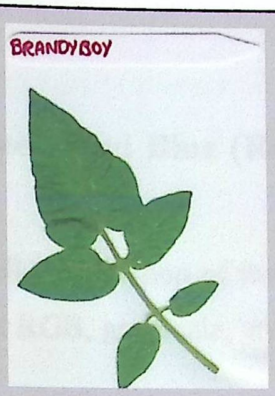
Figure 2.5 Growth of tomato leaf [8]
In the left image the leaf looks as simple leaf, when it grows the number of leaflet increase so it looks like compound, in the right image is obviously compound.

There are many types of tomato leaflets, but there are two main types: regular leaf (RL) type and potato leaf (PL) type. Table 2.1 contains the differences between these two types.

Table 2.1 Types of tomato leaflets
This table contains the differences between the PL type and RL type, and examples of these two types

Tomato leaf types / Differences	RL Type	PL Type
Definition	Is the typical leaf type that most people are familiar with. It contains number of leaflet, each has a toothed and asymmetrical edge, this type is differs of shape, length, color depending in many things such as claim [10].	These leaves usually have few lobes on their leaves, smooth edge, and they are thicker than regular type, so it is more resistance to diseases. The color is usually a deep green [10].
Example of leaflet shape	 Example of regular leaflet [6]	 Example of potato leaflet [6]
Examples of tomato leaf types	 Cherokee Purple [9]	 Pruden Purple [9]

Lemon Boy [9]



Brandy Boy [9]

2.3 Image processing techniques for feature extraction

Image processing can be defined as analyzing and manipulating images with a computer, or it is a technique in which the elements of the image are digitized and different mathematical operations are applied to it. This is done in order to create 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 [11]. Image processing is used in many areas, including astronomy, medicine, industrial robotics and other fields.

Image processing generally involves three steps:

1. Capturing an image with a camera, optical scanner or through other sensors.
2. Analyzing and manipulating the image: this step may include image enhancement, or the image analysis to extract some important features. This process is done using various methods.
3. The output may be a new image or a report from the analysis of the image.

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

These techniques are:

2.3.1 Converting the color of the image from Red, Green, and Blue (RGB) to binary

Before talking about how to convert the color of the image, a small introduction of three color models needed in this project will be introduced. These models are RGB, grayscale, and binary model.

RGB color model: RGB color model is an additive model in which red, green, and blue are collective in various ways to reproduce other colors [12]. All the colors can lie in a cube extending from the origin (black) as illustrates in Figure 2.6.

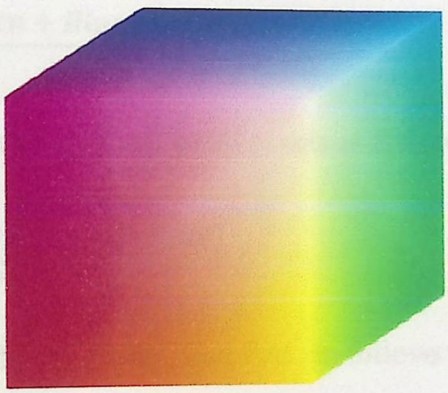


Figure 2.6 RGB color model [13]
RGB color model contains three primary colors; red, green, blue.

Grayscale color model: grayscale is a model for encoding the colors of an image which contains only black, white and shades of gray [14] The grayscale image contains certain levels of gray, from 0 (black) to 1 (white), for example the levels may be 256 different shades as illustrates in Figure 2.7. Grayscale is easier to process than RGB model.

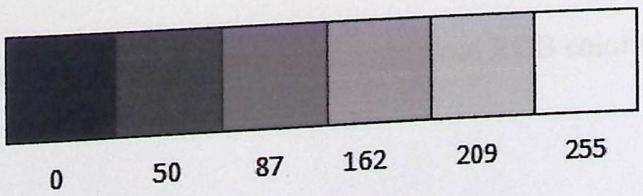


Figure 2.7 Grayscale Color model.
Grayscale image contains 256 levels of gray, from 0 (black) to 255 (white).

Binary image: This image contains two values, usually denoted 0 to represent black color, and 1 to represent white color in the image. The binary image is used in some applications such as [15]: identifying objects on a conveyor, identifying orientations of objects, and interpreting text.

In order to convert images from RGB to binary, we first convert RGB to gray then convert gray to binary as follows:

1. Converting RGB image to grayscale image:

   When converting RGB color of image to grayscale, all the colors are replaced with shades of gray. There are a number of techniques for converting RGB images into grayscale [16]:

   A simple method is to average the colors for all pixels as follows:

$$Y = \frac{Red + Green + Blue}{3} \qquad (2-1)$$

   Or a weighted average in applied for all pixels as follows:

$$Y = \frac{(3 * Red + 4 * Green + 2 * Blue)}{9} \qquad (2-2)$$

   Another common method is NTSC and PAL as follows:

$$Y = 0.3 * Red + 0.59 * Green + 0.11 * Blue \qquad (2-3)$$

   Where, PAL, short for Phase Alternate Line, is an analogue television encoding system used in broadcast television systems in large parts of the world.

   In this project we will use the NTSC and PAL because it is the most commonly used in the literature, and NTSC, named for the National Television System Committee, is the analog television system used in most of North America, most countries in South America, Burma, South Korea, Taiwan, Japan, Philippines, and some Pacific island nations and territories.

   Figure 2.8 shows the results of converting process from RGB color to grayscale

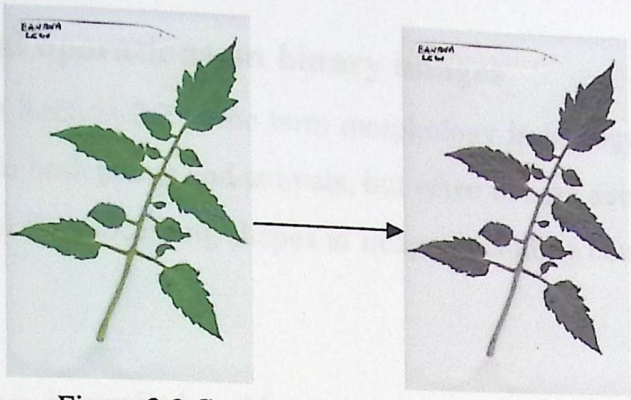


Figure 2.8 Converting from RGB to grayscale.
In the left, the image is in RGB model, and it converted to grayscale in the right image.

2.Converting Grayscale to Binary:

As mentioned before, binary images have only two possible intensity values, to represent black and white. The two possible values are often 0 for black and either 1 for white.

To convert the grayscale images into binary images, a given threshold must be used in order to separate an object in the image from the background. The color of the object referred to as the foreground color, the reset is referred to as the background color. Given a threshold T, we can convert a gray scale image to a binary image by setting all gray values equal or above T to white and all values below T to black .As follows:

Algorithm 2.1

{If gray value> T Then
Binary value = 1 (white)
Else
Binary value = 0 (black)
}

Figure 2.9 shows the results of converting process from grayscale to binary.

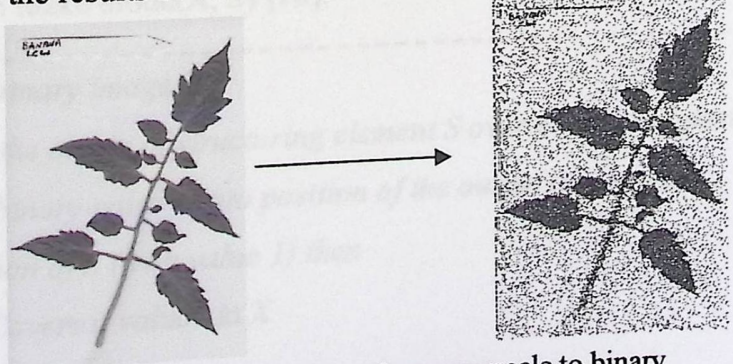


Figure 2.9 Converting from grayscale to binary.
In the left, the image is in grayscale model, and it converted to binary in the right image.

2.3.2 Morphological operations on binary images

As mentioned before in Section 2.2.2, the term morphology in biology refers to the study of the form and the structure in both plants and animals, but when talking about image processing refers to the mathematical tool for processing shapes in image, including boundaries, skeletons, convex hulls, etc.

Morphological operation is applied to binary images, for shape and structural manipulation. Morphological operations are used in this project to extract primitive features of an object (leaf) that can be used to recognize or classify the object.

In a morphological operation an input image is operated using a structuring element to produce an output image of the same size. The value of each pixel in the output image is based on a relation between of the corresponding pixel in the input and the structuring element [17] The structuring element is a small binary image with fix size and content that depends on the purpose morphological operation.

There are two fundamental morphological operations; dilation and erosion:

- **Erosion:** this operation allows an object in the binary image to shrink by removing pixels in the boundaries of the object, so the object becomes smaller. With binary images, erosion completely removes objects smaller than the structuring element and removes perimeter pixels from larger image objects [18].
  The arguments to erosion are a binary image(X) and a structuring element(S) which can be defined as a shape mask is used in the basic morphological operations, and can be of any shape, and size, and each has an origin [18]. The Algorithm 2.2 illustrates the erosion operation to make erode(X, S) [18]:

Algorithm 2.2

{Takes a binary image X
Places the origin of structuring element S over every pixel position
Puts a binary value 1 into position of the output image only
If position of S (with value 1) then
Covers a value1 in X
End if }

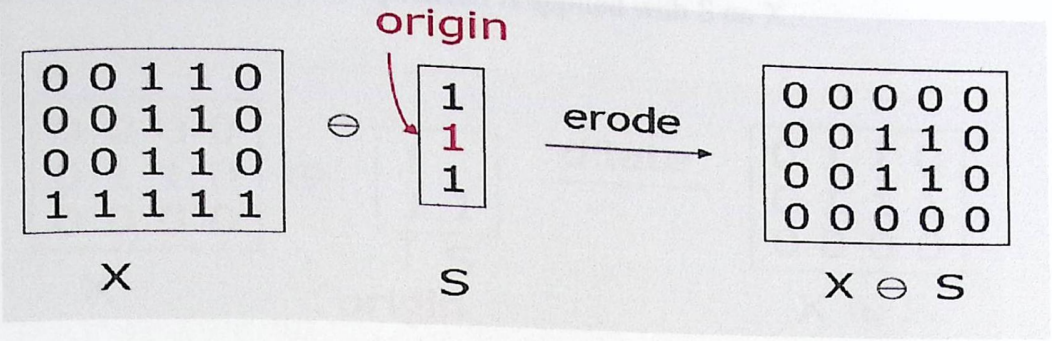


Figure 2.10 Erosion operations with structuring element [18] in the left, a binary image X, and structuring element S in the middle, and output image after erosion in the right

- **Dilation:** this operation allows an object in the binary image to grow, by filling small holes or connecting disjoint objects. Dilation operation is done by passing the structure element on the whole image so that pixels are added to the boundaries of the object in the image [18].

  With binary images, dilation connects areas that are separated by spaces smaller than the structuring element and adds pixels to the perimeter of each image object [18]. The Algorithm 2.3 illustrates the dilation operation to make dilate(X, S) [18]:

Algorithm 2.3

{ Take binary image X

Places the origin of structuring element S for each pixel of value 1

Apply Ors the structuring element S with X.

Output image is produces at the corresponding position

}.

In Figure2.11 illustrates dilation operation is applied with S on X.

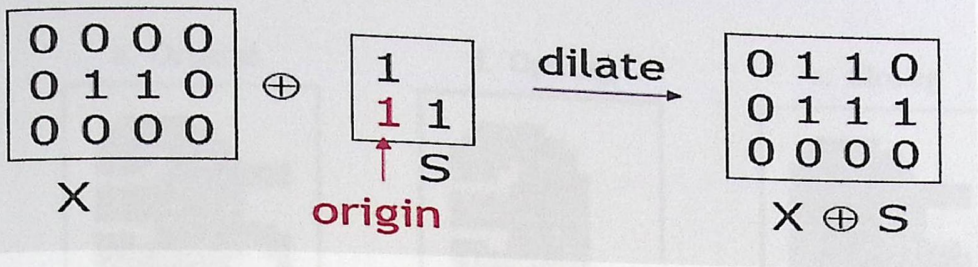


Figure 2.11 Dilation operations with structuring element [18]
in the left is a binary image X, and structuring element S in the middle, and output image after dilation in the right

Figure 2.12 represents the two types of morphological operations.

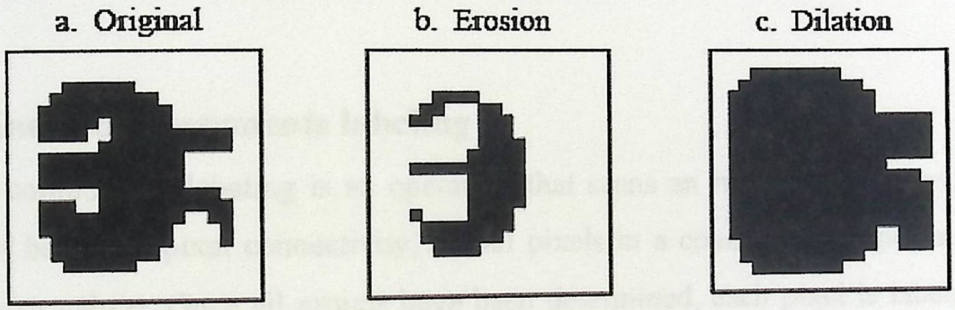


Figure 2.12 Basic morphological operations [19]
(a) is the original image, (b) is image after erosion and (c) is the image after dilation.

An important factor in morphological operation is the structure element, because the size of the structure element influences on the number of pixels added or removed, so we should be careful when we choose the size of the structure element.

Dilation and erosion can be combined to implement more complex image processing operations, such as the morphological opening of an image which is an erosion followed by dilation, using the same structuring element for both operations, and morphological closing of an image which is

the reverse, it consists of dilation followed by erosion with the same structuring element[17]. Figure 2.13 illustrates the two morphological operations.

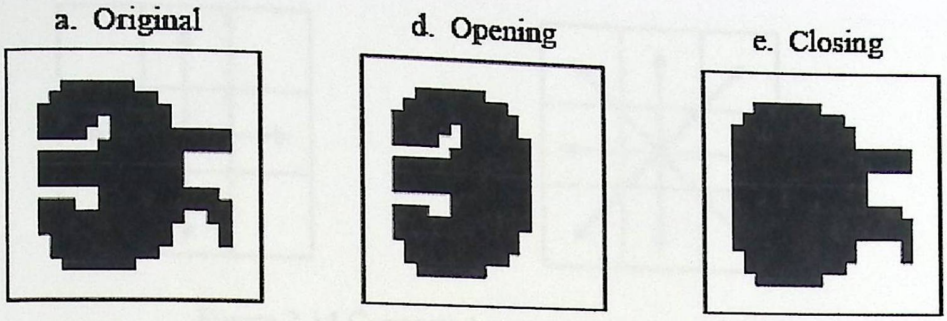


Figure 2.13 closing and opening morphological operations [19]
(a) is the original image, (d) is image after opening and (e) is the image after closing.

2.3.3 Connected components labeling

Connected components labeling is an operation that scans an image and groups its pixels into components based on pixel connectivity, *i.e.* all pixels in a connected component share similar pixel intensity values. Once all groups have been determined, each pixel is labeled with a gray level or a color (color labeling) according to the component it was assigned to [20].

Connected component labeling works by scanning an image, pixel-by-pixel from top to bottom and left to right to identify the regions of connected pixel, i.e. regions of adjacent pixels which share the same set of intensity values V. For a binary image V= 1; however, in a gray level image V can take a range of values, for example: V= {51, 52, 53, ..., 77, 78, 79, 80}[20].

Connected component labeling works on binary or gray images and different measures of connectivity are possible. Either 4 connectivity or 8 connectivity are possible as in Figure 2.14.

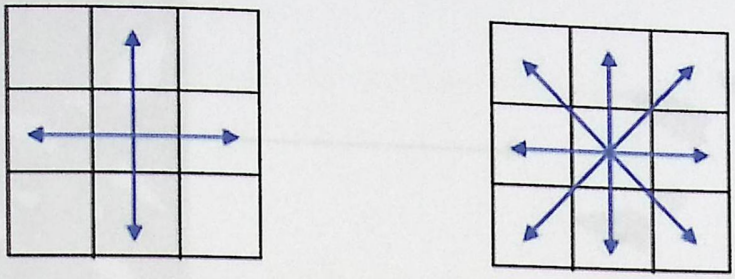


Figure 2.14 Connected components connectivity
In the lest is 4-neighbor metric and the right is 8-neighbor metric

The algorithm of connected component includes two passes over the image as follows:

Algorithm 2.4

{

Pass1:

Scan the image pixels from left to right and from top to bottom.

For every pixel P of value 1(an object pixel), test top and left neighbors (4-neighbor metric).

If 2 of the neighbors equal 0: assign a new mark of P, else

If 1 of the neighbors equals 1: assign the neighbor's mark to P, else

If 2 of the neighbors equal 1: assign the left neighbor's mark to P and note equivalence between 2 neighbor's marks.

Pass 2:

Divide all marks in to equivalence classes (marks of neighboring pixels are considered equivalent).

Replace each mark with the number of its equivalence class

}.

In Figure 2.15, example of how connected component labeling works.

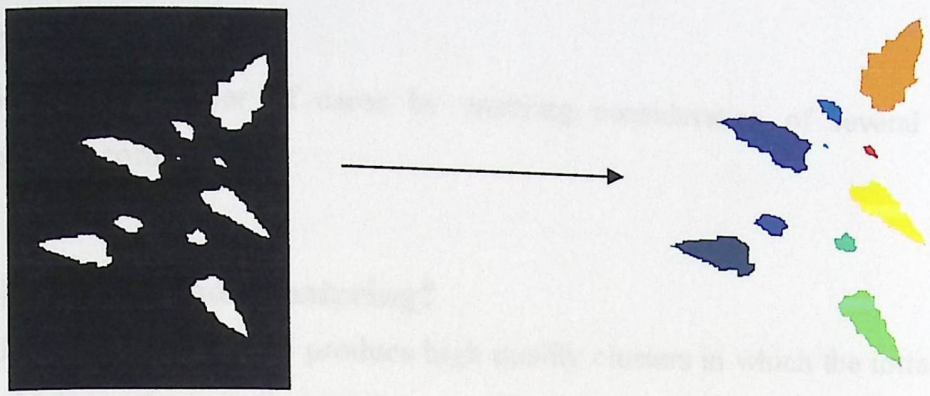


Figure 2.15 Connected Components labeling Example

2.4 Cluster Analysis

Cluster analysis is a collection of statistical methods, which identifies groups of samples that behave similarly or show similar characteristics, based on information found in the data that describes the object and their relationships [21]. Figure 2.16 shows the 4 clusters of data according to a given distance.

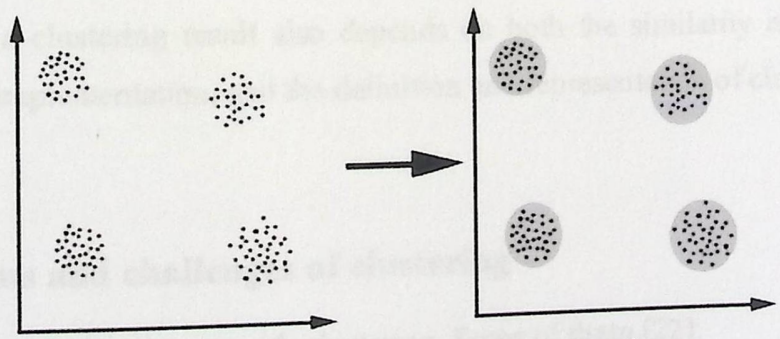


Figure 2.16 Example of cluster data based on distance [22]

The result of cluster analysis is a number of heterogeneous groups with homogeneous contents, where there are substantial differences between the groups, but the individuals within a single group are similar [21].

2.4.1 Advantages of cluster analysis

The main advantages of the cluster analysis are:

- Discovering types.
- Reducing the number of cases by enabling consideration of several types instead of numerous records.

2.4.2 What Is a Good Clustering?

A good clustering method will produce high quality clusters in which the intra-class similarity is high, and the inter-class similarity is low, see Figure 2.17.

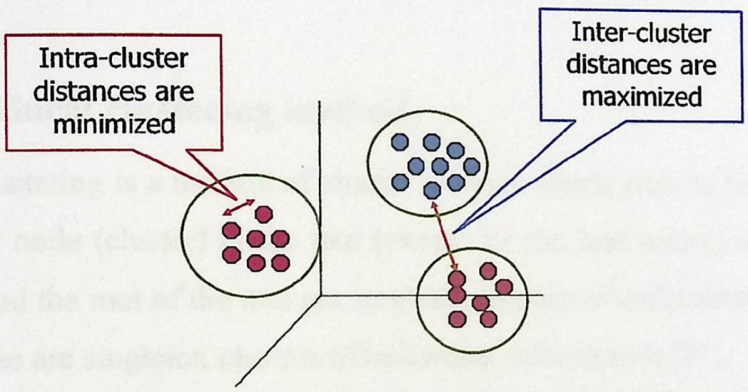


Figure 2.17 Distances of intra and inter classes [22]

The quality of a clustering result also depends on both the similarity measure is used by the method and its implementation, and the definition and representation of cluster is chosen.

2.4.3 Problems and challenges of clustering

There are a number of problems with clustering. Some of them [22]:

- Dealing with large number of data items can be challenging because of time complexity.
- The effectiveness of the method depends on the definition of "distance".
- The result of the clustering algorithm can be interpreted in different ways.

2.4.4 Clustering Algorithms

Clustering algorithms may be classified as [22]:

- Exclusive Clustering, such as K-means.
- Overlapping Clustering, such as Fuzzy C-means.
- Hierarchical Clustering, such as Agglomerative Hierarchical clustering.
- Probabilistic Clustering, such as Mixture of Gaussian.

In this project the hierarchical clustering algorithm will be used. So the next sections will be talked about the Hierarchical clustering.

2.4.5 Hierarchical clustering methods

Hierarchical clustering is a method of cluster analysis which aims to build a hierarchy of clusters (tree) [21]. Each node (cluster) in the tree (except for the leaf nodes) is the union of its children (sub cluster), and the root of the tree are singleton clusters of individual often, but not always, the leaves of the tree are singleton clusters of individual data objects [21].

A hierarchical clustering is represented graphically using a tree-like a diagram called dendrogram.

A dendrogram is a tree for visual classification of similarity, commonly used in Biology for grouping species, and it shows a hierarchy and the relation of subsets in a structure [23].

The dendrogram of average group linkage method as in Figure 2.17:

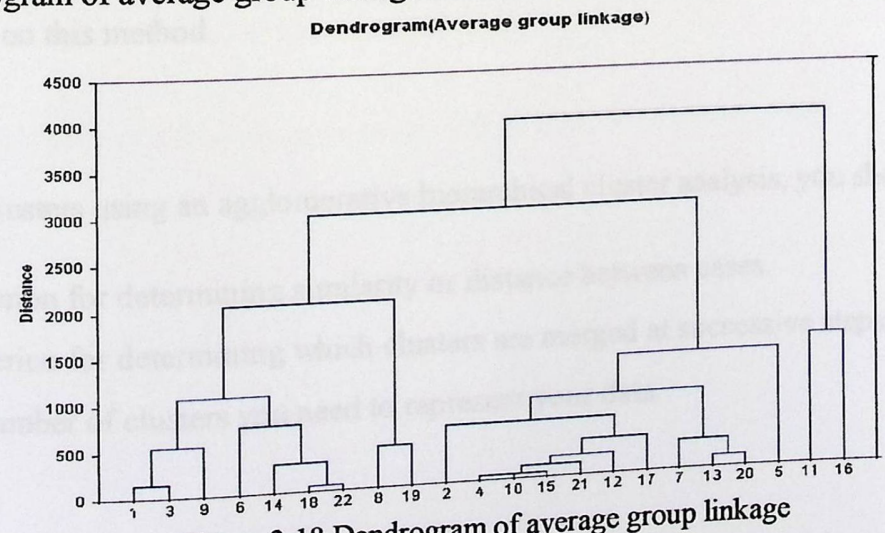


Figure 2.18 Dendrogram of average group linkage method [24]

Hierarchical clustering follows one of two approaches [25]:

- Agglomerative methods: each observation starts as a cluster and with each step merge observations to form clusters until there is only one large cluster. This approach is called button-up.
- Divisive methods: all observations start with one large cluster and proceed to split into smaller clusters items that are most dissimilar. This approach is called top-down.

Figure 2.18 illustrates the difference between the two approaches.

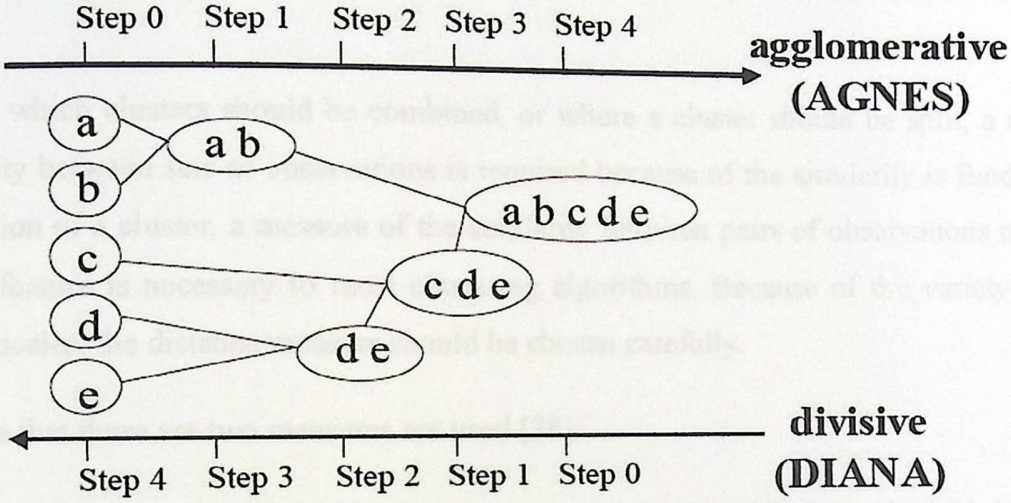


Figure 2.19 The two approaches of hierarchical clustering [26]
AGNES method merges clusters until there is only one cluster, where
DIANA method starts from one cluster and splits into smaller clusters.

Agglomerative hierarchical clustering technique will be used in this project, so in this section we will focus on this method.

To form clusters using an agglomerative hierarchical cluster analysis, you should select [27]:

- A criterion for determining similarity or distance between cases.
- A criterion for determining which clusters are merged at successive steps.
- The number of clusters you need to represent your data.

The algorithm of the basic agglomerative hierarchical clustering is explained in Algorithm 2.5

Algorithm 2.5

Compute the Linkage criteria (proximity matrix).

While (more than one cluster)

{ Merge the closest two clusters.

Update the linkage to reflect the linkage between the new cluster and the original cluster }

To decide which clusters should be combined, or where a cluster should be split, a measure of dissimilarity between sets of observations is required because of the similarity is fundamental to the definition of a cluster, a measure of the similarity between pairs of observations drawn from the same feature is necessary to most clustering algorithms. Because of the variety of feature types and scales, the distance measure should be chosen carefully.

To achieve that there are two measures are used [28]:

- In most methods of hierarchical clustering an appropriate metric is used which is a measure of distance between pairs of observations, such as:
  - ✓ Euclidean distance: it is often used in hierarchical clustering, the formula is:

$$Euclidean\ distance\ (d_{a,b}) = \sqrt{\sum_i (a_i - b_i)^2} \qquad (2-4)$$

  - ✓ Squared Euclidean distance, the formula is:

$$Squared\ Euclidean\ distance(d_{a,b}) = \sum_i (a_i - b_i)^2 \qquad (2-5)$$

In this project a Euclidean distance will be used as distance metric.

- Linkage Functions:

Linkage functions computes the dissimilarity between two groups of data points, Some commonly used linkage criteria between two sets of observations are: Single linkage (MIN), Complete linkage (MAX) and Group Average [21]:

1) Single linkage: it is used to describe minimum distance between two groups, see Figure 2.19

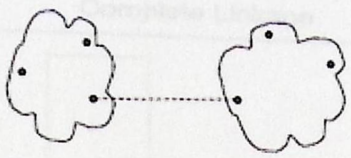


Figure 2.20 Single linkage [21]
Single linkage describes the minimum distance
between two clusters

It can handle non-elliptical shapes but it is sensitive to noise and outliers.

2) Complete linkage: it is used to describe maximum distance between two groups, see Figure 2.20

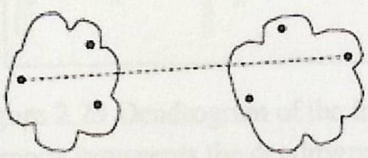


Figure 2.21 Complete linkage [21]
Complete linkage describes the maximum distance
between two clusters

It is less susceptible to noise and outliers but it tends to break large clusters.

3) Group average :it is used to describe average distance between two groups, see Figure 2.21

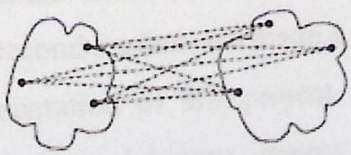


Figure 2.22 Group average [21]
Group average describes the average distance
between two clusters

Group average linkage compromises single and complete linkage, and it is less susceptible to noise and outliers.

When drawing a dendrogram for computing the dissimilarity between two groups of data points using three types of linkage functions the dendogram was as Figure 2.22:

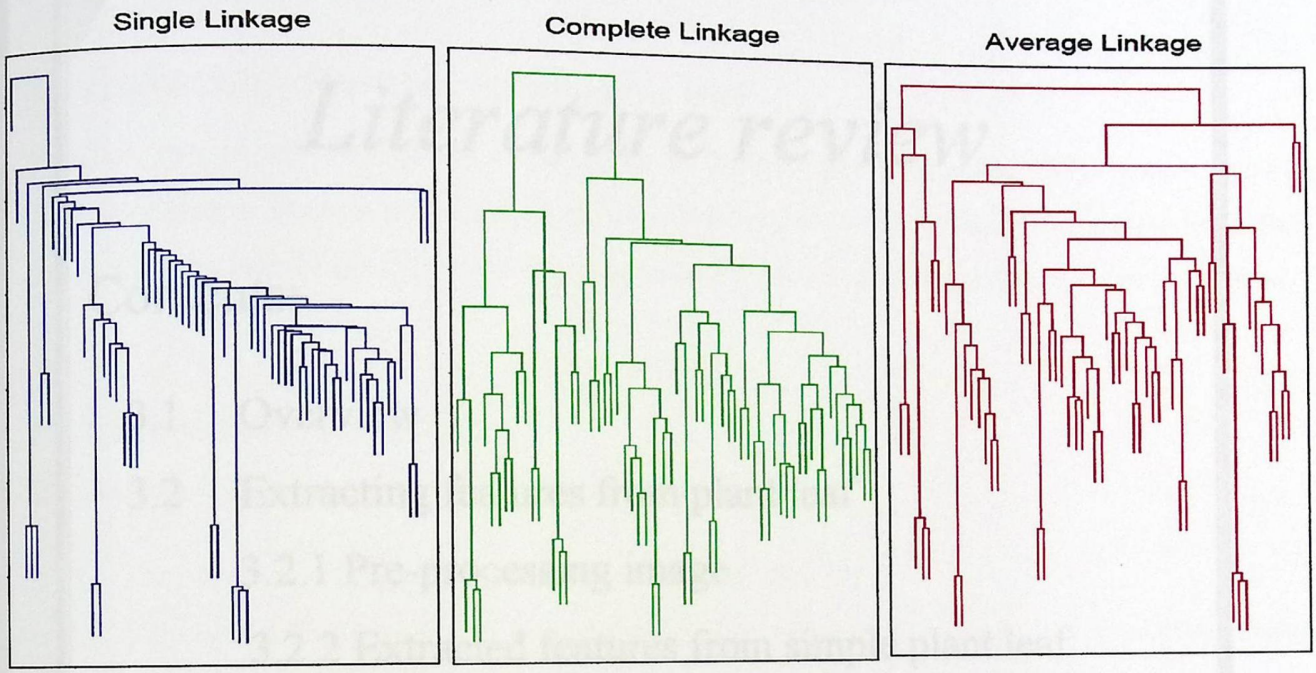


Figure 2.23 Dendrogram of the linkage types
First image represents the dendrogram of the single linkage, the middle for the complete, and the last for the average linkage [29]

In this project an average linkage will be used as linkage criteria.

2.4 Summary

This chapter included an introduction about the main morphological features of plant leaf, and the main characteristics of the tomato leaf, also this chapter included the difference between simple and compound leaf. In the second section, we made a review on the image processing part which is required for the implementation of this project. First, different color models were introduced including RGB, grayscale and binary images. Second, the basic morphological features were introduced erosion, dilation, opening, and closing. Then, the connected component algorithm was discussed. Finally a cluster analysis, hierarchal clustering methods were covered.

Chapter Three

Literature review

Contents:

3.1 Overview

This chapter contains a summary of some important contributions related to leaf classification. The chapter includes different methods to extract the features, the methodologies that were used to extract the features from plant leaf. In the final section of this chapter, the importance of these works is presented in this project.

3.2 Extracting features from plant leaf

There are many methodologies to extract features from simple plant leaf in the literature in order to classify plants. These methodologies differ in the way of studying the features were extracted, but most of the papers agree on how the image is pre-processed before the features are extracted.

3.2.1 Pre-processing image

The methodology that was used to pre-process included:

1) The image was acquired by scanners or digital cameras.
2) Converting RGB image to binary image: The color of the image was converted firstly from RGB to grayscale using Formula 2-3.

   Then image was converted from grayscale to binary using suitable threshold (T), as in Algorithm 2-1.

3) A filter was applied to eliminate noises in the image.

   The whole operation is demonstrated in Figure 3.1.

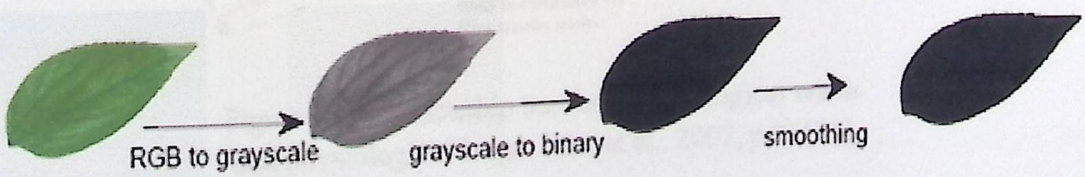


Figure 3.1 Pre-processing example (Xu, et al., 2007, p.2) [31]

3.2.2 Extracted features from simple plant leaf

There are different leaf features were extracted in the literature, some of these features are:

First: Extracting morphological and geometrical features from plant leaf

Some geometrical features were extracted in literature works are (Xu, et al., 2007, p.2) [31]:

1) Diameter (D): the diameter is defined as the longest distance between any two points on the margin of the leaf.
2) Physiological Length (Lp): physiological Length is defined as the distance between the two terminals. This feature is only the feature was needed the human interfered to detect the two terminals of the main vein of the leaf via mouse click. Is defined as the physiological length.
3) Physiological Width (Wp): drawing a line passing through the two terminals of the main vein, one can plot infinite lines orthogonal to that line. The number of intersection pairs between those lines and the leaf margin is also infinite. The longest distance between points of those intersection pairs is defined at the physiological width. The relationship between physiological length and physiological width is illustrated in Figure 3.2.

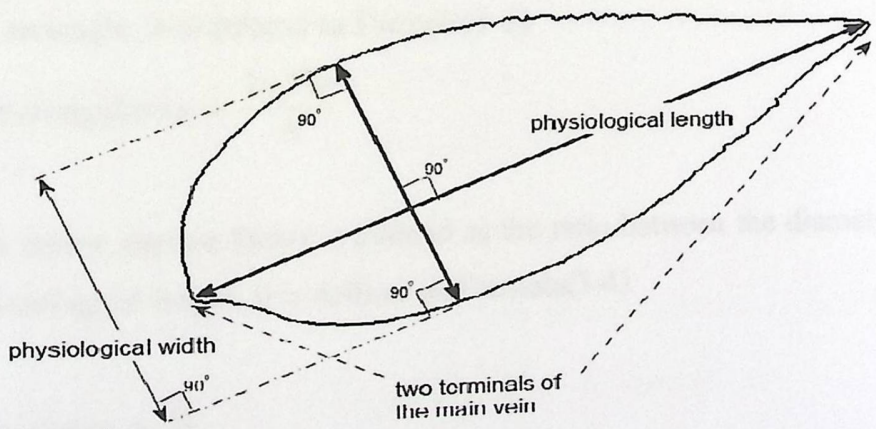


Figure 3.2 Relationship between physiological width and physiological length (Xu, et al., 2007, p.2) [31]

4) Leaf Area (A): leaf area is calculated by counting the number of pixels of binary value 1 on smoothed leaf image.

5) Leaf Perimeter (P): leaf perimeter is calculated by counting the number of pixels consisting leaf margin.

These geometrical features help to extract the morphological features of plant leaf, so some morphological features are defined based on geometrical features are (Xu, et al., 2007, p.2) [31]:

1) Smooth factor: the effect of noises to image area was used to describe the smoothness of leaf image. The smooth factor is defined as the ratio between area of leaf image smoothed by 5 × 5 rectangular averaging filter and the one smoothed by 2 × 2 rectangular averaging filter.

2) Aspect ratio: the aspect ratio is defined as the ratio of physiological length to physiological width, it is defined as Formula (3-1)

$$\text{Aspect ratio} = \frac{L_P}{W_P} \qquad (3-1)$$

3) Form factor: the form factor is used to describe how the shape of the leaf is different from a circle, it is defined as Formula (3-2)

$$\text{Form factor} = \frac{4\pi A}{P^2} \qquad (3-2)$$

4) Rectangularity: rectangularity is used to describes how the shape of the leaf is different from a rectangle, it is defined as Formula(3-3)

$$\text{Rectangularity} = \frac{L_P\, W_P}{A} \qquad (3-3)$$

5) Narrow factor: narrow factor is defined as the ratio between the diameter of the leaf and the physiological length, it is defined as Formula(3-4)

$$\text{Narrow factor} = \frac{D}{L_P} \qquad (3-4)$$

6) Perimeter ratio of diameter: ratio of perimeter to diameter, and calculated as Formula (3-5)

$$\text{Perimeter ratio of diameter} = \frac{P}{D} \qquad (3-5)$$

7) Perimeter ratio of physiological length and physiological width: this feature is defined as the ratio of leaf perimeter and the sum of physiological length and physiological width, thus as Formula (3-6)

$$\text{Perimeter ratio of Lp and Wp} = \frac{P}{L_P + W_P} \qquad (3-6)$$

8) Vein features: by performing morphological opening on grayscale image with flat, disk-shaped structuring element, the results look like the vein.

9) Image Thresholding: Thresholding separates the leaves from their background creating binary image that facilitates feature extraction and evaluation.

10) Determination of the Center of Gravity: For a leaf surface is described by function f(m, n), consisting of N pixels, the Center of Gravity coordinates (m*, n*) can be calculated as Formula (3-7)

$$m* = \frac{1}{N} \sum_{(m,n)\in R} \sum m \qquad n* = \frac{1}{N} \sum_{(m,n)\in R} \sum n \qquad (3-7)$$

11) Moments of Inertia: the moments of inertia for binary image defined with respect to the center of gravity of the leaf as Formula (3-8)

$$\mu_{p,q} = \sum_I \sum_J (i - m*)^p \ (j - n*)^q \qquad (3-8)$$

Where $p, q = 0,1,2.$

12) Then, the leaf oriented which is determined by using the following functions:

$$I(\theta) = \sum\sum [(n - n*)\cos(\theta) - (m - m*)\sin(\theta)]2 \qquad (3-9)$$

Where the I is array of θ. Resulting in following angle θ:

$$\theta = \frac{1}{2} \tan^{-1} \left[\frac{2.\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}} \right] \qquad (3-10)$$

Second: Extracting general visual and domain related visual features from plant leaf

General visual features used to describe common features of all images without related to specific type, or content of the image, such as color, texture and shape. While, domain features related to visual features more important of leaf classification.

After combining these two types, three aspects of leaf features were studied. These aspects are (Wu, et al., 2006, p.6) [32]:

1) Leaf shape: Four visual features are defined to represent the shape of leaf :

- Slimness: This is the ratio of length to width of leaf. This feature is similar to the aspect ratio in Equation 3-1.

- Roundness: express the extent to which the shape is circle. This feature is equivalents to form factor in Equation 3-2.

- Solidity: the ratio of internal area to external (Figure 3.3), defined as Formula (3-11)

$$Shape\ Solidity = \frac{S1}{S2} \qquad (3-11)$$

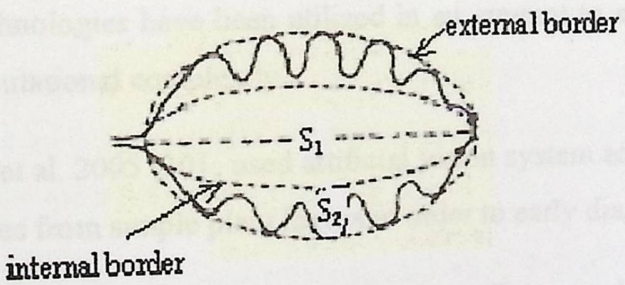


Figure 3.4 Internal and external area (Wu, et al., 2006, p.6) [32]

- Moment invariants: slimness and roundness is a rough description, so in order to describe the leaf shape in detail they adopted moment invariants as shape descriptor in their method.

2) Leaf Dent: Four visual features are extracted to represent leaf dent. These feature are:

- Coarseness: this feature is express of coarseness of the leaf margin.
- Size: this is a large scale of leaf dent appear.
- Angle: it is measured by the extreme value at each scale.
- Sharpness of leaf.

3)Leaf Vein: the venation of main vein and secondary vein give the structure of whole plan, by analyzing the venation, more characteristic of leaf obtained. So two features are used to represent this aspect:

- Ramification: The number of ramification is used to measure the complexity of venation.
- Camber: express of the degree of crook of main vein.

3.3 Important contributions related to leaf classification

A lot of efforts have been directed towards the extracting features from simple plant leaf, and many techniques and technologies have been utilized in an attempt to achieve this process with minimum error and computational complexity.

In their work, Panayiotis et al. 2005 [30] , used artificial vision system and extracting geometrical and morphological features from simple plant leaves in order to early diagnosis plant diseases.

Their proposed system includes: an artificial vision system (frame grabber and camera), image processing algorithm implemented in Lab View (operating under the control of proposed GUI that use image processing libraries), and a feed-forward neural network based classifier implemented in MatLab.

This paper applied fuzzy surface technique that is used for building fast model of system from subset of candidate features. The proposed system was able to automated image capture, counting the number of leaves for each category and morphological classification for holes.

The system provides description and evaluation of overall image processing system by (GUI) Graphical Users Interface was developed to obtain efficient interaction image processing programs, GUI was designed using visual programming in MatLab, Labview and IMAQ libraries to control the camera and frame-grabber parameters. (See Figure 3.4 for more illustration)

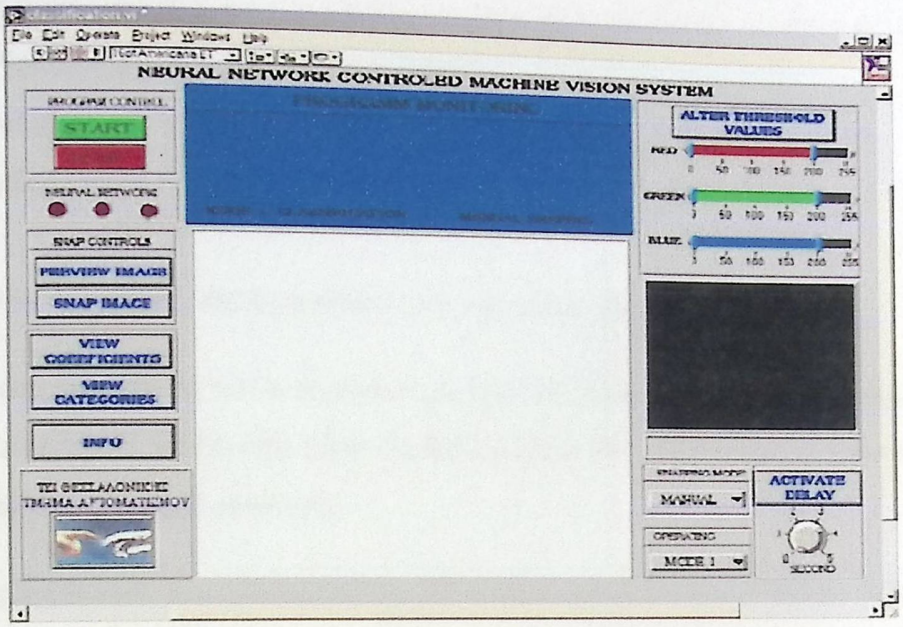


Figure 3.4 the Graphical Users Interface [30]

In any case, vary high classification success was achieved at about 99%.

In their work, Wu, et al. 2006 [32]' depended on extracting general visual and domain related visual features from plant leaf. After the feature extraction the researcher used the feed-forward back-propagation neural network to recognize.

The recognition system in C++ has been implemented on a PC (CPU: PIV 2.6GHz, RAM: 512M). Using 1200 leaf images, the average time recognizing one image is about 0.45 seconds and the training time is about 12.3 seconds.

In their work, Xu et al. 2007 [31], depended on the geometrical and morphological features to be extracted.

After processing image, researcher applied number of steps to classify plant based on their leaves, these steps included:

1. Principal computer analysis (PCA) ,The PCA is used to present the information of original data as the linear combination of certain linear irrelevant variable, in mathematically the PCA convert the data to a new coordinate system each coordinate is called PCA .
2. Train for Probabilistic Neural Network (PNN).
3. Test for PNN.
4. Display and compare results.

After this method is applied, the researcher team arrived to this conclusion:

The computer can classify 32 kinds of plants via the leaf images that loaded by digital cameras or scanners. When compare PNN with other method it finds the PNN is fast speed on training, easy in implementation and simple structure.

12 features are extracted and processed by PCA then input vector of PNN. The result of this algorithm was accuracy greater than 90% on 32 kinds of plants. Future work is under consideration to improve it.

Some of previous works related to extracted features from plant leaf have many shortage in different sides, all of the previous works depended on simple plant leaf to extract features. Although Panayiotis et al. 2005 [30] solved a problem of holes, but he faced a problem to add and abstract suitable features each time. Wu, et al. 2006 [32] the problem of their work is the lack of representation of domain features of leaves. Xu, et al. 2006 [31] can't get rid of the user intervention in extracting features.

3.4 Importance of literature review in this project

Although many of previous works emphasized on extracting features from simple leaves and not compound leaves, they give the project important information, some of these information are:

1) The leaflet of a compound leaf is simple leaf, so some methodologies in these papers can be followed to extract important features from leaflet.

2) These works presented to this project important information about the drawbacks of previous works, some of them mentioned in chapter one, and the important improvements that will be in the future works.

3.5 Summary

This chapter contained a summary of literature reviews related to this project. Firstly, the method that was used to pre-process image was introduced in this chapter. Then, the main two different ways to classify features that were extracted from plant leaf were discussed in the previous section in this chapter, these two ways are: extracting morphological a geometrical features of plant leaf, and extracting general visual and domain related visual features.

In addition, the chapter handles the main contributions to the previous work, and the importance of these contributions in features extraction of compound plant leaf project.

Chapter Four

Methodology

Contents

4.1 Overview

This chapter covers the methodology that we used in this project to extract the features of the compound leaves. The chapter includes the preparation phases to implement this project; the tools were used in this project, and the methods were used to verify and evaluate the extracted features. These methods are: identifying a point of matching and hierarchical clustering of the plant leaf.

4.2 Preparation phases for implementation of the project

The first step of this project, is studying the previous works related to features extraction from plant leaf, through reading many papers that talked about features extraction of plant leaf as mention viewed in Chapter 3. Then we are consulting an expert in the field of plant to do so we relied on the second supervisor who works at the Biotechnology Training and Research Unit in Palestine Polytechnic University. He plays an important role in evaluating the features that can be extracted from compound leaf.

After that there are many phases are followed before implementing the project, such as collecting images of tomato leaf, using MATLAB® to implement a project, and Defining the requirements of project. These phases are explained in this section

4.2.1 Collecting images of tomato leaves

Because a tomato leaf is taken as a case study in this project, the second step of this project is collecting images for tomato leaves. These images are obtained from online database of (111) leaves of different types [9].

4.2.2 Using MATLAB® to implement a project

To study the feature extraction process, MATLAB® was proposed to be used in this project.

MATLAB® "is a high-level language with an interactive environment that enables us to perform computationally intensive tasks faster than with traditional programming languages such as C, C++ and FORTRAN." [33] Because it has many predefined functions.

The Image Processing Toolbox found in MATLAB® is a collection of functions that extends the capability of the MATLAB® and supports a wide range of image processing operations, such as: geometric operations, neighborhood and block operations, linear filtering and filter design, image analysis and enhancement, binary image operations, and region of interest operations [34].

In other hand, Statistics Toolbox provides a comprehensive set of tools to assess and understand data. Statistics Toolbox includes functions and interactive tools for modeling data, analyzing historical trends, simulating systems, developing statistical algorithms, and others [35].

4.2.3 Defining the requirements of project

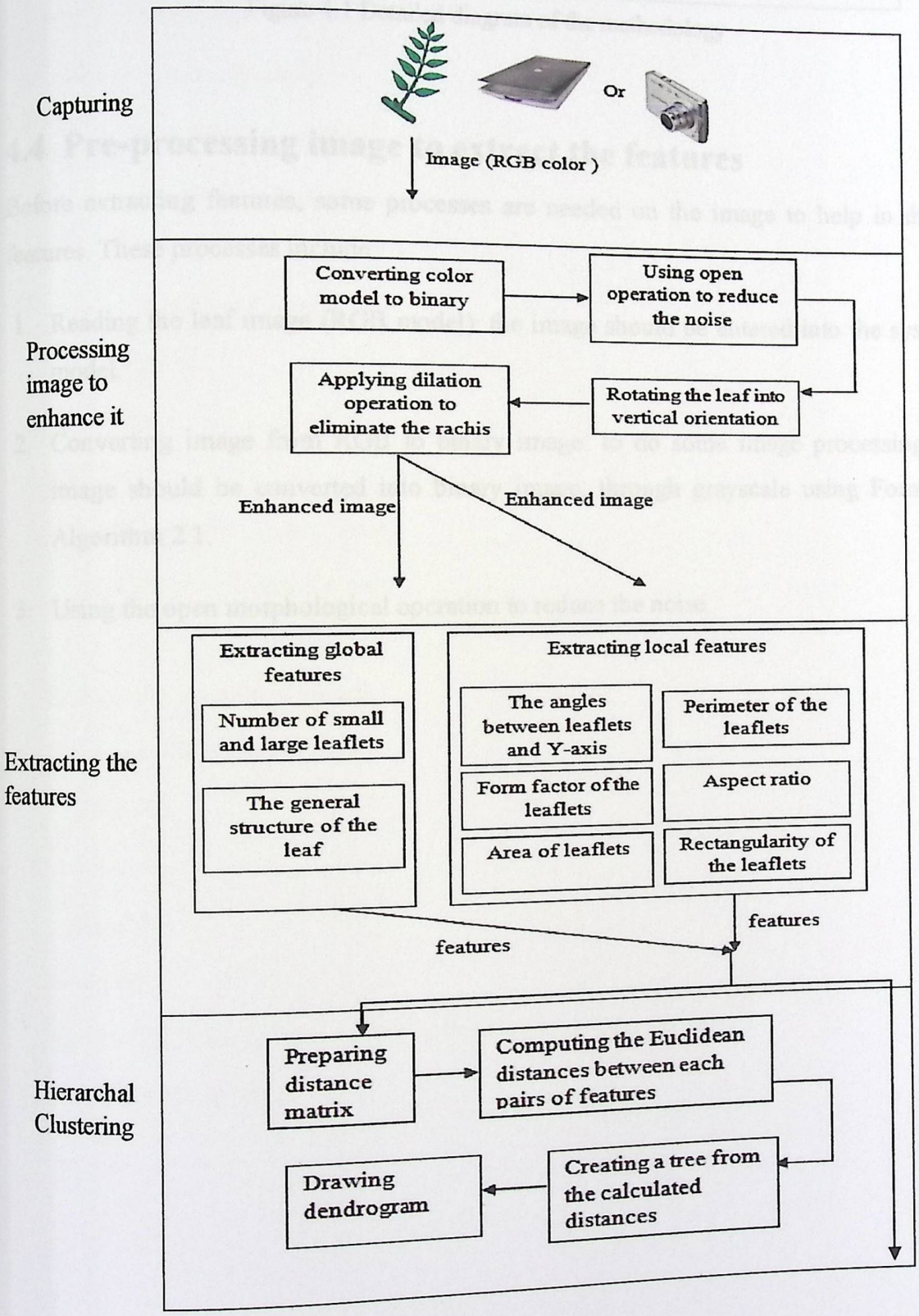
Before talking about the methodology of this project, it should be emphasized that the accuracy of the features will be extracted depending on the image of the leaf, so in order to make the features extraction more accurate and effective by defining the requirements of project on these images were assumed.

These requirements are:

1. The leaves should be digitized using a scanner. Although we use digital images, so we assume that the user who will work on the system in the future should scan the leaf first.
2. The image should be a complete leaf of plant and not leaflet.
3. The leaflets of compound leaf should not be overlaps; this can be adjusted manually by the user during the scanning process.
4. The image should be in RGB format and the leaves should be isolated from any background.
5. The resolution of the image should be 800 x 600 pixels in this project.
6. There is no restriction on the direction of leaves.

4.3 The Detailed diagram of the project's methodology

The diagram in Figure 4.1 describes and summarizes the methodology is used in this project. All of the steps will be described and explained in this chapter.



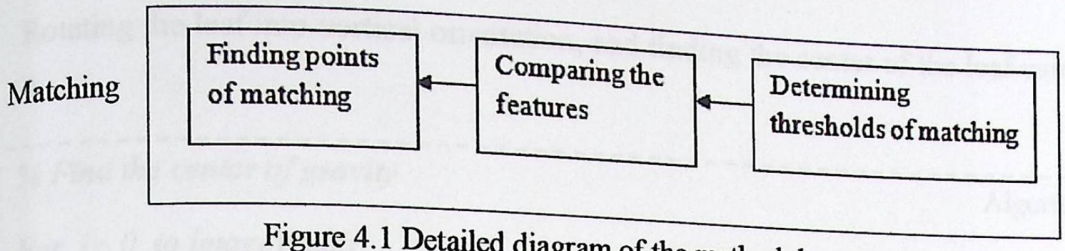


Figure 4.1 Detailed diagram of the methodology

4.4 Pre-processing image to extract the features

Before extracting features, some processes are needed on the image to help in the extracting features. These processes include:

1. Reading the leaf image (RGB model): the image should be entered into the system in RGB model.
2. Converting image from RGB to binary image: to do some image processing techniques, image should be converted into binary image, through grayscale using Formula 2-3 and Algorithm 2.1.
3. Using the open morphological operation to reduce the noise.

4. Rotating the leaf into vertical orientation, and finding the center of the leaf using Formula 3-8.

Algorithm 4.1

```
% Find the center of gravity
For  i= 0  to imageWidth
  For j =0 to imageHeight
        If ( BW( i , ,j) is white )
           X_Sum = X_Sum+i
           Y_Sum = Y_Sum+j
          Count = Count + 1
        End if
  End For
End For
X_leafCenter = X_Sum / Count
Y_leafCenter = Y_Sum / Count

% Find the moment of inertia
For each white pixel in image
   Find μ1,1
   Find μ2,0
   Find μ0,2
End For
Theta= 0.5 * tan (2*μ1,1 / (μ2,0  -  μ0,2 ))
Rotate image by theta
```

5. Applying the dilation morphological operation many times on binary image to eliminate the rachis of the compound leaf.

Algorithm 4.2

```
For i= 0 to threshold
    dilationImage = imdilate(openImage , se)
End For
```

4.5 The main features of compound leaf

As mentioned in Chapter 2, the compound leaf contains a number of simple leaves (leaflets) that make the process of extracting features from compound leaf is more difficult than simple leaf, because we should detect the relationship's positions between leaflets.

In this project, we have defined two types of features: global features which are extracted from the compound leaf as a whole, and local features which are extracted from each leaflet in the compound leaf. The following is an explanation of each part.

4.5.1 Global features

The global features have two components: the number of leaflets and the general structure of the plant.

First: The number of small and large leaflets:

To find the number of leaflets in an image, the connected component (it is explained in Chapter 2) technique will be used, because after applying the technique each leaflet will be a connected component.

To make a more distinctive feature, we can count the number of small and large leaflets from the connected components. This feature should be extracted by knowing the area of each leaflet, and by using a good threshold that helps to determine the area of small and large leaflets.

To find the number of small and large leaflets, the below algorithm will be followed:

Algorithm 4.3

```
give label for each leaflet black and white image(x)
For i=1 to number of leaflets
    Area (i) =find area in x (label==i)
End For
If Area >threshold Then
 Number large leaflets++;
else
Number small leaflets++;
End if
```

Second: The general structure of the plant leaf

After the leaf is rotated into the vertical orientation, the general structure of the plant leaf can be achieved. Firstly, the center of each leaflet should be detected. Then, the structure of leaf is built by establishing a graph of the leaflets. In Figure 4.2 each node represents the distance between the leaflet's center and the leaf's center. Figure 4.2 illustrates this process.

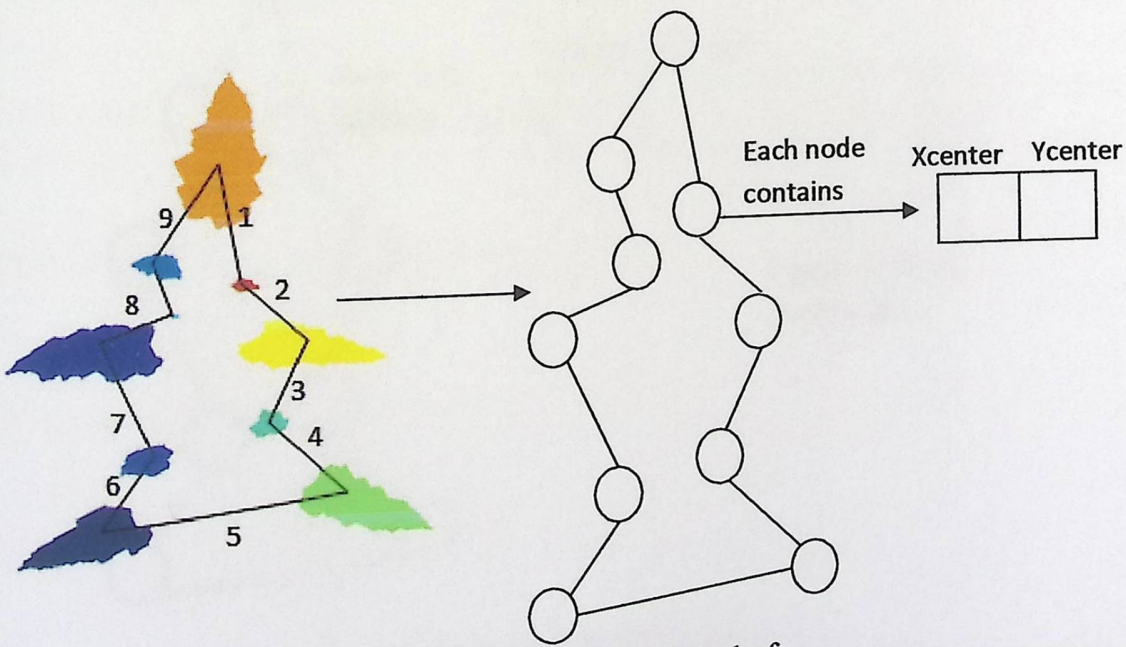


Figure 4.2 Extracting the structure of the leaf
by establishing a graph of leaflets

To find the distances between the leaflets centers and the leaf center, the below algorithm will be followed:

Algorithm 4.4

```
For i=1 to number of leaflets
    centroid (i) =Find the center of the leaflet (i)
    X_center(i) = X_centroid(i) – X_leafCenter;
    Y_center(i) = Y_centroid(i) – Y_leafCenter;
End For
```

4.5.2 Local features

The local features are extracted from each leaflet. From Figure 4.2 we can establish a vector of features for each node. The vector contains some features were discussed in Chapter 3 such as the area, diameter, aspect-ratio, form factor, etc. Figure 4.3 illustrates the graph.

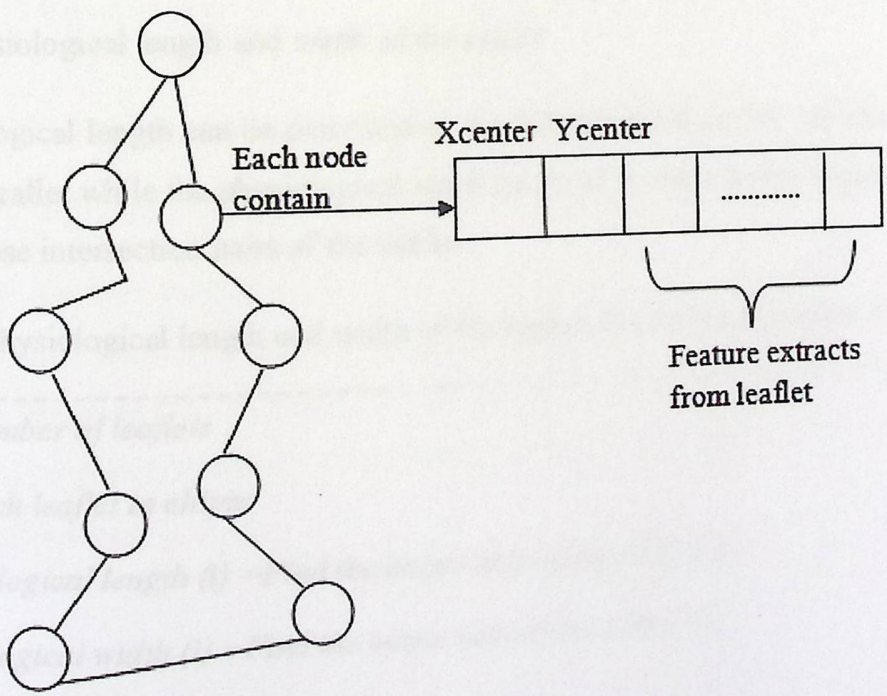


Figure 4.3 Graph for features of compound leaf

Features will be extracted from each leaflet in this project are:

- Area of the leaflet: which is the number of pixels represents the leaflets.
- Perimeter of the leaflet: which is the number of the pixels represents the boundary of the leaflet.
- The form factor of leaflet

It can be described as how the shape of the leaflet is different from a circle.

To find the form factor of leaflet, the below algorithm will be followed:

Algorithm 4.5

```
For i=1 to number of leaflets
    Find the perimeter (i)
    Find the area (i)
    Form factor (i) = (4*PI * area (i))/ (perimeter (i) ^ 2)
End For
```

- The physiological length and width of the leaflet

The physiological length can be described as the distance between the two terminals of the main vein of the leaflet while the physiological width can be described as the longest distance between points of those intersection pairs of the leaflet.

To find the physiological length and width of the leaflet, the below algorithm will be followed:

Algorithm 4.6

```
For i=1 to number of leaflets
    Put each leaflet in ellipse
    Physiological length (i) =Find the major axis of the ellipse (i)
    Physiological width (i) =Find the minor axis of the ellipse (i)
End For
```

- The aspect ratio of the leaflet

It is the ratio of physical length to physical width of the leaflet.

To find the aspect ratio of the leaflet, the below algorithm will be followed:

For i=1 to number of leaflets

Algorithm 4.7

Aspect ratio (i) = physiological length (i) / physiological width (i)

End For

- The rectangularity of the leaflet

It is used to describe how the shape of the leaflet is different from a rectangle.

For i=1 to number of leaflets

Algorithm 4.8

*Rectangularity (i) = (physiological length (i) * physiological width (i)) / area (i)*

End For

4.6 Verifying the importance of extracted features

After talking about extracting the main features of compound leaf, it should be verified that extracted features play an important role in representing the leaves by using two important processes on the same type of plant (Tomato as a case study in this project): matching features between two compound leaves and clustering analysis for compound leaves. The following is an explanation of each part.

4.6.1 Matching features

After extracting the main features of compound leaf, the first way to make sure that the extracted features from the leaf reflects the leaf, is matching features between two compound leaves. It is used to search for matching between two compound leaves depending on previous features of compound leaf.

Choosing suitable thresholds is very important to determine the degree of matching between compound leaves.

The most important feature in matching process is the position of each leaflet and they should be close together then comparing the main features is started between compound leaves. These features are angle of each leaflet, form factor, aspect ratio, and rectangularity.

To find matched leaflets between two images of compound leaves, the below algorithm will be followed:

Algorithm 4.9

```
For i=1 to number of leaflets in image1
 For j=1 to number of leaflets in image2
    If (-1 * threshold1< X-center (i) and X-center (j) < threshold1) and
        (-1 * threshold1 < Y -center (i) and Y-center (j) < threshold1) Then
      If (-1 * threshold2 < Angel (i) and Angle (j) < threshold2) Then
       If (-1 * threshold3< Form factor (i) and Form factor (j) < threshold3) Then
       If (-1 * threshold4< Aspect ratio (i) and Aspect ratio (j) < threshold4) Then
       If (-1 * threshold5< Rectangularity (i) and Rectangularity (j) < threshold5) Then
            Matching was done
        End if
      End if
    End if
  End if
 End if
End For   End For
```

The ratio of matching is determined by dividing the number of matched leaflets on the number of leaflets belongs to the leaf which has lowest number of leaflets.

- After extracting features from each leaflet in each leaf, the features are stored in matrix; this matrix is a three dimension array .The first index of the array indicates to the leaf number, the next index indicates to the leaflet number in this leaf and the last index indicates to the features . Algorithm 4.10 shows the storing features process:

Algorithm 4.10

```
For I=1 :number of leaves in the sample
    For J=1 :number of leaflet in I leaf
        Leaflet_features (I, J, 1)=area (J)
        Leaflet_features (I, J, 2) =perimeter (J);
        Leaflet_features (I, J, 3 ) =X_center(J);
        Leaflet_features(I, J, 4)=Y_center(J);
        Leaflet_features(I, J, 5) =angle (J);
        Leaflet_features(I, J, 6)=form_factor(J);
        Leaflet_features(I, J, 7) =aspect_ratio(J);
        Leaflet_features(I, J, 8)=rectangularity(J);
        Leaflet_features(I, J, 9)=large_leaflets(J);
        Leaflet_features(I, J, 10)=small_leaflets(J);
    End J
    End I
```

- Then, each leaflet is compared to other leaflets in other leaves using Euclidean distance of leaflet's features, In Figure 4.4 Leaflet 1 in Leaf 1 compared with each leaflet in the database and using the Euclidean distance to find the distance between each two leaflets' features, which called a Mean Square Error (MSE). This process is applied on each leaflet in the database. See Figure 4.4.

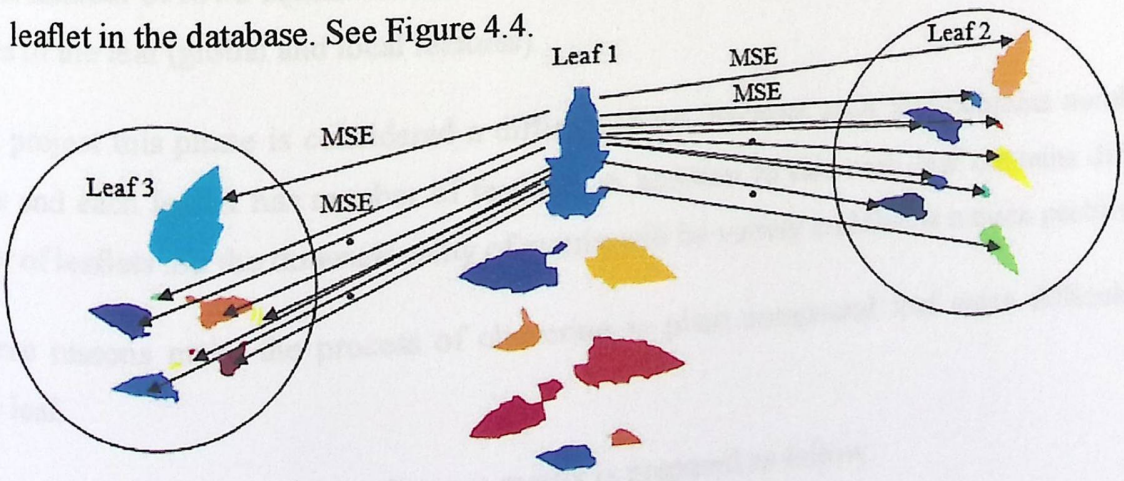


Figure 4.4 Finding the distance between each two leaflets in the sample

4.6.2 Clustering analysis of plant leaves

The second way to verify the importance of extracted features from the compound leaf is a clustering of the plant leaves.

Clustering process of the plant leaves shows if the extracted features can be used to distinguish the groups of plant from each in the same plant leaf (tomato plant in this project), also the clustering process uses to reduce the numbers of the plant's types to help in the classification.

Both local and global features are used in this process, and all features are normalized to become in the same range (0-1).

In this project a hierarchal clustering method is used on the plant leaves, the following steps are followed in this project to do a clustering process:

- Preparing the distance matrix of the leaves.
- Choosing the suitable distance metric.
- Choosing the suitable linkage function.
- Drawing a dendrogram of the clusters.

Each step will be explained in details.

Preparing the distance matrix of the leaves:

Before doing cluster process, the matrix of plant's features should be prepared .This matrix contains number of rows equals the number of leaves in the sample, each row contains the main features of the leaf (global and local features).

In this project this phase is considered a difficult phase, because each leaf contains number of leaflets and each leaflet has number of features, in addition to this each leaf contains different number of leaflets .So the dimensionality of matrix will be variety and this is a main problem.

All these reasons make the process of clustering in plant compound leaf more difficult than simple leaf.

To overcome these problems, the distance matrix is prepared as follow:

Algorithm 4.11 shows this process.

Algorithm 4.11

```
For I=1 :number of leaves in the sample
  For J=1 :number of leaflet in I leaf
   For K=1 :number of leaves in the sample
    For M=1 :number of leaflet in K leaf
For D=1 to 10
      MSE1(D)= (Leaflet_features (I, J, D) - Leaflet_features (K, M, D))^ 2;
End D
   Average=sqrt(sum(MSE1));
   MSE2(M) =Average;
  End M
 End K
End J
End I
```

- After this, the minimum distance between each leaflet in the leaf and other leaves is taken and stored, Figure 4.5 shows this step.

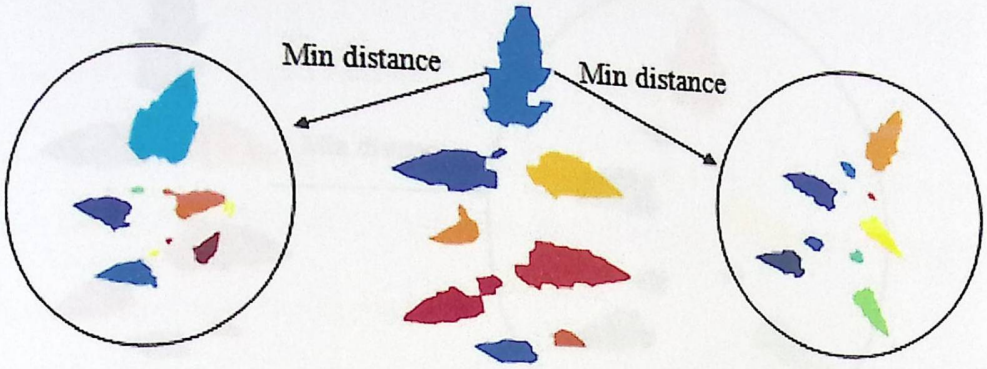


Figure 4.5 The minimum distance between each leaflet and other leaves

- Then, a matrix contains the minimum of distances between each leaflet in the leaf and other leaves is prepared for each leaflet. See Figure 4.6.

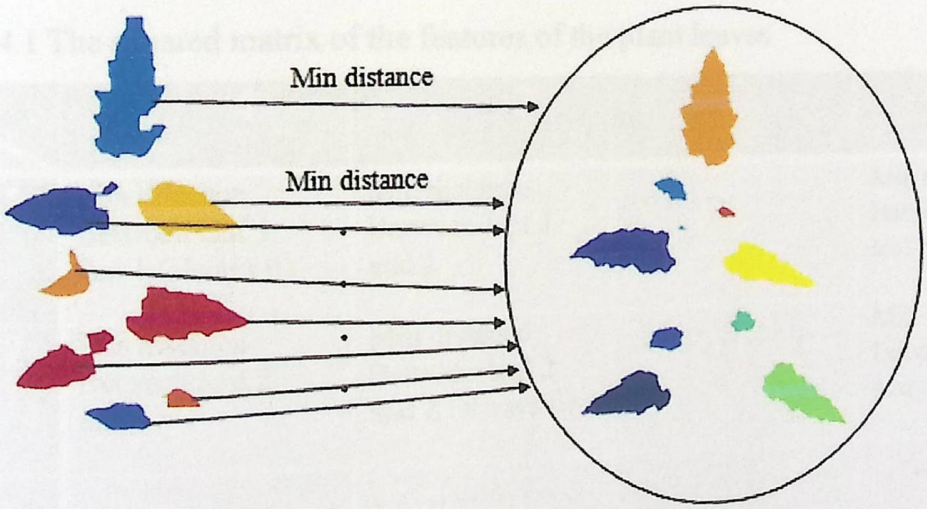


Figure 4.6 Minimum distance between each leaflet in one leaf and other leaves

- Next, the minimum distances between each leaf and others is stored in another matrix. See Figure 4.7.

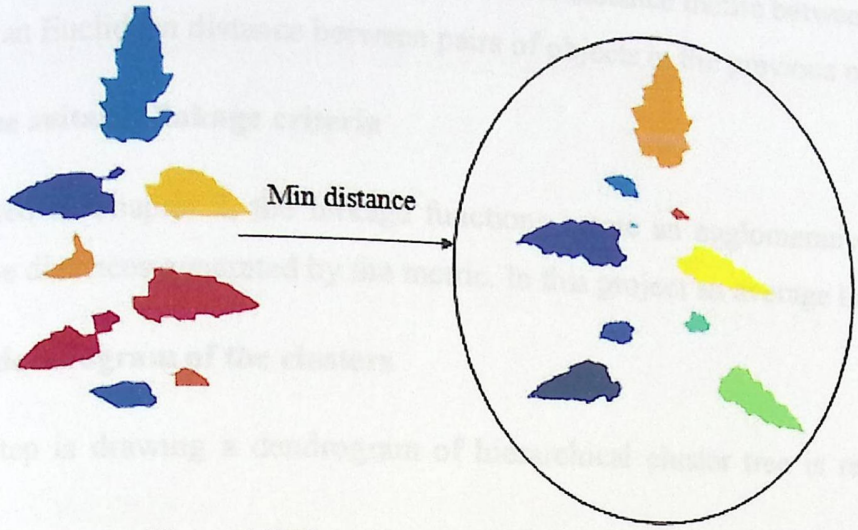


Figure 4.7 The minimum distance between each two leaves

These steps enabled us to find the matrix of leaves' features, this square matrix was contained the following data as in Table 4.1. Rows and columns of matrix correspond to objects (leaves).

Table 4.1 The squared matrix of the features of the plant leaves

leaf / leaf	1	2	.....	n
1	Min distance Between leaf 1 and 1 (always 0)	Min distance Between leaf 1 and 2	.....	Min distance Between leaf 1 and n
2	Min distance Between leaf 2 and 1	Min distance Between leaf 2 and 2 (always 0)	.....	Min distance Between leaf 2 and n
. .	.....	.....	.....	.....
n	Min distance Between leaf n and 1	Min distance Between leaf n and 2	.....	Min distance Between leaf n and n (always 0)

Choosing the suitable distance metric

The cluster process starts from choosing the suitable distance metric between a pair of objects. In this project an Euclidean distance between pairs of objects in the previous matrix is chosen.

Choosing the suitable linkage criteria

As mentioned in Chapter 2, the linkage functions create an agglomerative hierarchical cluster tree from the distances generated by the metric. In this project an average linkage is used.

Drawing a dendrogram of the clusters

The final step is drawing a dendrogram of hierarchical cluster tree is represented by linkage function.

The previous three steps can be summarized in Algorithm 4.12.

E_distance=compute Euclidean distance(feature matrix, 'Euclidean') Algorithm 4.12
Hc_tree =create a hierarchical cluster tree (E_distance , 'average')
dendrogram(Hc_tree)

4.7 Summary

This chapter included the preparation phases for implementation of the project that contained: studying previous works related to this project, consulting an expert in the field of plant, collecting images of tomato leaves, and using MATLAB® to simulate a project.

The proposed to be used to extract features from compound leaf was explained. These features can be extracted from compound leaf as a whole, such as: number of leaflets and the structure of leaf, and others, from each leaflet of compound leaf.

The proposed to be used to verify extracted features from compound leaf was explained. These ways are used to search for matching between two compound leaves depending on extracted features, and to distinguish the groups of plant from each in the same plant leaf.

Chapter Five

Experiments and Results

Contents:

5.1 Overview

In Chapter 4, a set of algorithms were developed to apply them in the project, these algorithms are converted into implementation code (see Appendix A). In this chapter the experiments and results of these algorithms will be covered.

The experiments will be done in three steps; pre-processing experiments, extracting features experiments and verifying the features. The results of these experiments will be also introduced in this chapter.

Pre-processing experiments contain the testing of algorithms were explained in chapter 4 and experiments will be applied on a compound leaf image before extracting features from it, such as: converting the color model of the image and rotating the leaf into vertical orientation.

Then the extracting features experiments will be applied to pre-process the compound leaf image to extract both global and local features. Finally experiment will be done to verify and evaluate the extracted features.

5.2 Testing Environment

The system is written using image processing, statistics toolboxes and GUIDE (Graphical User Interface Development Environment) in MATLAB®. It runs under windows platform.

5.3 Testing Sample

The sample that was chosen to test the algorithms is taken from the online database [9], the experiments were applied on 29 tomato leaf from this database (see Appendix B); all these leaves conformed the conditions were assumed in Chapter 4.

5.4 Experiments and Results

The following sections contain the experiments and results for the project steps: pre-processing, features extraction and verifying the features experiments and results.

5.4.1 Pre-processing experiments and results

Pre-processing image is a necessary step before extracting features. So this section contains experiments and results of this process, the image processing toolbox in MATLAB® was used to do this process. The experiment process contains:

1. Converting image from RGB to a binary image through grayscale using suitable functions in MATLAB®.

'rgb2gray' function was used to convert from RGB image to grayscale image as in Figure 5.1.

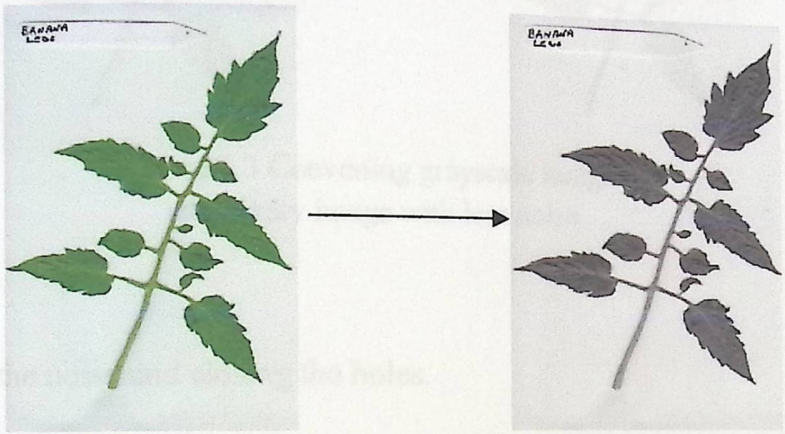


Figure 5.1 Converting RGB image of compound leaf into grayscale image.

Then, to convert image from grayscale into binary, the first experiment was done using 'dither' function in MATLAB®, but the output image was not clear and contained noise as in Figure 5.2.

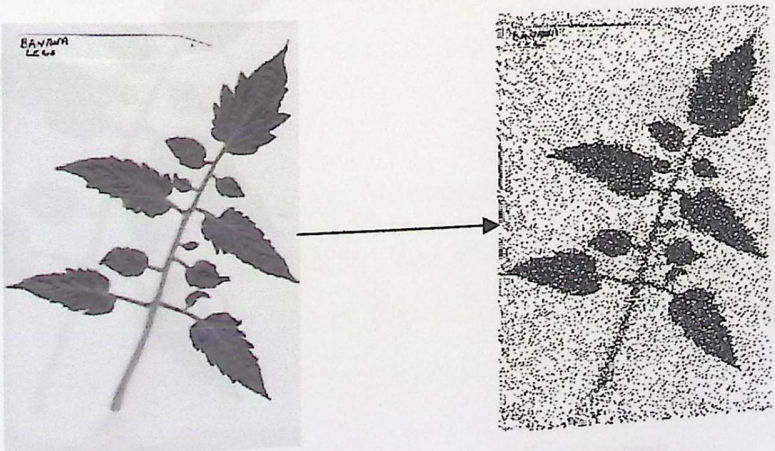


Figure 5.2 Converting grayscale image into binary image with noise

So, to make the binary image more clear and accurate, we used the 'graythresh' and 'im2bw' functions in MATLAB®. The output image of this experiment became clearer than the first experiment as in Figure 5.3.

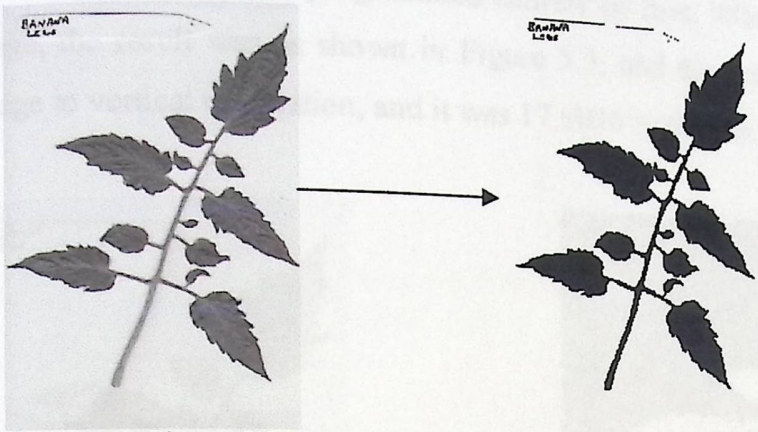


Figure 5.3 Converting grayscale image into binary image with less noise

2. Reducing the noise and closing the holes.

This process was done using 'open' morphological operation to eliminate the rest of noises and closing any holes in the leaf. See Figure 5.4 for more illustration.

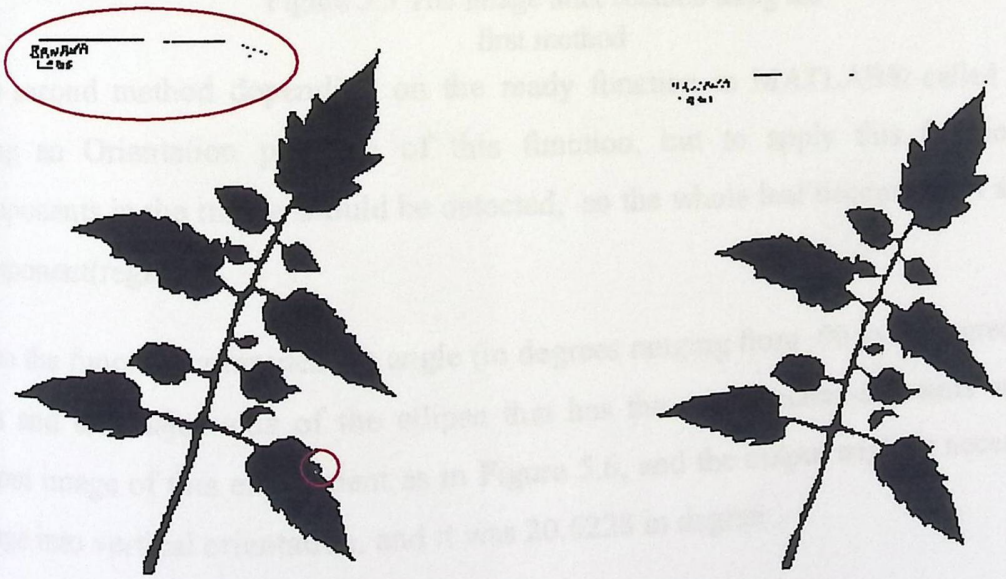


Figure 5.4 Eliminating noise from image using open morphological operation

3. Rotating the leaf into vertical orientation, and finding the center of the leaf.

Two different methods were tested, in order to find the best method to rotate the leaf. The first method depending on the Formula 2-8, and this method considered more difficult to programmer, because it should be programmed entirely by him. After testing this method on the previous image, the result was as shown in Figure 5.5, and the output angle is necessary into rotate the image to vertical orientation, and it was 17.6036 in degree.

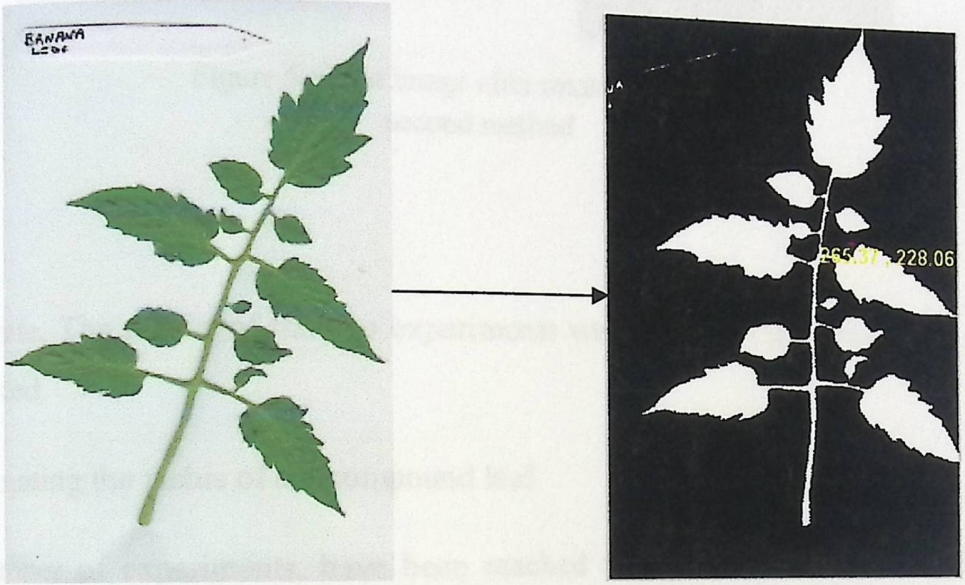


Figure 5.5 The image after rotation using the first method

The second method depending on the ready function in MATLAB® called 'regionprops' by using an Orientation property of this function, but to apply this function the connected components in the image should be detected, so the whole leaf become often a single connected component(region).

Then the function computes the angle (in degrees ranging from -90 to 90 degrees) between the x-axis and the major axis of the ellipse that has the same second-moments as the region. The output image of this experiment as in Figure 5.6, and the output angle is necessary to rotate the image into vertical orientation, and it was 20.6228 in degree.

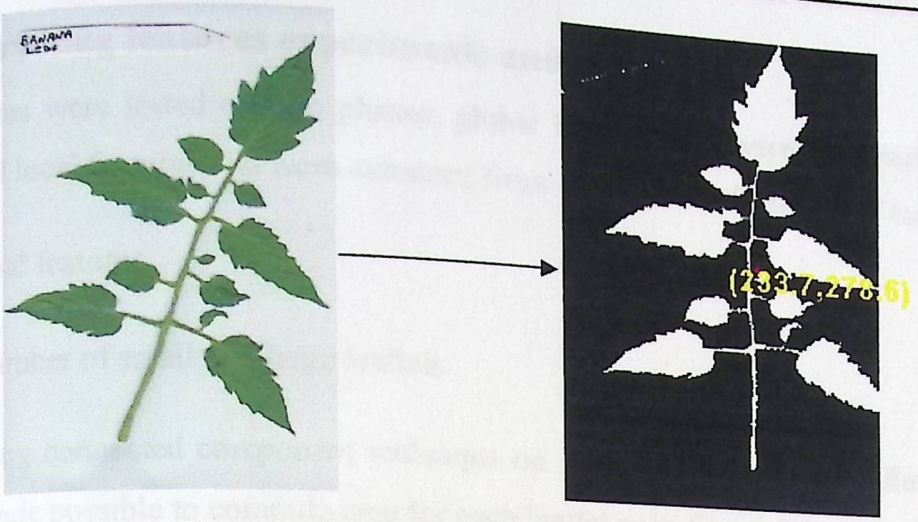


Figure 5.6 The image after rotation using the second method

As illustrate, The results of the two experiments were close. In this project the second method was adopted.

4. Eliminating the rachis of the compound leaf.

After number of experiments, have been reached that the most suitable number of 'dilation' morphological operation was 10 times on our sample to eliminate the rachis. The output image as in Figure 5.7.

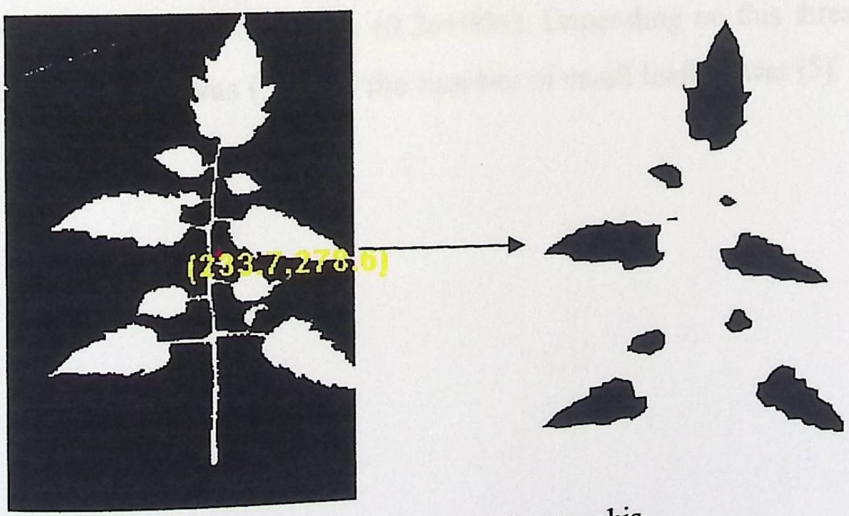


Figure 5.7 The leaf image without rachis

5.4.2 Extracting features experiments and results

The features were tested on two phases; global features that were extracted from a leaf as a whole, and local features that were extracted from each leaflet of compound leaf.

First: Global features

- The number of small and large leaflets

After testing connected component technique on a binary image, each leaflet had a distinctive label, it made possible to compute area for each leaflet as in Figure 5.8.

Figure 5.8 leaflets of compound leaf

After number of experiments have been reached for the most appropriate threshold to determine the area of small and large leaflets was (0.2e+004). Depending on this threshold in Figure 5.8 the number of large leaflets was (5), and the number of small leaflets was (5).

- The distances between the center of leaflet and the center of leaf

After abstracting the center of each leaflet from the center of the leaf the results as in Figure 5.9.

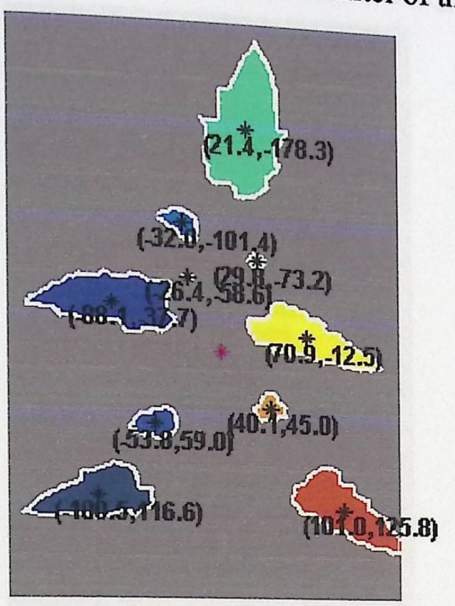


Figure 5.9 The distances between the center of each leaflet and the center of compound leaf.

Second: Local features

- The area and perimeter features were extracted from each leaflet of plant leaf.
- Angles between leaflets and Y- axis

After testing the second method was used to rotate the leaf on each leaflet, the results were as shown in Figure 5.10

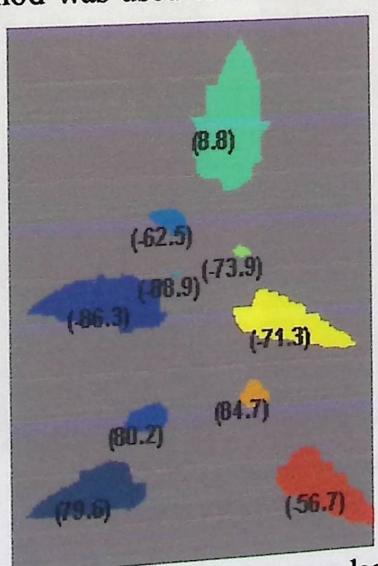


Figure 5.10 The angles between leaflets and Y-axis

- Form factor of each leaflet

After applying Algorithm 4.5, the results appeared as in Figure 5.11

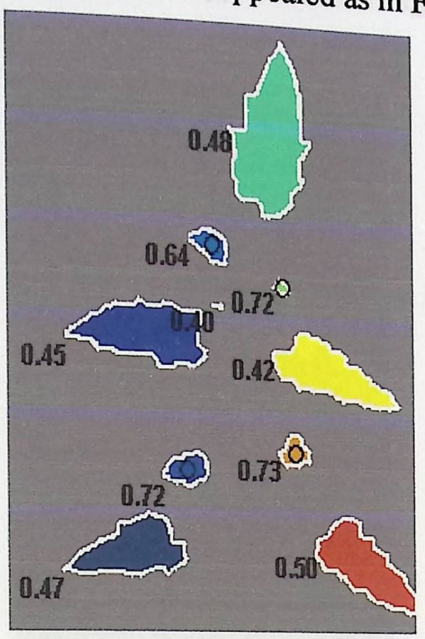


Figure 5.11 The form factor of each leaflet

- The physical length and width of each leaflet

After testing Algorithm 4.6, the results appeared as in Figure 5.12.

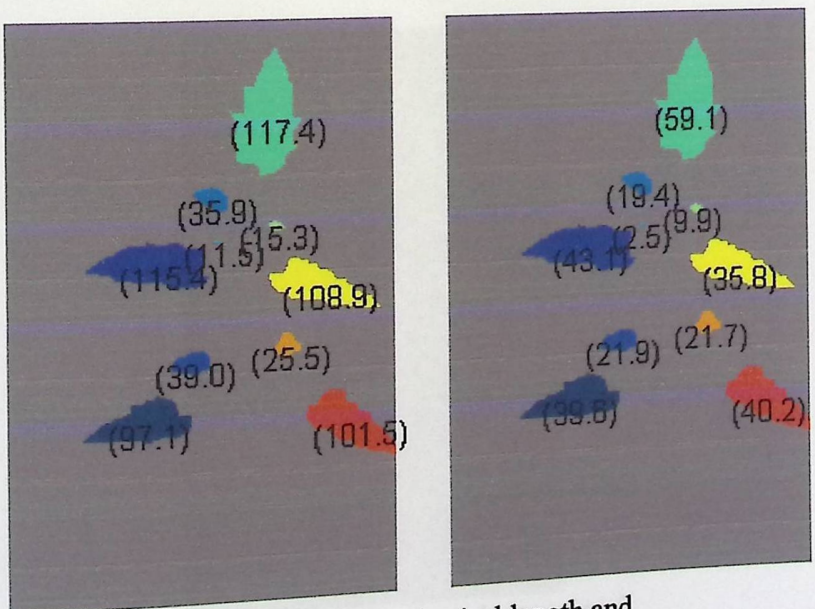


Figure 5.12 The physical length and width of each leaflet

- Aspect ratio of each leaflet

After testing Algorithm 4.7, the results appeared as in Figure 5.13.

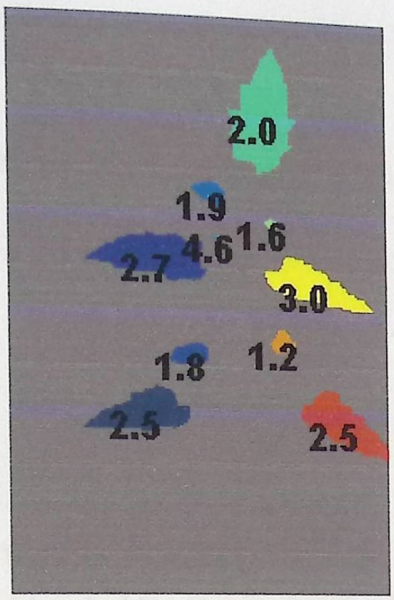


Figure 5.13 Aspect ratio of each leaflet

- The rectangularity of each leaflet

After applying Algorithm 4.8, the results appeared as in Figure 5.14.

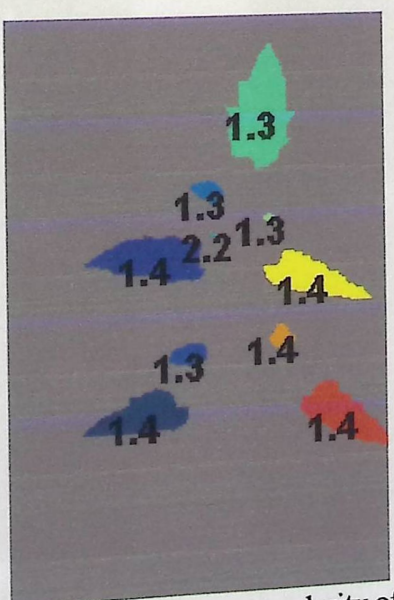


Figure 5.14 The rectangularity of each leaflet

5.4.3 Verifying extracted features experiments and results

First: Matching plant leaves experiments and results

The first steps to implement this process, were choosing the suitable thresholds to determine the points of matching.

After number of experiments, we found the best thresholds were as follow:

- The threshold was used to determine the center matching of leaflets (threshold1 in Chapter 4) was 20.
- The threshold was used to determine the angles matching of leaflets (threshold2 in Chapter 4) was 3.
- The threshold was used to determine the form factor, aspect ratio and rectangularity matching of leaflets (threshold3, threshold4 and threshold5 in Chapter 4) was 0.10

After applying these thresholds in Algorithm 4.9 on two similar leaf images to find the matched leaflets the system showed a perfect matching between the two leaves. For more illustration see Figure 5.15

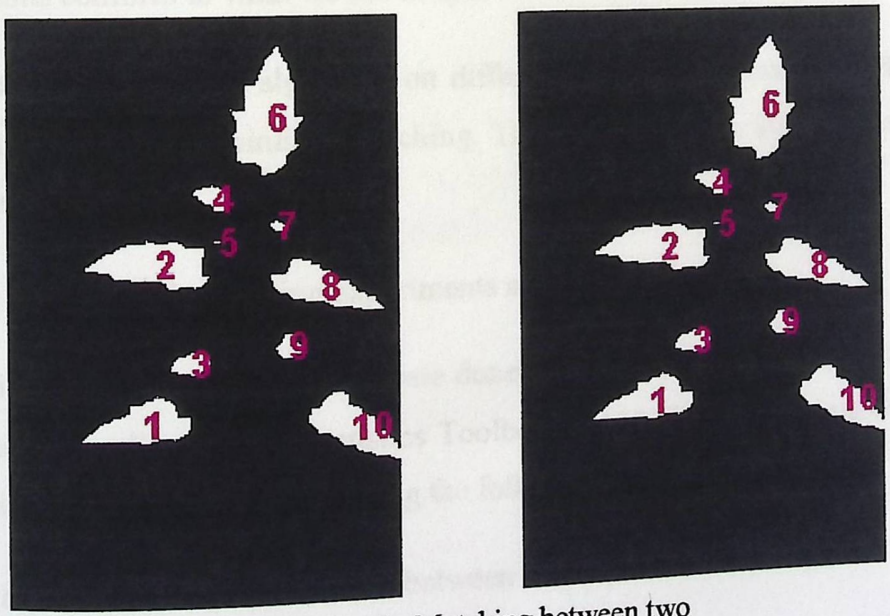


Figure 5.15 Matching between two similar compound leaves

Also, after applying these thresholds in Algorithm 4.9 on two different leaf images to find the matched leaflets the results appeared as in Figure 5.16.

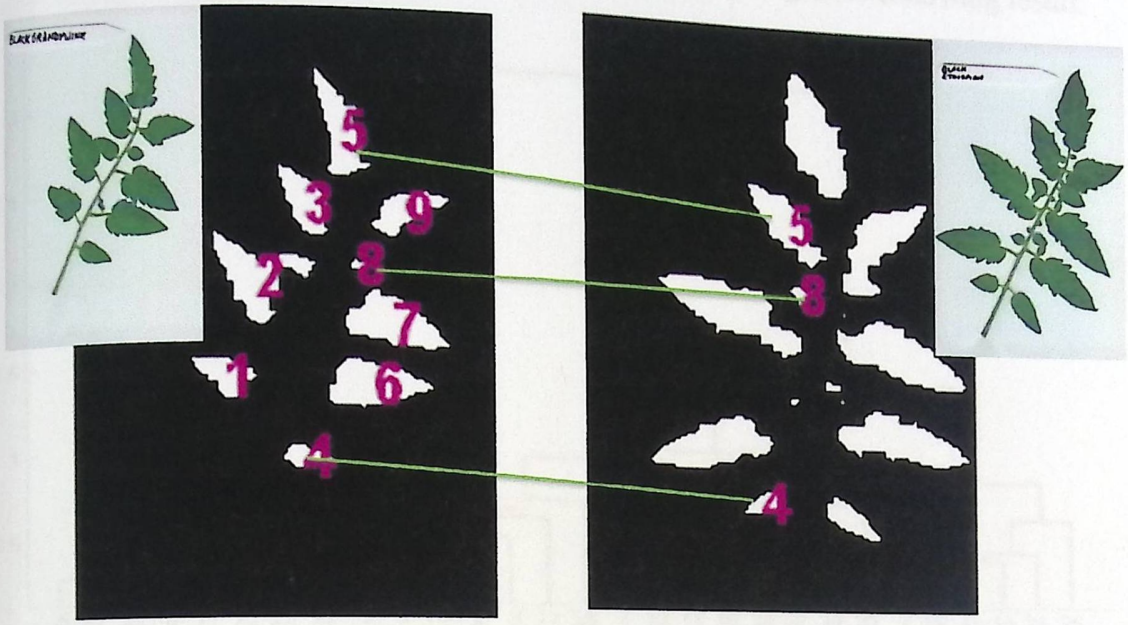


Figure 5.16 Matching between two different compound leaves

The ratio of matching between these two leaves was 0.333 this number was resulting by dividing 3 (number of matched leaflets) on 9 (number of leaflets in the first leaf).

These results conform to what we see in eye.

After testing the matching algorithm on different images, we found that the algorithm gives a good results to find the points of matching. These results mean that the features were extracted in accurate way.

Second: Hierarchical clustering experiments and results

Hierarchical clustering experiments were done on sample contains 29 tomato leaf from different types (see Appendix B). The Statistics Toolbox in MATLAB® was used in this process. The Hierarchical Clustering was done using the following steps:

1. First the Euclidean distances between each pairs of features was computed using the 'pdist' Function.
2. Creating a tree from the calculated distances using 'linkage' function.

3. The output was drawn using the 'dendrogram' function.

After applying the Hierarchical Clustering on the sample, we got the following result:

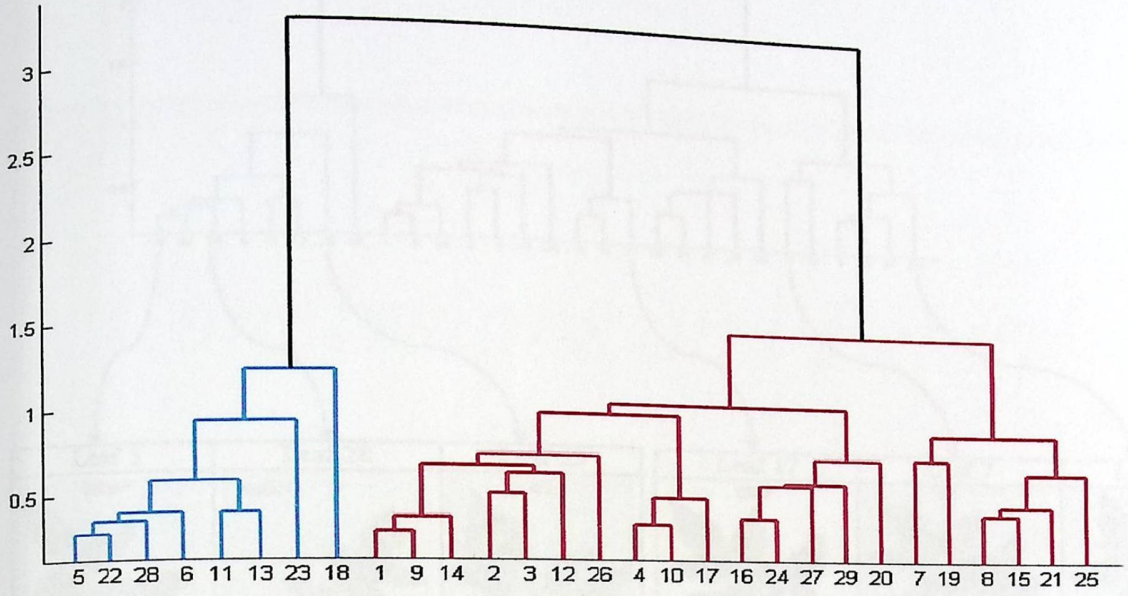


Figure 5.17 Dendrogram represents the clustering of sampled tomato leaves

Figure 5.17 shows the result of applying hierarchical clustering on the sample, this figure shows there are number of clusters in this sample, this number depending on when we want to cut the dendrogram at the proper level.

The result on this figure is agreed to what we see in eye. Figure 5.18 shows an example of the two big clusters (Red and Blue). In this Figure we can see that the leaves in the blue cluster belongs to Potato Leaflet(PL) type that contains smooth edges and there are have a thick leaflets, belongs to Regular Leaflet(RL) type that contains a toothed and symmetrical edges

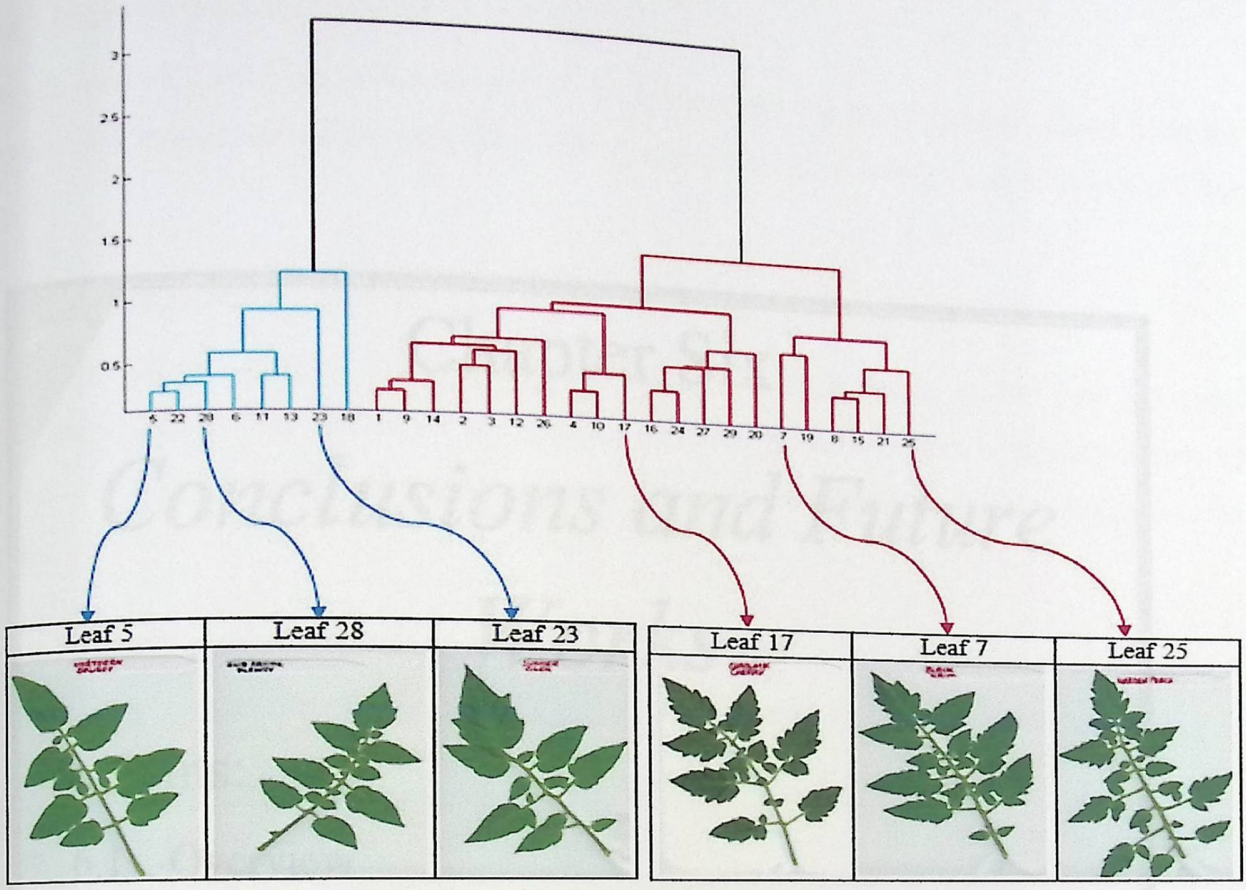


Figure 5.18 Three sample leaves from the two main clusters revealed by dendrogram

The result of clustering confirms that the features that have been extracted in this project play an important role in the representation of the leaf's shape in accurate way, and this result will help in the classification of the plant compound leaves.

5.5 Summary

This chapter contained the results of testing the algorithms were proposed in Chapter 4 , the results of pre-processing image, extracting features, matching plants leaf.

The results were illustrated in this chapter proved that the methodology was used to extracted features from compound leaf gave good results in representing plant compound leaves.

Chapter Six

Conclusions and Future Works

Contents:

6.1 Overview

This chapter will state the achievements of the system and the main problems faced us in the project. The second part of this chapter is talking about the future work proposed to improve this project.

6.2 Conclusions

Extracting features from a plant compound leaf is a new approach in the field of image processing. This project will open the way for studying plants that have compound leaves by introducing a new way of features extraction, and consequently help in plant classification in the future.

6.2.1 Project achievements

In this point the main achievements of our project are discussed and the ways of achieving it.

- Proposing a methodology to extract features from compound leaf

  The project succeed in introducing a methodology to extract features from plants compound leaf, both local and global features depending on image processing techniques. These features describe the leaf in a good way.

- Reducing user intervention to extract features

  The main features in the previous researchs that depending on user were the physiological length and width. In this project these features are no longer dependent in the user, these features were extracted in this project depending on the axes of the ellipse contained the leaf.

- Finding the similarity in groups of plants that have compound leaves

  In this project we used a hierarichial clustering analysis to cluster the plant compound leaves into similarity groups, in order to reduce the size of large numbers of plant's types to help in classification.

- For this project we wrote a paper called 'Using computer vision to cluster plants with compound leaves based on morphological and geometrical features', and we aim at publishing it in the recent future (See Appendix D).

6.2.2 Project problems

Many problems faced us during implementing this project. Most of these problems have been solved and others have been covered here.

- Sample of study

  The main problem of this project was there is no sufficient sample identical to conditions proposed in Chapter 4. This problem has not been able us to start in the working of classification.

- Eliminating the rachis of the leaf, matching leaflets

  When elimenating the rachis in this project some small leaflets disapear, this because the number of 'dilation' morphological process is used to eleminiate the rachis of the leaf depending on the area of the leaf so the number should be dynamic. This project had been applied an appropriate thresould of the sample. This problem is similar to the matching problem related to the thresoulds are used to find the points of matching.

  One suitable suggestion is provided here for solving this problem:

  - Making the number of dilation process changes depending on the area of the leaf, by dividing the area of the leaflet on the leaf area.

- Scale of the images

  The scale of the images effects on the some features such as the area, and the perimeter of the leaflets.

  One suitable suggestion is provided here for solving this problem:

  - Making features depending on these features, so these features will be not used directly

6.3 Future work

In the future many changes could help in obtaining better results; by extracting new set of features such as, smooth factor, narrow factor, vein features, etc to represent the compound leaf accurately, and that reflect other characteristics of the plant.

Also the selection techniques can be used to find the important features and the most effected features in representing the plant leaf, such as Fuzzy selection techniques.

In the future these extracted features can be used in the classification of plants that have compound leaves that represent an important component in human life, by using computer machine for example.

Appendices

Appendix A

Implementation Code

A.1 Extracting features function

The function was used to extract features from compound leaf is illustrated here, this function returns the features of the leaf as an output

```
% This function take the RGB image (I) as input and return the features of
% this image as output, where:
% LeafletArea: the area of the leaflet.
% perimeter: the perimeter of the leaflet.
% leaflet_centerX: The x-coordinate of the center of the leaflet.
% leaflet_centerY: The y-coordinate of the center of the leaflet.
% angle: The angle between the y-axis and the leaflet.
% metric: The form factor of the leaflet.
% aspectRatio: The ratio between the length and width of the leaflet.
% rectangularity: Describes how the leaflet shape differs from the rectangle.
% small:Number of small leaflets in the leaf.

function[leafletArea, perimeter, leaflet_centerX, leaflet_centerY ,angle
,metric ..., aspectRatio ,rectangularity, large, small] = Features(I);


%% 1.Convert image from RGB to binary
 % Convert the image to black and white in order to prepare for the next
   steps.

grayImage=rgb2gray(I);                    % Converts the true color image (I)
                                            to the grayscale intensity image
                                            (grayImage).

threshold = graythresh(grayImage);        % Computes a global threshold
                                            (threshold) that can be used to
                                            convert an intensity image to a
                                            binary image with IM2BW.

BW = im2bw (grayImage, threshold);        % Converts the intensity image
                                            (grayImage) to black and white by
                                            thresholding (threshold).


%% 2.Remove the noise
 % Using morphology functions, remove pixels which do not belong to the
   objects of interest and close the holes in the objects.

se=strel('square',2);                     % Creates a square morphological
                                            structuring element whose width is
                                            2 pixels.

openImage=imopen(BW,se);                  % Performs morphological opening on
                                            the binary image (BW) with the
                                            structuring element se.
```

```
dilateImage=imdilate(openImage,se);        % Dilates the binary image
                                             (openImage), returning the dilated
                                             image (dilateImage).

comImage=~dilateImage;                     % Takes the complement of logical
                                             image (dilateImage)by setting the
                                             0's elements to 1,and 1's element
                                             to 0, to do the next steps.

%% 3.Rotate the leaf image into vertical orientation
% Find the connected components in the images and give each one label,
 %in order to know the label of the whole leaf to rotate it.

[L, num] = bwlabel(comImage);              % Label connected components in
                                             binary image, and return a
                                             matrix(L) of the same size
                                             as(comImage),containing labels for
                                             the connected components in
                                             (comImage),and returns in NUM the
                                             number of connected objects found
                                             in BW.

% Find the large area that belong to the leaf as a whole
for i=1:num
   area(i) = bwarea(L==i);                 % Estimates the area of the  objects
                                             in binary image(comImage), and
                                             stores the results in (area)
                                             matrix.
end

maxarea = max(area);                       % Finds the largest element in
                                             (area) matrix that belong to the
                                             leaf component in tha image.
x=find (area == maxarea);                  % Finds indice of the large element
                                             in (area) matrix, and returns the
                                              result in (x).


% Rotate the leaf into vertical oriantation                                  %
stats = regionprops(L,'Orientation','Centroid' );
       Measures a set of properties for each labeled region in the label
       matrix L. % Positive integer elements of L correspond to different
       regions.Where:
        % 'Orientation' -- Scalar; the angle (in degrees) between the x-
          axis and the major axis of the ellipse that has the same second-
          moments as the region.
        % 'Centroid'-- vector; the center of mass of the region. Note that
          the first element of Centroid the horizontal coordinate (or x-
          coordinate) of the center of mass,and the second element is the
          vertical coordinate (or y-coordinate).
```

```
if stats(x).Orientation < 0

  rotateImage=imrotate(comImage,1*(90+(stats(x).Orientation)),'nearest',...
  'crop');                                          % Rotates the image A by
                                                    angle degrees in a
                                                    counterclockwise direction

else
   rotateImage =imrotate(comImage,90-(stats(x).Orientation),'nearest',...
   'crop');
end


%% 4.Eliminate the rachis
% Using morphological operation (dilation) to eliminate the rachis, the
 dilate operation is done 10 times as a proposed threshold.
roImage=~ rotateImage;
for i=0:10
   roImage =imdilate(roImage,se);
end
rotateImage =~roImage;



%% 5.Global features:
% a.Find the number of leaflet and give each one label
    [L, num] = bwlabel(rotateImage);

    % Find the number of large and small leaflet
    for i=1:num
        area(i) = bwarea(L==i);
    end
    x2 = find( area > 0.2e+004 *1);    % proposed a threshold to determine
                                        the large and small leaflets.

    large=numel(x2);
    small=num-large;

% b.Find the center of the whole leaf
    center= stats(x).Centroid;          % Where the center(1) is the x-
                                        coordinate, and center(2)is the y-
                                        coordinate.


% c.Find the distance between the center of each leaflet from the center of
    the leaf
    % Find the boundry of the leaflets. The boundary of leaflets will be used
      to extract below features
    [B,L] = bwboundaries(rotateImage,'noholes');  % Trace  exterior
                                                    boundaries of objects in
                                                    a binary image only,
                                                    bwboundaries returns B as
                                                    indicator to the number
                                                    of objects, and returns
                                                    the label matrix L.
```

```
stats = regionprops(L,'Centroid');
for k = 1:length(B)
    centroid = stats(k).Centroid;
    centroid(1) = centroid(1) - center(1);  % Abstracts the x-coordinate
                                              center of leaf from the
                                              x_coordinate center of each
                                              leaflet.
    centroid(2) = centroid(2) - center(2);  % Abstracts the y-coordinate
                                              center of leaf
                                            % from the y-coordinate
                                              center of each leaflet.
    leaflet_centerX(k)=centroid(1);
    leaflet_centerY(k)=centroid(2);
end

%% 6.Local features:
Stats= regionprops(L,'Area','Centroid','MajorAxisLength','MinorAxisLength',
'Orientation');
                % Measures a set of properties for each leaflet. where:
                % 'Area'-- Scalar; the actual number of pixels in the
                  leaflet.
                % 'MajorAxisLength'-- Scalar; the length (in pixels) of
                  the major axis of the ellipse that has the same
                  normalized second central moments as the leaflet.
                % 'MinorAxisLength' -- Scalar; the length (in pixels) of
                  the minor axis of the ellipse that has the same
                  normalized second central moments as the leaflet.


% a.Find the Form factor (metric) of leaflets
   for k = 1:length(B)
      boundary = B{k};
      delta_sq = diff(boundary).^2;
      perimeter(k) = sum(sqrt(sum(delta_sq,2)));
      leafletArea(k) = stats(k).Area;
      metric(k) = 4*pi*leafletArea(k)/perimeter(k)^2;
   end


% b.Find the angles of leaflets
   % The angle between the Y_axis and the leaflet.
   for k = 1:length(B)
      angle= stats(k).Orientation;

if angle < 0
         angle2=-1*(90+(stats(k).Orientation));
      else
         angle2= 90-(stats(k).Orientation);
      end
      angle(k)=angle2;
   end
```

```
% c.Find the Aspect ratio
   % This features is extracted depending on the leangth and width of the
   % leaflets
   for k = 1:length(B)
      aspectRatio(k)=stats(k).MajorAxisLength/stats(k).MinorAxisLength;
   end



% d.Rectangularity
   % This feature is extracted depending on the length, width and area of
   % the leaflets
   for k = 1:length(B)

      rectangularity(k)=(stats(k).MajorAxisLength*stats(k).MinorAxisLengt)
      /stats(k).Area;
   end
```

A.2 Clustering Code

The clustering code was used to implement a hierarichical clustering on the sample (29 image) illustrates here:

```
%% Read the leaf image (RGB model) and extracts features:
fid = fopen('C:\Data.txt');                    % Open the Data file that
                                                stores the paths of the leaf
                                                image.

rawFeatures = fopen('C:\test\rowData.xls', 'w' );

% Extract features from 29 leaf.
for P=1:29
   name= fscanf(fid, '%s',1)                   % Read 1 rows each time from
                                                the Data file, convert it to
                                                string and stores it in
                                                (name) variable.

   I=imread(name);                             % Read image and return the
                                                image data in the array (I).

[leafletArea,perimeter,leaflet_centerX,leaflet_centerY ,angle ,metric...
,aspectRatio ,rectangularity,large,small] = Features(I);
                                                % Call Feature function, to
                                                extract features from the
                                                leaf. Send it the image data,
                                                and takes the features for
                                                this leaf.
```

```
K= small + large;

leaflets(P)=k;                                  % Number of leaflets

                                                % Stores the number of leaflets for
                                                    each leaf and stores it in
                                                    (leaflets) array.


% Store the features in feature array
for k = 1:leaflets(P)
    feature(P,k,1)=leafletArea(k);
    feature(P,k,2)=perimeter(k);
    feature(P,k,3)=leaflet_centerX(k);
    feature(P,k,4)=leaflet_centerY(k);
    feature(P,k,5)=angle(k);
    feature(P,k,6)=metric(k);
    feature(P,k,7)= aspectRatio(k);
    feature(P,k,8)=rectangularity(k);
    feature(P,k,9)=large;
    feature(P,k,10)=small;
fprintf(rowData, '%*d\t %d\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t%f\t
%f\n',2,P,k,leafletArea(k),perimeter(k),leaflet_centerX(k),leaflet_centerY(k),
angle(k),metric(k),aspectRatio(k),rectangularity(k),large,small);

end


% Clear the variables to prevent the error in data.
clear perimeter
clear leafletArea
clear metric
clear aspectRatio
clear rectangularity
clear leaflet_centerX
clear leaflet_centerY
clear large
clear small
end                                             % Finishing the read and extract
                                                    the features from 29 leaf
                                                    image.
fclose(fid);                                    % Closes the open file(Data),
                                                    where fid argument is a file
                                                    identifier associated with an
                                                    open file (Data).


fclose(rowData);


%% Prepare data to cluster:
result=fopen('C:\test\newFile.xls', 'w' );      % Open the newFile to store the
                                                    Euclidean distnce of features
                                                % between each two leaflet.
```

```
result2=fopen('C:\test\newFile2.xls', 'w' );% Open the newFile2 to store the
                                              minimum distance between
                                            % each leaflet and other leaves.

% Normalized data
M=load ('C:\test\ rowData.xls');
for z=3:12
normalizedFeatures(:,z)=M(:,z)/ max(M(:,z));
end


v=0;
for i=1:P
    for j=1:leaflets(i)
        for g=3:12
           newFeature(i,j,g-2)=normalizedFeatures(j+v,g);
        end

    end
    v=v+j;
end



% Compare each leaflet in leaf with other leaflet from other leaves.
for n=1:P
    for z=1:leaflets(n)
        for x=1:P
            for k=1:leaflets(x)
                % Compute the Euclidean distance between features of each
                  two leaflet,
                % and stores the result in (newFile).
                MSE1=(newFeature(n,z,1)-newFeature(x,k,1))^2;
                MSE2=(newFeature(n,z,2)-newFeature(x,k,2))^2;
                MSE3=(newFeature(n,z,3)-newFeature(x,k,3))^2;
                MSE4=(newFeature(n,z,4)-newFeature(x,k,4))^2;
                MSE5=(newFeature(n,z,5)-newFeature(x,k,5))^2;
                MSE6=(newFeature(n,z,6)-newFeature(x,k,6))^2;
                MSE7=(newFeature(n,z,7)-newFeature(x,k,7))^2;
                MSE8=(newFeature(n,z,8)-newFeature(x,k,8))^2;
                MSE9=(newFeature(n,z,9)-newFeature(x,k,9))^2;
                MSE10=(newFeature(n,z,10)-newFeature(x,k,10))^2;

Avg=sqrt(MSE1+MSE2+MSE3+MSE4+MSE5+MSE6+MSE7+MSE8+MSE9+MSE10);


fprintf(result, '%*d\t %d\t %d\t %d\t %d\t %f\t %f\t %f\t %f\t %f\t %f\t
%f\t %f\t %f\t% f\n',2,n,z,x,k,MSE1,MSE2,MSE3,MSE4,MSE5,MSE6, MSE7, MSE8,
MSE9,MSE10, Avg);

                MSE(k)=Avg;
```

```
            end

            % Find the minimum distance between the leaflet in leaf and
              other leaves.
            minAvg(x)=min(MSE);
            clear MSE;

            % Store the results in newFile2.
            fprintf(result2, '%*d\t %d\t %d\t %f\n',2,n,z,x, minAvg(x));
        end
    end
end


% Close the opened files
fclose(result);
fclose(result2);



% To Find the minimum distance betwwn each leaves, and stores the results
% in (newFile3)
Arr=load ('C:\test\newFile2.xls');        % Load the newFile2 into (Arr)
                                            matrix
result3 = fopen('C:\test\newFile3.xls', 'w' );

c=1;e=c;d=0;
for x=1:P
    for j=1:P
        for i=1:leaflets(x)
            A2(i)= Arr(c,4);
            c=c+P;
        end
        v=min(A2);
        fprintf(result3, '%*d\t %d\t %f\n',2,x,j,v);
        clear A2;
        c=e+j;
    end
    d=d+ leaflets(x); c=d*P; c=c+1;e=c;
end
fclose(result3);



% Prepare a matrix for cluster process, and stors it in newFile4.
B=load ('C:\test\newFile3.xls');
result4 = fopen ('C:\test\newFile4.xls', 'w' );
c2=1;
for i=1:P
    for j=1:P
        B2(i, j)=B(c2,3);
        c2=c2+1;
        fprintf(result4, '%f\t', B2(i,j));
```

```
    end
    fprintf(result4, '\n');
end
fclose(result4);


% To Do the cluster analysis and draw a dendogram.
C=load ('C:\test\newFile4.xls');     % Load the newFile4 to C array.

Y = pdist(C);                        % computes the Euclidean distance between
                                       pairs of objects in n-by-p data matrix
                                       C.
Z = linkage(Y, 'average');           % Creates a hierarchical cluster tree
                                       from the distances in y using UPGMA
                                       method.


H = dendrogram(Z,0,'colorthreshold','default');    % generates a dendrogram
                                                     plot of the hierarchical,
                                                     binary cluster tree
                                                     represented by Z, and
                                                     assigns a unique color to
                                                     each group of nodes.
```

A.3 Matching Code

```
function varargout = Matching_interface(varargin)
% MATCHING_INTERFACE M-file for Matching_interface.fig
%      MATCHING_INTERFACE, by itself, creates a new MATCHING_INTERFACE or
raises the existing
%      singleton*.
%
%      H = MATCHING_INTERFACE returns the handle to a new MATCHING_INTERFACE
or the handle to
%      the existing singleton*.
%
%      MATCHING_INTERFACE('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in MATCHING_INTERFACE.M with the given input
arguments.
%
%      MATCHING_INTERFACE('Property','Value',...) creates a new
MATCHING_INTERFACE or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Matching_interface_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Matching_interface_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Matching_interface

% Last Modified by GUIDE v2.5 08-May-2010 23:31:04

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Matching_interface_OpeningFcn, ...
                   'gui_OutputFcn',  @Matching_interface_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before Matching_interface is made visible.
function Matching_interface_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Matching_interface (see VARARGIN)

% Choose default command line output for Matching_interface
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Matching_interface wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Matching_interface_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in image1_axes.
function image1_Callback(hObject, eventdata, handles)
% hObject    handle to image1_axes (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
image_file = get(handles.edit1,'String');
global I;
I=imread(char(image_file));
set(handles.image1_axes,'HandleVisibility','ON');
axes(handles.image1_axes);
image(I);
axis equal;
axis tight;
axis off;



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a
double
```

```
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in image2.
function image2_Callback(hObject, eventdata, handles)
% hObject    handle to image2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
image_file = get(handles.edit2,'String');
global I2;
I2=imread(char(image_file));
set(handles.image2_axes,'HandleVisibility','ON');
axes(handles.image2_axes);
image(I2);
axis equal;
axis tight;
axis off;
```

```
% --- Executes on button press in matching.
function matching_Callback(hObject, eventdata, handles)
% hObject    handle to matching (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%% For first image
   global I;

    grayImage=rgb2gray(I);
    threshold = graythresh(grayImage);
    BW = im2bw(grayImage,threshold);


   % Make open operation to eliminate noise
    se=strel('square',2);
    openImage=imopen(BW,se);
    dilateImage=imdilate(openImage,se);
    comImage=~dilateImage;



   % Find the connected componenet in the images
    [L, num] = bwlabel(comImage);


   % Find the large area
    for i=1:num
        area(i) = bwarea(L==i);
    end
    maxarea=max(area);
    x=find( area ==  maxarea);


   % Rotate the leaf into vertical oriantation
    stats = regionprops(L,'Orientation', 'Centroid' );



   if stats(x).Orientation < 0
roImage=imrotate(comImage,1*(90+(stats(x).Orientation)),'nearest','crop');

   else
      roImage=imrotate(comImage,90-
(stats(x).Orientation),'nearest','crop');
   end



   % Find the center of the whole leaf
     center= stats(x).Centroid;
```

```
% Eliminate the rachis
roImage2=~roImage;
for i=0:10
    roImage2=imdilate(roImage2,se);
end
roImage=~roImage2;



% Find the number of leaflet and give each one label
  [L, num] = bwlabel(roImage);


% Find the number of large and small leaflet
for i=1:num
    area(i) = bwarea(L==i);
end
x2 = find( area > 0.2e+004 *1);

large=numel(x2);
small=num-large;


%set(handles.uipanel2 ,'Visible','ON');
%set(handles.f2 ,'Visible','ON');
%set(handles.f3 ,'Visible','ON');
%set(handles.edit3,'String', small ,'Visible','ON');
%set(handles.edit4,'String', large ,'Visible','ON');


% Find the boundry of the leaflets
[B,L] = bwboundaries(roImage,'noholes');
for k = 1:length(B)
    boundary = B{k};
end




    stats =
regionprops(L,'Area','Centroid','MajorAxisLength','MinorAxisLength','Orientati
on');

% Find the center of each leaflet
for k = 1:length(B)
    centroid = stats(k).Centroid;

    centroid(1) = centroid(1) - center(1);
    centroid(2) = centroid(2) - center(2);
```

```
    centerx(k)=centroid(1);
    centery(k)=centroid(2);

end


% Angles of leaflets
for k = 1:length(B)
    angle= stats(k).Orientation;
    if angle < 0
       angle2=-1*(90+(stats(k).Orientation));

    else
       angle2= 90-(stats(k).Orientation);
    end
    angle(k)=angle2;


end


% Find the Form factor of leaflets
for k = 1:length(B)
    boundary = B{k};

end


% loop over the boundaries
for k = 1:length(B)
     boundary = B{k};
     delta_sq = diff(boundary).^2;
     perimeter(k) = sum(sqrt(sum(delta_sq,2)));
     area2(k) = stats(k).Area;
     metric(k) = 4*pi*area2(k)/perimeter(k)^2;
     metric_string = sprintf('%2.2f',metric(k));
end


% Find the Aspect ratio
for k = 1:length(B)
  Aspect(k)=stats(k).MajorAxisLength/stats(k).MinorAxisLength;
end


% Rectangularity
 for k = 1:length(B)
Rectangularity(k)=(stats(k).MajorAxisLength*stats(k).MinorAxisLength)/stats(k)
.Area;
    end
```

```
        % Store the features
        for k = 1:length(B)


A(1,k,1)=area2(k);A(1,k,2)=perimeter(k);A(1,k,3)=centerx(k);A(1,k,4)=centery(k
);A(1,k,5)=angle(k);A(1,k,6)=metric(k);A(1,k,7)=
Aspect(k);A(1,k,8)=Rectangularity(k);
        end



%% For second image
global I2;

        grayImage2=rgb2gray(I2);
        threshold2 = graythresh(grayImage2);
        BW2 = im2bw(grayImage2,threshold2);

        se=strel('square',2);
        openImage2=imopen(BW2,se);
        dilateImage2=imdilate(openImage2,se);
        dilateImage3=~ dilateImage2;


       % Find the connected componenet in the images
       [L2, num2] = bwlabel( dilateImage3);


        % Find the large area
        for i=1:num2
            area3(i) = bwarea(L2==i);
        end
        maxarea2=max(area3);
        x2=find( area3 ==  maxarea2);


        % Rotate the leaf into vertical oriantation
        stats2 = regionprops(L2,'Orientation', 'Centroid' );

        if stats2(x2).Orientation < 0
            roImage2=imrotate( dilateImage3,-
1*(90+(stats2(x2).Orientation)),'nearest','crop');

        else
          roImage2=imrotate( dilateImage3,90-
(stats2(x2).Orientation),'nearest','crop');

        end

       % Find the center of the whole leaf
       center2= stats2(x2).Centroid;
```

```
% Eliminate the rachis
 roImage3=~roImage2;

for i=0:10
    roImage3=imdilate(roImage3,se);
end


roImage2=~roImage3;

% Find the number of leaflet and give each one label
[L2, num2] = bwlabel(roImage2);



% Find the number of large and small leaflet
for i=1:num2
    area2(i) = bwarea(L2==i);
end
x3 = find( area2 > 0.2e+004 *1);
large2=numel(x3);
small2=num2-large2;

% Find the boundry of the leaflets
[B2,L2] = bwboundaries(roImage2,'noholes');

for k2 = 1:length(B2)
   boundary2 = B2{k2};
end


global stats2;
stats2 =
regionprops(L2,'Area','Centroid','MajorAxisLength','MinorAxisLength','Orientat
ion');


% Find the center of each leaflet

for k2 = 1:length(B2)
   centroid2 = stats2(k2).Centroid;

   centroid2(1) = centroid2(1) - center2(1);
   centroid2(2) = centroid2(2) - center2(2);


    centerx2(k2)=centroid2(1);
    centery2(k2)=centroid2(2);

end
%  Angles of leaflets
 for k2 = 1:length(B2)
    angle3= stats2(k2).Orientation;
```

```
        if angle3 < 0
            angle4=-1*(90+(stats2(k2).Orientation));

        else
            angle4= 90-(stats2(k2).Orientation);
        end

        angle3(k2)=angle4;

    end



    % Find the Form factor of leaflets

    for k2 = 1:length(B2)
        boundary2 = B2{k2};
    end
    for k2 = 1:length(B2)
        boundary2 = B2{k2};
        delta_sq2 = diff(boundary2).^2;
        perimeter2(k2) = sum(sqrt(sum(delta_sq2,2)));
        area3(k2) = stats2(k2).Area;
        metric2(k2) = 4*pi*area3(k2)/perimeter2(k2)^2;
    end


    % Find the Aspect ratio
    for k2 = 1:length(B2)
        Aspect2(k2)=stats2(k2).MajorAxisLength/stats2(k2).MinorAxisLength;
    end


    % Rectangularity
    for k2 = 1:length(B2)

Rectangularity2(k2)=(stats2(k2).MajorAxisLength*stats2(k2).MinorAxisLength)/st
ats2(k2).Area;
    end

    % Store the features
    for k2 = 1:length(B2)


A2(2,k2,1)=area3(k2);A2(2,k2,2)=perimeter2(k2);A2(2,k2,3)=centerx2(k2);A2(2,k2
,4)=centery2(k2);A2(2,k2,5)=angle3(k2);A2(2,k2,6)=metric2(k2);A2(2,k2,7)=
Aspect2(k2);A2(2,k2,8)=Rectangularity2(k2);


    end
 %% Find matching
c=0;
subplot(1,2,1, 'Parent',handles.Matching_image), imshow(roImage)
```

```
hold on
for k3 = 1:length(B)
centroid = stats(k3).Centroid;

                interString = sprintf('%d', k3);
                   text(centroid(1),centroid(2), interString
,'Color','m',...
                     'FontSize',14,'FontWeight','bold');



  end

  hold off


subplot(1,2,2, 'Parent',handles.Matching_image),imshow(roImage2)
hold on
for k3 = 1:length(B)

   for k2 = 1:length(B2)
      if A2(2,k2,3)- A(1,k3,3) < 40  && A2(2,k2,3)- A(1,k3,3) > -40 &&
A2(2,k2,4)- A(1,k3,4) < 40  && A2(2,k2,4)- A(1,k3,4) > -40
       if  A2(2,k2,5)- A(1,k3,5) < 3 && A2(2,k2,5)- A(1,k3,5) > -3
           if  A2(2,k2,6)- A(1,k3,6) < 0.1 && A2(2,k2,6)- A(1,k3,6) > -0.1
               if  A2(2,k2,7)- A(1,k3,7) < 0.1 && A2(2,k2,7)- A(1,k3,7) > -
0.1
                   if  A2(2,k2,8)- A(1,k3,8) < 0.1 && A2(2,k2,8)- A(1,k3,8)
> -0.1
                      A3(k3)=k2;
                      c=c+1;
                      centroid2 = stats2(k2).Centroid;
                      interString = sprintf('%d', k3);
                      text(centroid2(1),centroid2(2), interString
,'Color','m',...
                                'FontSize',14,'FontWeight','bold');

                   end
               end


           end

       end

      end


   end
end
if k2 < k3
  set(handles.ratio1 ,'Visible','ON');
  set(handles.ratio2,'String', c/k2 ,'Visible','ON');
```

```
end


if k3 <= k2
 set(handles.ratio1 ,'Visible','ON');
set(handles.ratio2,'String', c/k3 ,'Visible','ON');
end


% --------------------------------------------------------------------
function Open_Callback(hObject, eventdata, handles)
% hObject    handle to Open (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
file = uigetfile('*.fig');
if ~isequal(file, 0)
   open(file);
end


% --------------------------------------------------------------------
function Print_Callback(hObject, eventdata, handles)
% hObject    handle to Print (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
printdlg(handles.figure1)

% --------------------------------------------------------------------
function File_Callback(hObject, eventdata, handles)
% hObject    handle to File (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Close_Callback(hObject, eventdata, handles)
% hObject    handle to Close (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

delete(handles.figure1)
```

Appendix B

Sample of Study

In Chapter 4, we talked about an online database that contains (111) compound leaf images, but in this project only 29 images were taken to apply this project, this related to many reasons, some of these reasons are:

- Some images in the online database have heavy leaflets, so some leaflets are overlapping, and we could not determine the correct moment of inertia to rotate the image.
- Some images have a very large or small scaling in the area, so it causes problems in applying.

So we used only 29 images from this database as a sample of study, these images are identical to conditions were assumed in Chapter 4. These images are illustrated in the below table.

				
Image 1 : Aunt Ruby's German Green	Image 2: Banana Lege	Image 3: Black Brandywine	Image 4: Black Etniopian	Image 5:Northern Deugnt

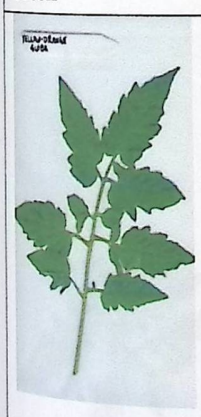
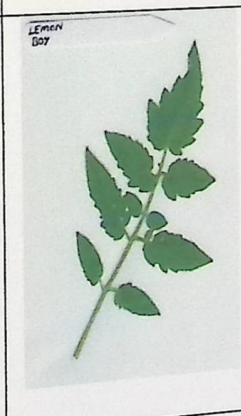
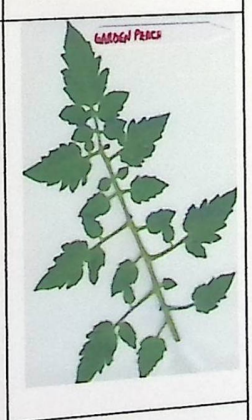
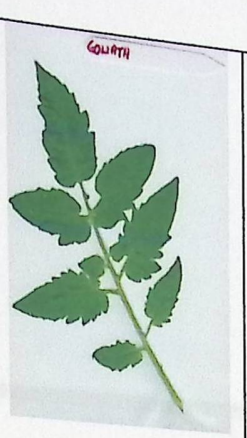



Image 6: Yellow Orange Globe	Image 7: Black Krim	Image 8: Blizzard	Image 9: Boxcar Willie	Image 10: Blonde Kuphchen

				
Image 11: Mister Stripey	Image 12: Brown Berry	Image 13: Mini Toms	Image 14: Bulgarian Triumph	Image 15: Lutchrist Zurich
				
Image 16: Light Pink Oxheart	Image 17: Chadwick Cherry	Image 18: Chebella Marzano	Image 19:Cherokee Chocolate	Image 20: Cherokee Purple
				
Image 21: Crimson Cushion	Image 22: Sub Arctic Plenty	Image 23: Summer Cider	Image 24: Lemon Boy	Image 25: Garden Peach

				
Image 26: Gigante Liscio	Image 27: Harbinger	Image 28: Red Satin	Image 29: Goliath	

Appendix C

The Resulting Tables

The below tables contains a sample of features were extracted in this project.

Leaf	Leaflet	leaflet_Area	Perimeter	leaflet_Xcenter	leaflet_Ycenter	angle	form factor	aspecr ratio	rectangularity	large leflets	small leflets
1	1	4360	393.5219	-72.872726	32.41956	-85.4548	0.353801	2.799399	1.388834	5	3
1	2	1778	232.7107	-76.578163	134.50902	-85.4706	0.412581	3.661701	1.339472	5	3
1	3	3316	311.9239	-50.136582	-39.472215	-61.4792	0.428279	2.785594	1.428018	5	3
1	4	6604	441.5219	-11.468878	-130.41503	-7.13742	0.425709	1.878726	1.353039	5	3
1	5	3995	346.1665	60.757733	-81.031989	39.97162	0.418945	2.159668	1.420626	5	3
1	6	183	70.62742	26.811303	-19.741771	-51.2426	0.461014	2.7796	1.544817	5	3
1	7	5124	406.5929	79.827697	41.963537	62.6891	0.389493	2.277786	1.411798	5	3
1	8	1670	225.4386	54.006015	191.143272	-83.3251	0.412924	2.146333	1.565024	5	3
2	1	2713	270.0244	-100.496383	116.633424	79.63456	0.467578	2.453912	1.414842	5	5
2	2	3635	317.4386	-88.14689	-37.686944	-86.2522	0.453309	2.677691	1.367262	5	5
2	3	655	106.7696	-53.7886	58.96152	80.24923	0.722032	1.782675	1.299395	5	5
2	4	516	100.7696	-31.975836	-101.374863	-62.5153	0.638559	1.853216	1.347731	5	5
2	5	13	20.24264	-26.431114	-58.566724	-88.8805	0.398675	4.614925	2.203914	5	5
2	6	5197	367.7229	21.353318	-178.258084	8.776018	0.482971	1.98452	1.335698	5	5
2	7	113	44.38478	29.762896	-73.150795	-73.9047	0.720809	1.55673	1.337816	5	5
2	8	2767	286.5097	70.924312	-12.499864	-71.31	0.423585	3.038986	1.409204	5	5
2	9	408	83.69849	40.102823	44.96759	84.65825	0.731871	1.17486	1.357334	5	5
2	10	2970	273.8234	100.968757	125.848091	-56.6593	0.497766	2.522561	1.374246	5	5

The below table contains a sample of distance matrix of this project

Leaf \ Leaf	1	2	3	4	5	6	7	8	9
1	0	0.334705	0.319235	0.410713	0.863281	0.814043	0.416498	0.527452	0.144423
2	0.334705	0	0.247389	0.328777	0.761005	0.798266	0.441503	0.37305	0.26672
3	0.319235	0.247389	0	0.416879	0.783869	0.828639	0.312718	0.473554	0.246168
4	0.410713	0.328777	0.416879	0	1.02184	1.037944	0.45753	0.422177	0.431012
5	0.863281	0.761005	0.783869	1.02184	0	0.236609	0.819047	0.767886	0.836784
6	0.814043	0.798266	0.828639	1.037944	0.236609	0	0.730457	0.800643	0.806316
7	0.416498	0.441503	0.312718	0.45753	0.819047	0.730457	0	0.316263	0.356743
8	0.527452	0.37305	0.473554	0.422177	0.767886	0.800643	0.316263	0	0.518085
9	0.144423	0.26672	0.246168	0.431012	0.836784	0.806316	0.356743	0.518085	0

Appendix D

Paper

Using computer vision to cluster plants with compound leaves based on morphological and geometrical features

Duaa Abu Maizer[1 *], Haneen Tartory[1 *], Asma Idais[1 *], Dr.Rami Arafeh[2] and Dr. Hashem Tamimi[2]

[1] Information Technology Department, Palestine Polytechnic University, Jabal Abu Roman, P.O.198,Hebron, Palestine.

[2] Biotechnology Training and Research Unit, Palestine Polytechnic University, Jabal Abu Roman, P.O.198,Hebron, Palestine.

Abstract. Extraction features from plant leaf represents a challenging problem in image processing. Many researchers extracted features from plant leaf to help in plant classification and also in early diagnosis of certain plant diseases. Many previous researches emphasized on extracting features from simple leaf using image processing techniques. This paper aims at proposing a methodology to extract features from a compound leaf using image processing, which is used for clustering plants have compound leaves into similarity groups, this will help to know the plants that have similarity features. The experiment results show that the proposed methodology of extracting features from a compound leaf can be used to cluster the plant compound leaf into similarity groups.

Index Terms - Extraction features, clustering, compound leaf.

I. Introduction

Plants represent an important component in human life. Recent advances in computer technology open new avenues in many life sciences including plant biology (Botany). Computer recognition of plant types and discrimination between plants by computer feature extraction are gaining more attention among scientists.

Many plants carry significant information for humans, features of plant can be extracted from different parts of the plant such as fruits, flowers, roots or plant leaves, and it could have many implications for example in plant taxonomy or in the early diagnosis of certain plant diseases.

* Equally contribution

Recently, some studies focused on extracting certain features from the plant depending on its leaves which is considered as an obvious feature using image processing techniques.

Nevertheless, the previous work in literature showed that there are many works on a simple leaf of different plant species, such as [1], [2] and [3]; on the other hand, few studies have focused on using a compound leaf, which is considered to be more complicated.

This paper aims at proposing a methodology to extract features from compound leaves using image processing techniques. In this paper, the main features of a compound leaf are proposed, and as a case study, tomato leaf is suggested because it appears in many types depending on the variety of fruit shape and size.

These extracted features are used in this paper to cluster plants that have compound leaves into similarity groups based on features; this will help the interested researchers in the plant field to know the similarity plants that have the similar features, also the clustering can be used to reduce the size of large numbers of plant's types to help in classification.

The proposed methodology uses digital images of a compound leaf, then using image processing to pre-process the images of the leaf to isolate the main leaf of tomato from other parts of leaf, and then to extract the proposed morphological and geometrical features from a plant compound leaf image. After that the hierarchal clustering is used to do the cluster process.

This paper implements the extracting features methodology with minimum user intervention, and gets benefits for the geometrical and morphological features from the previous works, such as the features have been proposed by *Wu, et al.* 2006 [1]. Our main improvements are on feature extraction from plants have compound leaf and clustering them into similarity groups.

The rest of this paper is organized as follows. Sec. II discusses image pre-processing. Sec. III introduces how the proposed features are extracted. Hierarchal clustering of plant compound leaf is discussed in Sec. IV. Experimental results are given in Sec. V. Sec. VI concludes this paper.

II. Image Pre-processing

A. Converting RGB image to Binary

The leaf image is acquired by scanner or digital camera. All leaf images are in 800 x 600 resolution, there is no restriction on the direction of leaves, the image should be a complete leaf of plant and not leaflets,

the leaflets of compound leaf should not be overlap, the image should be in the RGB format, and the leaves should be isolated from any background.

An RGB image is firstly converted into a grayscale image.

Then, to convert the grayscale images into binary images, a given threshold should be used in order to separate an object in the image from the background.

B. Reducing the noise in the image.

The open morphological operation is performed on the binary image of the leaf, to remove the pixels which do not belong to the leaf of interest, and to close the holes in the leaf image. The square morphological structuring element whose width is two pixels is used.

C. Rotating the leaf into vertical orientation.

Rotating the leaf into vertical orientation is done in this paper by finding the center of the gravity, and then moment of inertia of the leaf is determined.

Determination of the Center of Gravity: For a leaf surface is described by function f(m, n), consisting of N pixels, the Center of Gravity coordinates (m*, n*) can be calculated as Formula (1)

$$m* = \frac{1}{N} \sum_{(m,n)\in R} \sum m \qquad n* = \frac{1}{N} \sum_{(m,n)\in R} \sum n \qquad (1)$$

Moments of Inertia: the moments of inertia for binary image defined with respect to the center of gravity of the leaf as Formula (2)

$$\mu_{p,q} = \sum_{I} \sum_{J} (i - m*)^p \ (j - n*)^q \qquad (2)$$

Where $p, q = 0,1,2.$

Then, the leaf oriented which is determined by using the following functions:

$$I(\theta) = \sum \sum [(n - n*)\cos(\theta) - (m - m*)\sin(\theta)]2 \qquad (3)$$

Where the I is array ofθ. Resulting in following angleθ:

$$\theta = \frac{1}{2}\tan^{-1}\left[\frac{2.\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}\right] \quad (4)$$

D. Eliminate the rachis of the compound leaf

The dilation morphological operation is used many times on binary image to eliminate the rachis of the compound leaf. The number of dilation process depending on constant number in this paper.

An example of image pre-processing is illustrated in Figure 1.

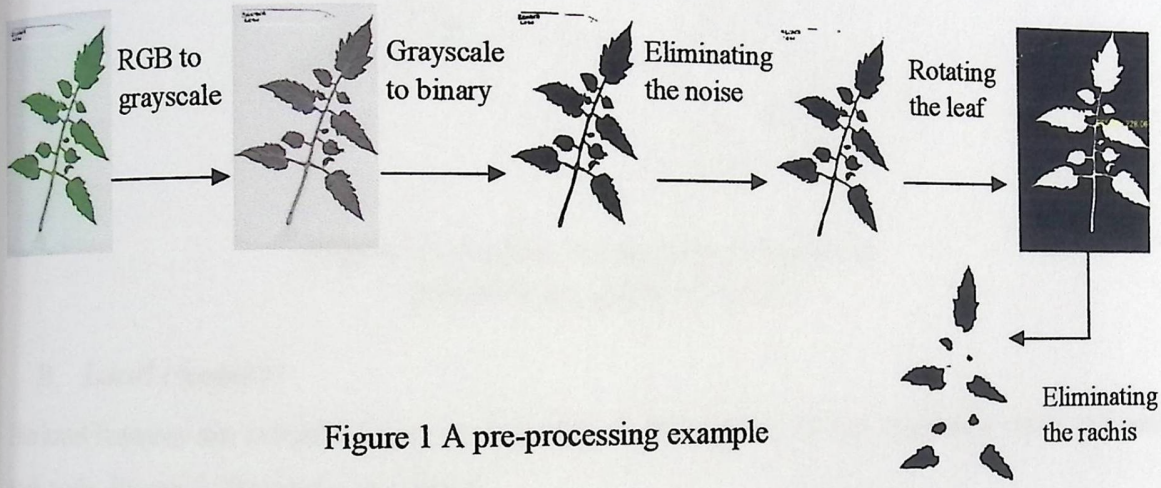


Figure 1 A pre-processing example

III. Features Extraction

In this paper, several morphological and geometrical features are extracted from the compound leaf as a whole, and from each leaflet in the compound leaf using image processing. These features are very important for the compound leaf and these provide significant information about its visual representation. In this paper, we have defined two types of features:

A. Global Features

Global features which are extracted from the compound leaf as a whole and they have two components: the number of leaflets and the general structure of the plant.

1. The number of small and large leaflets:

To find the number of leaflets in an image, the connected component technique will be used, because after applying the technique each leaflet will be a connected component.

To make a more distinctive feature, we can count the number of small and large leaflets from the connected components. This feature should be extracted by knowing the area of each leaflet, and by using a good threshold that helps to determine the area of small and large leaflets.

2. The general structure of the plant leaf:

After the leaf is rotated into the vertical orientation, the general structure of the plant leaf can be achieved. Firstly, the center of each leaflet should be detected. Then, the structure of leaf is built by establishing a graph of the leaflets. In Figure 2 each node represents the distance between the leaflet's center and the leaf's center.

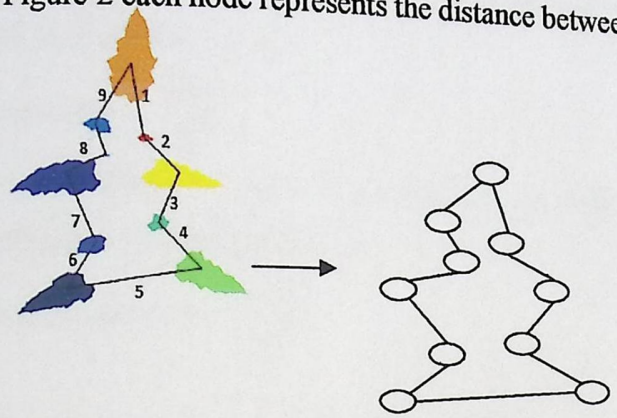


Figure 2 extracting the structure of the leaf by establishing a graph of leaflets

B. *Local Features*

The local features are extracted from each leaflet. From Figure 2 we can establish a vector of features for each node. Figure 3 illustrates the graph.

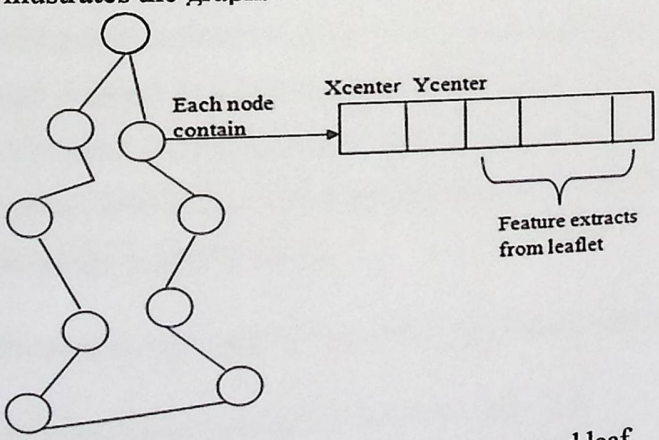


Figure 3 Graph for features of compound leaf

The vector contains some features are:

1. *Leaflet Area (A):* leaflet area is calculated by counting the number of pixels of binary value 1 on smoothed leaf image.

2. *Leaflet Perimeter (P):* leaflet perimeter is calculated by counting the number of pixels consisting leaf margin.
3. *Aspect ratio:* the aspect ratio is defined as the ratio of physiological length (L_P) to physiological width (W_P), which are calculated depending on the axes of ellipse contained the leaflets, it is defined as Formula (2) [1]

$$\text{Aspect ratio} = \frac{L_P}{W_P} \quad (2)$$

4. *Form factor:* the form factor is used to describe how the shape of the leaflet is different from a circle, it is defined as Formula (3) [1]

$$\text{Form factor} = \frac{4\pi A}{P^2} \quad (3)$$

5. *Rectangularity:* rectangularity is used to describes how the shape of the leaflet is different from a rectangle, it is defined as Formula (4) [1]

$$\text{Rectangularity} = \frac{L_P\, W_P}{A} \quad (4)$$

6. *Angle between leaflet and Y- axis:* is calculated using the same method to find the rotation angle of the leaf.

IV. Hierarchal Clustering of Plant Compound Leaf

Clustering plant is an important process in the plant taxonomy; it is used to reduce the large number of plant's types of specific plant compound leaf to help in classification, and it can be used to find the similarity in the groups of plants that have compound leaves [5]. In this paper a hierarchal clustering is used to achieve the clustering process because it may correspond to meaningful taxonomies, and do not have to assume any particular number of clusters, any desired number of clusters can be obtained by cutting the dendogram at the proper level [6].

In this paper an agglomerative approach of hierarchal clustering is used to cluster the plant leaves.

The Hierarchical Clustering is done using the following steps [6]:

4. First the Euclidean distances between each pairs of features is computed, as Formula 5 [7].

$$Euclidean\ distance\ (d_{a,b}) = \sqrt{\sum_i (a_i - b_i)^2} \quad (5)$$

5. Creating a tree from the calculated distances using average linkage.

6. The output is drawn using the dendrogram.

But Before doing cluster process, the matrix of plant's features should be prepared .This matrix contains number of rows equals the number of leaves in the sample, each row contains the main features of the leaf (global and local features).

In this paper this phase is considered a difficult phase, because each leaf contains number of leaflets and each leaflet has number of features, in addition to this each leaf contains different number of leaflets. So the dimensionality of matrix will be variety and this is a main problem.

All these reasons make the process of clustering in plant compound leaf more difficult than simple leaf.

To overcome these problems, the distance matrix is prepared as follows:

1. Each leaflet belongs to leaf in the sample is compared with other leaflets in other leaves using Euclidean distance of leaflet's features.
2. Then the minimum distance between each leaflet in the leaf and other leaves is taken and stored.
3. The minimum of distances between each leaflet in the leaf and other leaves are prepared for each leaflet.
4. Finally, the minimum distances between each leaf and others are stored.

These steps enabled us to find the matrix of leave's features; this squared matrix contains the following data as in Table 1. Rows and columns of matrix correspond to objects (leaves).

Table 1 the squared matrix of the features of the plant leaves

Leaf / Leaf	1	2	.....	n
1	Min distance Between leaf 1 and 1 (always 0)	Min distance Between leaf 1 and 2	.....	Min distance Between leaf 1 and n
2	Min distance Between leaf 2 and 1	Min distance Between leaf 2 and 2 (always 0)	.....	Min distance Between leaf 2 and n
.	.....	.....	.....	.....
.				
n	Min distance Between leaf n and 1	Min distance Between leaf n and 2	.....	Min distance Between leaf n and n (always 0)

V. Experiments and Results

A. *Extracting Features Experiments and Results*

We applied an extraction features methodology on 29 tomato leaf from different types in this paper from online database [8], by using images processing toolbox in MALAB. All features were extracted in an accurate way from this sample.

B. *Hierarchal clustering Experiments and Results*

We have applied Hierarchical clustering experiments on sample contains 29 tomato leaf from different types in this paper, by using statistical toolbox in MATLAB. The output shows the 'dendrogram' for two main clusters (Red and Blue). Figure 4 shows this result.

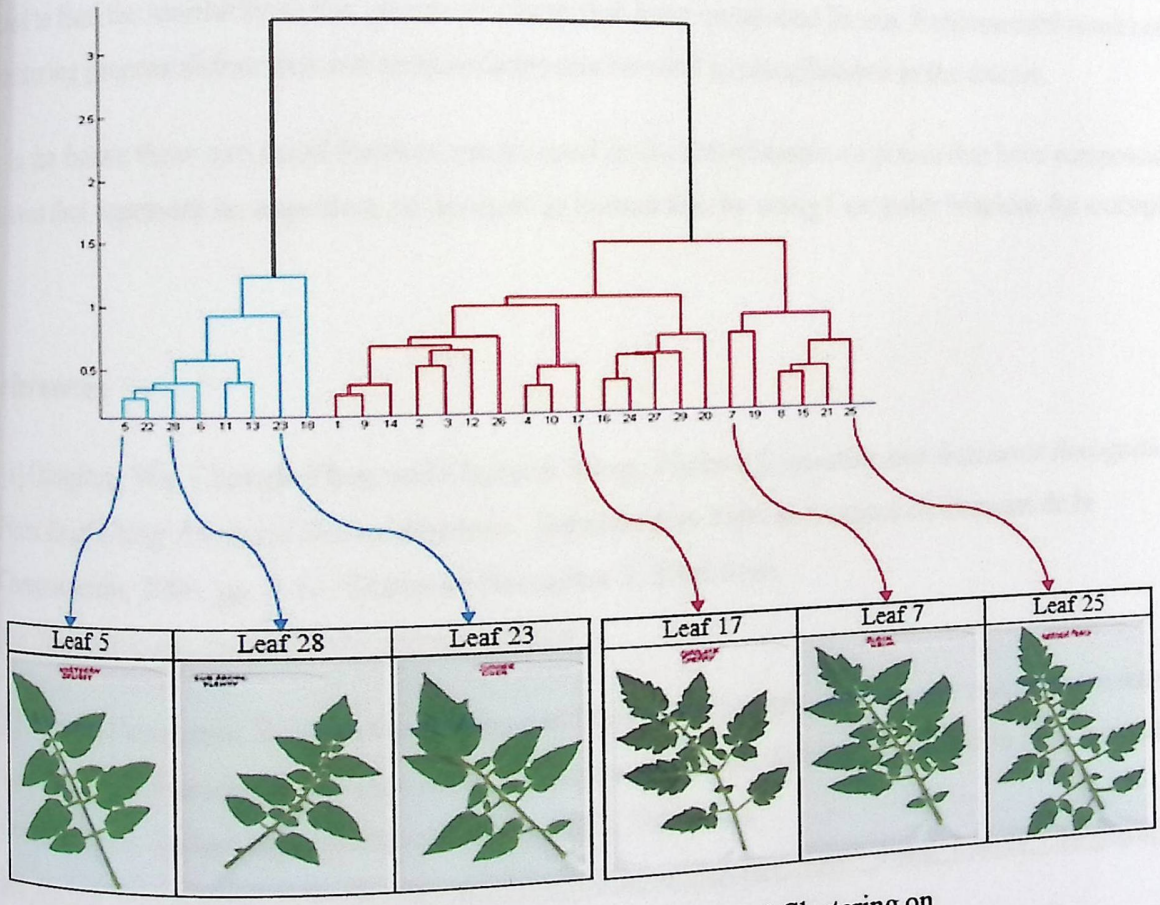


Figure 4 The result of applying Hierarchical Clustering on Three sample leaves from the two main clusters revealed by dendrogram

Figure 4 shows the result of applying hierarchical clustering on the sample, this figure shows there are number of clusters in this sample, this number depending on when we want to cut the dendrogram at the proper level, and shows an example of the two main clusters (Red and Blue). In this Figure we can see that the leaves in the blue cluster belongs to Potato Leaflet(PL) type that contains smooth edges and there are have a thick leaflets [9], but that the leaves in the red cluster belongs to Regular Leaflet(RL) type that contains a tootheed and symmetrical edges [9].

VI. Conclusion

This paper introduces a methodology to extract morphological and geometrical features from a plant compound leaf. 10 features are extracted from input a compound lead image, this methodology is not depending on the user. Hierarchal Clustering of plant compound leaf is applied on the sample to reduce the large number of plant's types of specific plant compound leaf to help in classification, and it can be used to find the similarity in the groups of plants that have compound leaves. Experimental results of the clustering process shows that our methodology can be used to classification in the feature.

In the future these extracted features can be used in the classification of plants that have compound leaves that represent an important component in human life, by using Computer Machine for example.

References

[1] Qingfeng Wu, Changle Zhou and Chaonan Wang, *Feature Extraction and Automatic Recognition of Plant Leaf Using Artificial Neural Network*. Submitted on 2006 in Avances en Ciencias de la Computación, 2006, pp. 5-12. Retrieved November 3, 2009 from http://prometeo.cic.ipn.mx/2006/cd/papers/1.pdf.

[2] Tzionas Panagiotis, E.Papaadais Stelios and Manolakis Dimitris, *Plant leaves classification based on morphological features and a fuzzy surface selection technique*. Submitted on 2005 in Fifth International Conference on Technology. Retrieved November 3, 2009 from http://www.autom.teithe.gr/gr/drastiriotites/impro3Dpos/papers/ICTA05_Plant_Leaves_Classification.pdf

[3] Gang Wu, Bao, You Xu, Wang, Chang, Xiang, *A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network*. IEEE International Symposium on Signal Processing and Information Technology, Dec. 2007, pages 11-16. Giza, from

http://arxiv.org/PS_cache/arxiv/pdf/0707/0707.4289v1.pdf.

[4] Fanning Consulting. *Convert RGB image to Grayscale*. Retrieved October 31, 2009 from http://www.dfanning.com/ip_tips/color2gray.html.

[5] *Cluster Analysis*: *Basic concepts and analysis*. Retrieved April 20, 2010 from

http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

[6] Zaïane, O. Principles of Knowledge Discovery in Databases, *Data Clustering.* Retrieved April 25, 2010 from http://webdocs.cs.ualberta.ca/~zaiane/courses/cmput690/slides/ch8s.pdf

[7] Statsoft Electronic Statistic Toolbox. *Cluster Analysis.* Retrieved April 25, 2010 from http://www.statsoft.com/textbook/cluster-analysis/

[8] Wintersown.org. *Tomato Leaf Scans*. Retrieved November 29, 2009 from

http://www.wintersown.org/wseo1/Tomato_Leaf_Scans.html

[9] Garden web. *Are there different types of tomato leaves?*. Retrieved November 22, 2009 from http://faq.gardenweb.com/faq/lists/tomato/2004111539004321.html

References

[1] Tamimi, H. "Vision-based Features for Mobile Robot Localization", ch.3, pp. 22-24, 2006.

[2] TutorVista.com. *Leaf*. Retrieved November 22, 2009 from http://www.tutorvista.com/content/biology/biology-iii/angiosperm-morphology/leaf.php.

[3] Britannica Encyclopedia. *morphology*. Retrieved October 29, 2009 from http://www.britannica.com/EBchecked/topic/392797/morphology.

[4] Biology of Horticulture. *Leaf Types*. Retrieved November 22, 2009 from http://www.hcs.ohio-state.edu/hort/biology/Lab/leaftypes.html.

[5] *Once compound leaves*. Retrieved November 22, 2009 from http://z.about.com/d/forestry/1/0/H/7/1compound_leaf.jpg.

[6] *Tomato*. Retrieved November 22, 2009 from http://www.scribd.com/doc/19826925/0756650941-Tomato.

[7] *Tomato Anatomy*. Retrieved October 29, 2009 from http://www-plb.ucdavis.edu/labs/rost/tomato/Leaves/leafvar.html.

[8] Favre, A. & Favre, J. *Solanaceae Stems and Leaves*. Retrieved November 22, 2009 from http://www.geauga4h.org/clubs/plantmasters/solanaceaenine.pdf.

[9] Wintersown.org. *Tomato Leaves Scans*. Retrieved October 29, 2009 fromhttp://www.wintersown.org/wseo1/Tomato_Leaf_Scans.html.

[10] Garden web. *Are there different types of tomato leaves?*. Retrieved November 22, 2009 from http://faq.gardenweb.com/faq/lists/tomato/2004111539004321.html.

[11] The Free Dictionary. *Image processing*. Retrieved October 31, 2009 from http://encyclopedia2.thefreedictionary.com/image+processing.

[12] The Free Dictionary.*RGB color model*. Retrieved November 22, 2009 from http://encyclopedia.thefreedictionary.com/RGB+color+model.

[13] *Geometric representation*. Retrieved December 28, 2009 from http://en.wikipedia.org/wiki/RGB_color_model

[14] Processing. *Color*. Retrieved November 22, 2009 from http://processing.org/learning/tutorials/color/.

[15] *Binary image*. Retrieved November 23, 2009 from http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT2/node3.html.

[16] Fanning Consulting. *Convert RGB image to Grayscale*. Retrieved October 31, 2009 from http://www.dfanning.com/ip_tips/color2gray.html.

[17] The math work. *Morphology Fundamentals: Dilation and Erosion*. Retrieved November 20, 2009 from http://www.mathworks.com/access/helpdesk/help/toolbox/images/f18-12508.html.

[18] Computer Vision. *Binary Morphology*, Retrieved May 25, 2009 from http://portal.ku.edu.tr/~yyemez/ecoe508/morphology.pdf

[19] Smith, S., *The Scientist and Engineer's Guide to Digital Signal Processing*, chapter 25, page 437. Retrieved November 20, 2009 from http://www.dspguide.com/ch25/4.htm.

[20] *Connected components labeling*. Retrieved October 20, 2009 from http://homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm.

[21] *Cluster Analysis*: *Basic concepts and analysis*. Retrieved April 20, 2010 from http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

[22] *A Tutorial on Clustering Algorithms*. Retrieved April 25, 2010 from http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/

[23] *ESOMAR* . Retrieved April 25, 2010 from http://www.esomar.org/index.php/glossary-d.html

[24] Stanley, L. , *Notes on Cluster Analysis.* Retrieved April 27, 2010 from http://www.uic.edu/classes/idsc/ids472/clustering.htm

[25] *Hierarchical Clustering.* Retrieved April 25, 2010 from http://www.resample.com/xlminer/help/HClst/HClst_ex.htm

[26] Zaïane, O. Principles of Knowledge Discovery in Databases, *Data Clustering.* Retrieved April 25, 2010 from http://webdocs.cs.ualberta.ca/~zaiane/courses/cmput690/slides/ch8s.pdf

[27] *Cluster Analysis.* Retrieved April 25, 2010 from http://www.norusis.com/pdf/SPC_v13.pdf

[28] Statsoft Electronic Statistic Toolbox. *Cluster Analysis.* Retrieved April 25, 2010 from http://www.statsoft.com/textbook/cluster-analysis/

[29] *Clustering.* . Retrieved April 27, 2010 from http://slidefinder.net/c/clustering/5430476

[30] Tzionas Panagiotis, E.Papaadais Stelios and Manolakis Dimitris, *Plant leaves classification based on morphological features and a fuzzy surface selection technique*. Submitted on 2005 in Fifth International Conference on Technology. Retrieved November 3, 2009 from http://www.autom.teithe.gr/gr/drastiriotites/impro3Dpos/papers/ICTA05_Plant_Leaves_Classification.pdf

[31] Gang Wu, Bao, You Xu, Wang, Chang, Xiang, *A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network.* IEEE International Symposium on Signal Processing and Information Technology, Dec. 2007, pages 11-16. Giza, from http://arxiv.org/PS_cache/arxiv/pdf/0707/0707.4289v1.pdf.

[32] Qingfeng Wu, Changle Zhou and Chaonan Wang, *Feature Extraction and Automatic Recognition of Plant Leaf Using Artificial Neural Network*. Submitted on 2006 in Avances en Ciencias de la Computación, 2006, pp. 5-12. Retrieved November 3, 2009 from http://prometeo.cic.ipn.mx/2006/cd/papers/1.pdf.

[33] The Mathworks. *MATLAB - The Language Of Technical Computing*. Retrieved November 22, 2009 from http://www.mathworks.com/products/matlab/

[34] The Mathworks. Image Processing Toolbox 7.0 Retrieved April 22, 2010 from http://www.mathworks.com/products/image/

[35] The Mathworks. *Statistics Toolbox 7.3* Retrieved April 22, 2010 from http://www.mathworks.com/products/statistics/