Active shape models with focus on overlapping problems applied to plant detection and soil pore analysis

Von der Naturwissenschaftlichen Fakultät
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades

Doktor der Gartenbauwissenschaften

Dr. rer. hort.

genehmigte Dissertation

von

M.Sc. Julio César Pastrana Pérez

geboren am 17 Juli 1978 in Cunduacan, Tabasco, Mexiko

2012

Referent: Prof. Dr. Thomas Rath

Korreferent: Prof. Dr. Bernhard Hau

Tag der Promotion: 29.02.2012

To my family
Erica, Genoveva, Juan Carlos, Laureano & Maria de los Angeles

ACKNOWLEDGEMENTS

First I offer my sincerest and deepest gratitude to my supervisor, Prof. Dr. Thomas Rath, who guided me, inspired me and encouraged me throughout the whole research and writing period. His vast knowledge in the subject allowed him to make excellent and keen observations, which had positive influences in this research. Also, without his patience, comprehension and support through difficult times, this thesis would not have been completed or written. I can only say: Thank you very much for being a such a great supervisor.

My sincere thanks also goes to PD Dr. Stefan Dultz from the Institute of Soil Science, Leibniz University of Hannover, who provided the tomographic data which was used for part of this thesis.

Additionally, I would like to thank the organizations CONACYT Mexico, DAAD, Colegio de Postgraduados Mexico and the Leibniz Universität Hannover for their financial support, without which this thesis would not have been possible. Also, many thanks to Stefanie Büchl, the contact person for CONACYT and DAAD, who helped me find my way through many bureaucratic labyrinths.

Special thanks to all my colleagues at the Biosystems & Horticultural Engineering Section (BGT), Leibniz University of Hannover, who showed me the real German culture and hospitality. I am also grateful for their patience in teaching me the nuances of the German language.

I also would like to thank my family, to whom this thesis is dedicated, for their patience, understanding and love. To my wife Erica, who was always there for me; thank you very much for reading this work ten thousand times, for helping me correct and improve its content. To my parents, Laureano and Maria de los Angeles, and my brother and sister, Juan Carlos and Genoveva, who always supported me throughout these many years, even when it meant being so far away from home.

Lastly, hats off to the Linux and Open source community for providing great software that allowed the execution of this work.

CONTENTS

Acknowledgements **i**

Abstract **vii**

1 Introduction **1**

2 General Objectives **3**

3 State of the art of pattern recognition in plant sciences **5**

3.1 General shape representation 5

3.1.1 Block representation 7

3.1.2 Geometric descriptors 7

3.1.3 Distance transform 8

3.1.4 Skeletons 9

3.1.5 Moments 10

3.1.6 (r, θ) Signature 12

3.1.7 Chain codes 14

3.1.8 Polygon approximation 15

3.1.9 Fourier descriptors 18

3.2 Elliptical representation of shapes 20

3.2.1 The ellipse 21

3.2.1.1 General equations 22

3.2.1.2 Parametric equations 25

3.2.1.3 Ellipse fitting 25

3.2.2 The ellipsoid 27

3.2.2.1 General equations 27

3.2.2.2 Parametric equation 28

3.2.2.3 Ellipsoid fitting 28
3.3 Active Shape Models 28
3.3.1 Shape representation and landmarks 29
3.3.2 The correspondence problem 30
3.3.3 Shape alignment 32
3.3.4 Shape model statistical analysis 34
3.3.5 Shape model deformation 37
3.3.6 Applications of ASMs 39
3.3.7 Applications of ASMs in plant science 40

4 Shape analysis using 2D ASMs for overlapping problems in plant science 41

4.1 Introduction to the overlapping problem 41
4.2 Related work 42
4.3 Objectives 44
4.4 General idea on how solve the overlaping problem 45
4.5 Materials and methods 46
4.5.1 Model plant 46
4.5.2 Comparison with human recognition 47
4.5.3 Image acquisition 48
4.5.4 Image segmentation - RGB plant color extraction 49
4.5.5 Ellipse detection using a clustering algorithm 51
4.5.6 Ellipse detection using arcs 57
4.5.7 Landmarking procedure 65
4.5.8 Deformable plant templates 68
4.5.9 Neighboring leaves combination 71
4.5.10 Energy heuristic to match real plants 75
4.6 Results 76
4.6.1 Comparison of ellipse detection methods 76
4.6.2 *Nicotiana tabacum*: leaf detection evaluation and sensitivity analysis 79

4.6.3 Example detections of other type of leaves 84
4.6.4 Experimental results 86
4.6.4.1 Evaluation of the identification Algorithm 88
4.6.4.2 Results using artificially generated situations . . 89
4.6.4.3 Comparison with human recognition 91
4.6.4.4 Results using real situations 93

5 Taking the ASM to the 3D space 97
5.1 Overlapping in the 3D space 97
5.2 Related work 99
5.3 Objectives 102
5.4 State of the art of micro pore analysis 103
5.5 Materials and methods 104
5.5.1 Rock and soil samples 104
5.5.2 Image acquisition 104
5.5.3 Image segmentation 107
5.5.3.1 Gaussian smoothing 107
5.5.3.2 Grayscale threshold 108
5.5.3.3 3D Flood-fill 111
5.5.4 Shape representation 113
5.5.4.1 Introduction 113
5.5.4.2 Line representation 113
5.5.4.3 Cuboid representation 114
5.5.4.4 Ellipsoid representation 114
5.5.5 Landmarking 115
5.5.6 Exhaustive exploration approach 117
5.5.7 Generation of ASMs and their evaluation 117
5.5.8 Procedure to find the best size intervals 117
5.6 Results 119

6 Discussion 125

6.1 2D ASMs for detecting overlapping plants. 125
    6.1.1 Color segmentation 125
    6.1.2 Leaf detection 126
    6.1.3 ASM Plant recognition 129
    6.1.4 Possible applications 132
6.2 3D ASMs for overlapping problems 134
    6.2.1 Image preprocessing 134
    6.2.2 3D ASM identification 136
    6.2.3 Possible applications 137

7 Conclusions **139**

A Mathematical background **141**
A.1 Polar Coordinates 141
A.2 Arctangent function 141
A.3 Metric schemes 142
A.4 Centroid of a point set 142
A.5 Rotations in $\mathbb{R}^3$ 142
A.6 Singular Value Decomposition (SVD) 143
A.7 Complex numbers 144
A.8 Euler's formula 144
A.9 Fourier Transform 144
A.10 Gaussian smoothing of images 145

Symbols and Abbreviations **146**

List of Figures **151**

List of Tables **155**

List of Algorithms **157**

Bibliography **161**

ABSTRACT

The purpose of this research is to develop computerized identification models that have the ability to classify objects by only using patterns contained in their shape. Additionally, overlapping conditions are the main consideration of the problem at hand. This work presents two recognition systems (in 2 and 3 dimensions) that are able to perform recognition under such difficult overlapping situations. The applied methodology uses Active Shape Models as the main identification technique to accomplish the goal. Moreover, the complexity of the original shapes is reduced by approximating them to elliptical forms (ellipse in 2-dimensions and ellipsoid in 3-dimensions). This simplification has the benefit of allowing the use of simpler methods to find the points that best represent a form.

In the 2-dimensional space the problematic is the identification of small overlapping plantlets. Whereas, in the 3-dimensional space the problematic is to find the shape patterns that best characterize the overlapping pore structures of soil and rock samples.

The results indicate that it is possible to perform identification of objects under overlapping situations, by first decreasing the complexity of their form and then using these simplified characteristics in a statistical shape model. In the 2-dimensional space it is possible to automatize the identification of plantlets with 2, 3 and 4 leaves in a variety of overlapping situations by first executing an ellipse detection technique, which simplified the form of the leaves. The next step grouped these ellipses into individual plantlets by using a heuristic method based on deformation energy. A similar technique was used to identify complex overlapping structures in solid objects. The shape of the structures was first reduced to an ellipsoid which allowed the use of deformable models to search and identify specific shape characteristics within the complete structure set.

Keywords: Active shape models, image processing, pattern recognition, landmarking, ellipse fitting, plant identification, overlapping, 3D shape recognition, soil pores, ellipsoid fitting.

ZUSAMMENFASSUNG

Ziel dieser Arbeit ist die Entwicklung rechnergestützter Identifikations-Modelle zur Klassifizierung von Objekten einzig durch das Erkennen von Mustern in der Objektform. Eine besondere Herausforderung ist hierbei die Erkennung sich überlappender Objekte. Die vorliegende Arbeit stellt zwei Erkennungssysteme vor, die sich überlappende Objekte im zwei- bzw. im dreidimensionalen Raum identifizieren. Hierfür werden als zentrale Identifikationsmethode sog. Active Shape Models verwendet. Zudem wird die Komplexität der originalen Formen durch Approximation an elliptische Formen reduziert (Ellipsen für zwei, Ellipsoide für drei Dimensionen). Durch diese Vereinfachung wird der Einsatz einfacherer Methoden zur Ermittlung der die Objektform repräsentierenden Punkte ermöglicht.

Im 2-dimensionalen Raum wird sich diser Problematik genähert, indem kleine sich überlappende Jungpflanzen identifiziert werden. Hingegen besteht die Problematik im 3-dimensionalen Raum darin diejenigen Muster zu erkennen, die sich überlappende Poren in Stein-Boden-Proben am besten darstellen.

Die Ergebnisse zeigen, dass die Erkennung sich überlappender Objekte durch die Reduzierung der Komplexität ihrer Form und durch die anschließende Verwendung dieser vereinfachten Merkmale in statistischen Shape Models möglich ist. Im 2-dimensionalen Raum konnten Jungpflanzen im 2, 3 und 4-Blattstadium mit überlappenden Blättern mithilfe der Erkennung der entsprechenden Ellipsen automatisch identifiziert werden. Hierzu konnten die Ellipsen durch eine heuristische Methode, basierend auf der Deformationsenergie, den entsprechenden Pflanzen zugeordnet werden. Ein vergleichbarer Ansatz wurde zur Identifizierung komplexer, sich überlappender Strukturen in festen Objekten verwendet. Die Form der Strukturen wurde zunächst auf ein Ellipsoid reduziert, wodurch die Verwendung deformierbarer Modelle zur Suche und Identifikation spezifischer Formeigenschaften innerhalb der gesamten Menge an Strukturen ermöglicht wurde.

Schlagwörter: Active shape models, Bildverarbeitung, Mustererkennung, Landmarking, Ellipse fitting, Pflanzendetektion, Überlappung, 3D Formerkennung, Bodenporen, Ellipsoid fitting.

CHAPTER 1

Introduction

During the course of only a few decades, computers have influenced every aspect of our lives, including many steps of every production process. Agriculture and Horticulture are no exception to this: since the “Green Revolution” (Borlaug, 1996) new technologies have constantly been incorporated into the productive models.

> “Computers and their applications are now pervasive throughout the agricultural community - from the land grant system to the farm tractor.” “The role that computers play in agriculture has evolved over time from simplistic record keeping and spreadsheets, to incorporating expert systems to solve complex problems and manage complete farm systems.” (Wakefield & King, 1994, p. 1-2)

The need to incorporate new technologies and automatization in the area of food production has grown enormously. This is not only due to the increasing costs of manual labor and fuel, but also due to so-called Agricultural Megatrends (Havlicek, 1986; von Witzke et al., 2009), such as increase in population, biofuel demand and climate change, which put additional stresses on the existing food supply and increase the demand for more production output.

Scientists have dedicated much time and energy to the development of technologies focused on the automatization of monotonous and repetitive tasks, as well as those that are dangerous or may detrimentally effect the health of the agricultural laborers. Some of this work can be executed using preprogrammed routes where no visual information is required, such as irrigation, and the bulk application of herbicides, pesticides and fertilizers. Other operations, however, need intimate knowledge of the environment and location of specific plants in order to be successfully carried out. This includes trimming, weeding, classification of the produce, harvesting, and plant specific fertilization and precision application of herbicides and pesticides, etc.

Machine vision plays a very important role when developing robots that will autonomously perform farming or horticultural tasks. The capacity of a robotic system to cope with the production demands very much depends on how accurately and efficiently it recognizes the objects while performing its task. This visual information is obtained from sensory devices, such as cameras, which provide the system with information about the surrounding environment.

Currently, there are many image processing techniques that only take into account the color information provided by the captured images to perform object detection. For example, the recognition of ripe tomatoes by identifying their red color, or identifying rows of crops by only using the green color of the plant versus the black or brown color of the soil. This, of course, is a valid approach, but is very limited because only part of the available information is used and can readily be confused by other objects with the same color. Alternatively, there are many other techniques that also consider additional information contained within the image, such as the shape of the object to be identified (see Chapter 3). Certainly, taking into account the color and shape information will lead to the development of more accurate pattern recognition techniques. Nevertheless, when performing shape identification, there are many problems that one can encounter that will hinder the recognition of the objects, for example, variability in the object shape, overlapping, and total occlusions. This problem has motivated the search for alternative vision systems that are not susceptible to such conditions.

The current work presents new methods in the fields of image processing and pattern recognition, which have the ability to deal with difficult environments where the shapes that one wants to identify are overlapped by other objects. To be more specific, the techniques presented here are based on the reduction of the original shape to a simplified form and the application of Active Shape Models (ASMs), which are deformable templates that are employed to recognize and match similar forms.

CHAPTER 2

General Objectives

In the field of precision horticulture/agriculture, the recognition of objects is essential for further development of fully automated identification systems. One has to take into account that in real life circumstances, objects are usually not fully in view, and that part of its shape is almost always hidden, therefore, making it necessary to develop identification systems that are able to cope with such an environment.

Several attempts have been made to address such situations, but have been met with varying degrees of success. These take advantage of different qualities of the objects that are to be recognized, such as form/outline, color, texture, etc. The physical qualities are then processed with different computational techniques, such as , geometric descriptors, distance transformation, skeletons, chain codes, Fourier descriptors, etc. (these methods will be discussed in further detail in the following chapters). Although some of these methods have a high rate of success in artificial conditions, their usability rapidly decreases in overlapping and open field situations.

The general objective of this work is to solve the overlapping problem for plants by using a special ASM (Active Shape Model) algorithm and to transport this algorithm into a higher space using a complex real situation.

The fundamental idea behind ASMs is the use of statistical analysis of a training set, which is formed by certain number of similar objects, with the purpose of extracting all possible object variations. These variations are then placed into a mold, which can be deformed into similar shapes to the ones contained in the training set. Certainly, the shapes within the training set establish the boundaries of the deformations. However, this does not hinder the ability of the method to create new forms within these same boundaries. This allows the system to be able to cope with the variance which is found in groups of biological organisms and other naturally occurring structures.

However, the ability of ASMs to create and match similar shapes can be detrimentally affected when part of the information needed to perform the identification is hidden. In order to solve this problem, an additional step had to be executed, where the form of the object to be identified was first modeled as a simple geometric shape, such as an ellipse, thereby reducing the amount of information the pattern recognition system has to deal with. Because of the regularity of the simplified form, it is possible to reconstruct the whole form when only a fraction of the visual data is available. In this manner, the system uses the original information to locate geometrical shapes that will uncover the position of the partially occluded object one is looking for.

This method was then adapted to the 3-dimensional space in order to identify overlapping structures. This was carried out using the shape of amorphous pore systems, whose original irregular shape was reduced to an ellipsoid form. Thus, it was possible to apply the ASMs to generate deformable models that led to their correct identification.

CHAPTER 3

State of the art of pattern recognition in plant sciences

3.1 General shape representation

Object identification is the primary goal of computer vision. In order to perform this task, the relevant information contained in the images has to be extracted and coded in a way that it is meaningful to a computer.

Before attempting any object identification, the image has to be separated into different regions or sections (this is referred as the segmentation process), where each region contains information that can be dismissed or used in further analysis. Thus, it is possible to say that the segmentation process divides an image I into n different regions $R_1, R_2, \ldots, R_n$

$$I = \bigcup_{i=1}^{n} R_i, \quad R_i \cap R_j = \emptyset, \quad i \neq j \tag{3.1}$$

where each R_i might contain information about a particular object or can be ignored because it does not provide any valuable data.

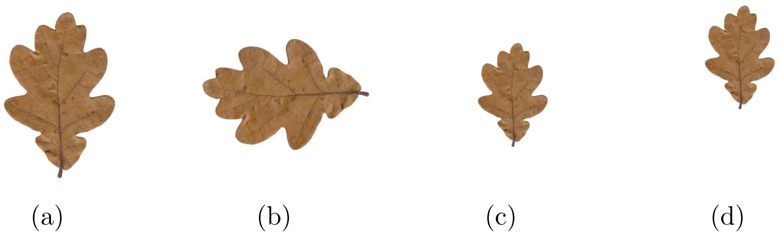
Segmentation processes are task specific because there is no general answer to the question of how to subdivide an image into specific regions . There are several approaches to carry out segmentation, i.e., threshold-based, cluster-based, template-based, boundary-based (Comaniciu & Meer, 2002; Fu & Mui, 1981; Gonzales-Barron & Butler, 2006). These techniques have to be adapted to work under the particular conditions of the environment or the problem domain.

Object representation has always been a topic of discussion in pattern recognition. Currently, there is no real consensus about how to represent nor how to select the distinct features of objects when developing a vision system. The intrinsic properties of an object or its invariant features are shape, texture, color, size, etc. Of all these properties, shape is the one that stands out most, because it provides a large amount of information that is unique to a given object (Bouet et al., 1999; Ullman, Shimon, 1996). Even in the complete absence of sight, shapes can be identified by humans using the sense of touch. In robotic systems, laser and ultrasonic range sensors, etc. can be used to extract the shape of an object (Burgard et al., 1999; Hähnel et al., 2003; Masaphy et al., 2009; Oppelt & Grandke, 1993; Triebel et al., 2005; Wells, 2006).

In regard to the problem of representing shapes, Ballard & Brown (1982, p. 229) mention that "One of the major challenges to computer vision is to represent shapes, or the important aspects of shapes, so that they may be learned, matched against, recollected, and used".

Researchers have already devoted much time and effort in developing different shape representation schemes, which can be classified into two main groups: Contour-based and Region-based (Bouet et al., 1999; Gonzalez & Woods, 2001; Zhang & Lu, 2004). Contour-based representation techniques only use the contour of a shape to extract information, whereas Region-based techniques use the whole object region to obtain information.

Regardless of the selected representation technique, it has to provide a way to describe each object in a unique way. It also has be sensitive enough to find the differences between similar objects (Pitas, 2000, Ch. 7). Additionally, it has to be able to tolerate transformations for invariant features, such as scaling, rotation and translation. This way the object can be analyzed and identified regardless of its position and size. Figure 3.1 shows some invariant transformations applied to a leaf.



(a) (b) (c) (d)

Figure 3.1: This figure shows some invariant transformations of an object. (a) original leaf. (b) rotation. (c) scale. (d) translation and scale.

In the following subsections, some basic methods for representing and matching shapes are briefly described. The main idea of presenting these popular techniques is to familiarize the reader with the pattern recognition subject before the primary method used in this work is introduced.

3.1.1 Block representation

The use of blocks to represent shapes is one of the simplest methods in computer vision. A template is built by extracting a rectangular section of an image where the object or region of interest is contained, see Figure 3.2.

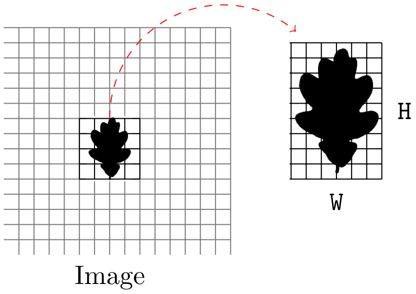


Figure 3.2: Shape block representation. The object is contained within a rectangular section of `W` × `H` pixels.

The comparison between two different blocks is executed in a pixel-wise manner; in ideal situations, two blocks are said to be a match when all of their pixels are equal, however, in reality such cases where there is an exact match are rare. There are different criteria to correlate two blocks, for example: mean absolute distance, mean squared distance, normalized cross-correlation, etc. (Gyaourova et al., 2003). The following equation computes the similarity of two blocks (**A** and **B**) using the mean absolute distance (mad):

$$\text{mad} = \frac{1}{\mathtt{WH}} \sum_{i=1}^{\mathtt{W}} \sum_{j=1}^{\mathtt{H}} |\mathbf{A}(i,j) - \mathbf{B}(i,j)| \tag{3.2}$$

where i and j are pixel positions within each block

This description scheme is very sensitive to noise, it also has problems when objects are scaled or oriented differently than the block template; this means that different templates are required for different situations.

Velduis & Brodland (1999) used an adapted version of the block matching technique to find the differences in pigmentation of Axolotl epithelial cells. Garcia & Gomez (2008) also use this method to track the motion of a tractor that is on an open filed carrying out agricultural tasks.

3.1.2 Geometric descriptors

Shapes can also be represented using its geometric attributes, such as, area, width, length, perimeter, roundness, aspect, etc., see Figure 3.3.

Area is the space or site on which the shape stands. In image processing, it is the number of pixels that compose the shape.

Width is the length of the shortest side of the minimum bounding rectangle where the shape is contained.

Length is the length of the longest side of the minimum bounding rectangle where the shape is contained.

Perimeter is the length of the outline of the shape.

Roundness is the circularity of the shape. This descriptor is calculated using the following formula: $\text{Perimeter}^2/4\pi\text{Area}$.

Aspect is the ratio between Length/Width.

Rectangularity is the ratio between the area occupied by the shape and the area occupied by the minimum bounding rectangle.

Convexity is a ratio similar to rectangularity, however, in this case the ratio is computed using the area occupied by the shape and the area occupied by its convex-hull.

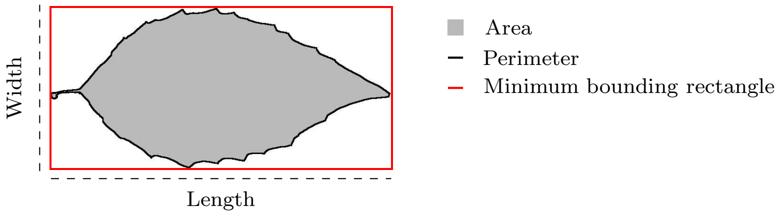


Figure 3.3: Example of geometric descriptors to represent the shape of a leaf.

A geometric descriptor by itself does not provide enough information to perform shape identification. However, if several descriptors are put together in the form of feature vectors, they can be used create different templates that can later be used to perform shape matching. A simple identification procedure is to compare an unknown feature vector to different feature vector templates, which represent specific shapes, and calculate how much it deviates from them. If the deviation is not greater than certain threshold, a match can be established.

For example, Hemming (2000); Rath (1997) used geometric descriptors as means to identify weeds on a field. Antonio, Dell'Aquila (2006) also used similar schemes to evaluate the quality of different plant seeds.

3.1.3 Distance transform

The distance transform is a type of representation that describes a shape using a map of distances. The generated map contains the distance of each point inside the shape to the closest point on the boundary, see Figure 3.4.

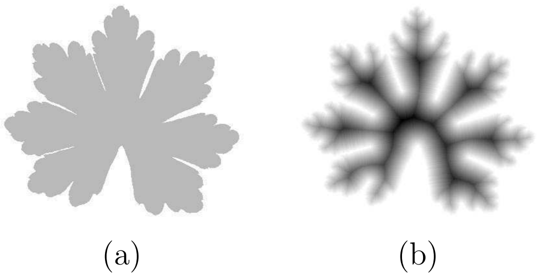

(a) (b)

Figure 3.4: Example of a distance transform: (a) Leaf shape. (b) Map of distances, this gray scale map depicts shades of gray where the distances to the boundary are small, and black for those distances that are the furthest to the boundary.

There are different types of distance metrics that can be used to compute the distance map (Fabbri et al., 2008; Pedro & Daniel, 2004), for example:

$$\begin{array}{cc} \text{Chessboard distance} & \max(|x_2 - x_1|, |y_2 - y_1|) \\ \text{Manhattan distance} & |x_2 - x_1| + |y_2 - y_1| \\ \text{Euclidean distance} & \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \end{array} \tag{3.3}$$

In order to perform shape identification, templates are created from the distance maps. Then unknown shape maps can be correlated with these templates in order to find the best match (Holzer et al., 2009). Normally, the correlation between two templates is calculated using the Chamfer system (Barrow et al., 1977). Gavrila (2007) also mentions that the use of distance transformation maps allows more variability between templates and the objects of interest.

As application examples, Norell (2011) employed this method to count the annual rings on wood sections of Pine tree samples. Duan et al. (2011) used distance maps to discriminate and count rice spikelets. Carroll & Holden (2005) used this technique to quantify weed distribution on open fields.

3.1.4 Skeletons

Skeletonization is an operation that iteratively reduces the regions in an image to a single-pixel-wide representation. Figure 3.5 depicts an example of two shapes and their skeletons.

There are three main approaches to extract the skeleton of a region (Klette, 2002; Lakshmi & Punithavalli, 2009):

1. thinning or skeletonizing, which iteratively use erosion to reduce the region.
2. distance transform techniques, which detect the highest elevations in the distance map to generate the skeleton.

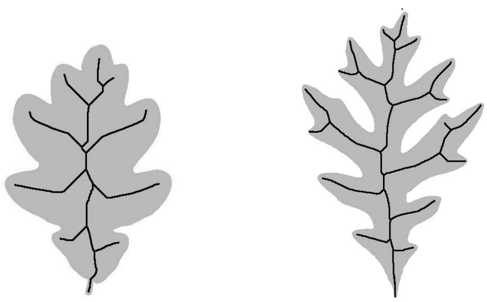


Figure 3.5: Shape of leaves and their skeletons.

3. Voronoi diagrams, which are generated using the outline of the region.

The skeleton of a shape contains important geometrical and topological features. However, in order to perform object recognition, the skeleton has to be represented in the form of graph-structures or interconnected nodes that are later used in the matching step (Bai & Latecki, 2008; Golland et al., 2000; Gonzalez & Woods, 2001). The nodes that model an skeleton are of 3 types "leaves nodes, junction nodes and internal nodes" (Golland et al., 2000).

Skeleton identification schemes are very sensitive to shape variations and to partial occlusions; small irregularities generate different skeletons that might influence the identification result. In order to overcome some of these problems, pruning techniques are used to avoid extra parts in the skeleton.

Pla & Juste (1995) proposed a technique based on skeletonization "to detect, locate and estimate the length of fruit stems from binary profile images. The algorithm uses the shape information provided by the pattern skeletons to characterize and identify the stems or any protrusion from the fruit body." Huang & Lee (2010) used skeletonization to identify the grasping points of culture plantlets; after the identification process, these points are sent to a robotic gripper for further manipulation of the plantlets.

3.1.5 Moments

Flusser et al. (2009, p. 6) describe moments as "scalar quantities used to characterize a function and to capture its significant features." Moments were first studied by David Hilbert in the 19th century and introduced into the world of pattern recognition by Hu Ming-Kuei in 1962 (Flusser, 2006; Hilbert, 1994; Hu, 1962).

In a two dimensional space, the moments $\mathfrak{m}_{ij}$ of an image function $f(x, y)$ are defined in terms of Riemann integrals as:

$$\mathfrak{m}_{ij} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^i y^j f(x, y) dx dy \quad i, j \in \{0, 1, 2, 3, \dots\} \tag{3.4}$$

The above integral can be replaced by summations in the case of digital images of size $\mathtt{W} \times \mathtt{H}$

$$m_{ij} = \sum_{x=1}^{\mathtt{W}} \sum_{y=1}^{\mathtt{H}} x^i y^j \, f(x,y) \quad i,j \in \{0,1,2,3,\dots\} \tag{3.5}$$

where $(i+j)$ represents the order of the moment; (x,y) are the coordinates of pixels and $f(x,y)$ denotes the intensity of the pixel. In the case of binary images, there are only two possible values: 1 to express a point on the object and 0 for every other point.

The lowest ordered moment, m_{00}, denotes the mass or the area of the object in the image. Furthermore, the two first order moments m_{10}, m_{01} define the centroid or center of gravity of the object $(\bar{x}, \bar{y})$

$$\bar{x} = \frac{m_{10}}{m_{00}} \; , \; \bar{y} = \frac{m_{01}}{m_{00}} \tag{3.6}$$

The second order moments, (m_{20}, m_{11}, m_{02}), can be employed to ascertain some useful characteristics, such as the principal axes, the image ellipse and the radii of gyration (Prokop & Reeves, 1992). Moments can also be invariant to geometric transformations such as translation, scale and rotation; equation 3.7 shows how to eliminate the translational variability after computing the centroid $(\bar{x}, \bar{y})$:

$$m_{ij} = \sum_{x=1}^{\mathtt{W}} \sum_{y=1}^{\mathtt{H}} (x-\bar{x})^i (y-\bar{y})^j f(x,y) \quad i,j \in \{0,1,2,3,\dots\} \tag{3.7}$$

Translation invariance is achieved by subtracting the centroid $(\bar{x}, \bar{y})$ from every point that belongs to the shape, as shown in the above equation. Size or scale invariant moments are a result of a simple size normalization using the zeroth moment m_{00}; by calculating the principal axes of the object, it is possible to achieve the rotation invariance.

Hu (1962) proposed a set of seven invariant moments to describe and identify shapes, these so-called *hu-moments* are invariant to translation, rotation and scale:

$$\begin{array}{rcl}
hm_1 & = & (m_{20} + m_{02}) \\
hm_2 & = & (m_{20} - m_{02})^2 + 4m_{11}^2 \\
hm_3 & = & (m_{30} - 3m_{12})^2 + (3m_{21} - m_{30})^2 \\
hm_4 & = & (m_{30} + m_{12})^2 + (m_{21} + m_{03})^2 \\
hm_5 & = & (m_{30} - 3m_{12})(m_{30} + m_{12})((m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2) \\
 & & + (3m_{21} - m_{03})(m_{21} + m_{03})(3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2) \\
hm_6 & = & (m_{20} - m_{02})((m_{30} + m_{12})^2 - (m_{21} + m_{03})^2) \\
 & & + 4m_{11}(m_{30} + 3m_{12})(m_{21} + m_{03}) \\
hm_7 & = & (3m_{21} - m_{03})(m_{30} + m_{12})((m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2) \\
 & & + (m_{30} - 3m_{12})(m_{21} + m_{03})(3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2)
\end{array} \tag{3.8}$$

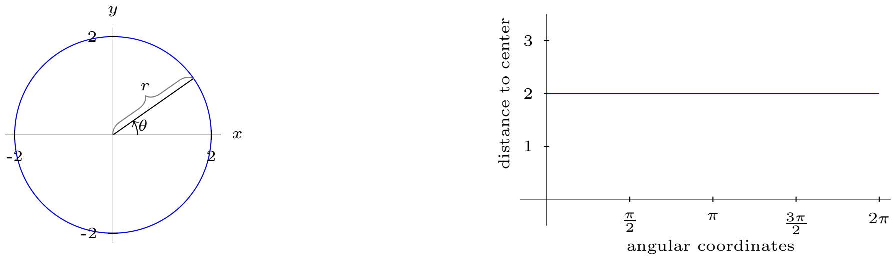
An invariant moment does not supply enough information to execute shape identification. However, if they are grouped in the form of feature vectors, they can be used to perform object classification and matching. It is worth noting that invariant moments have been proven to be effective with simple shapes. Nevertheless, in the presence of noise and partial occlusions, they do not produce adequate results (Premaratne et al., 2006).

Some example applications where moments were used are: Zion et al. (1999) used moments to sort different varieties of fish. Panigrahi et al. (1998) employed them to classify germplasms (ear of corn). Unay et al. (2011) also used this approach to grade apple fruits. Other example is the research carried out by Du et al. (2007), which used moments to identify different plant species based on their leaf shape.

3.1.6 (r, θ) Signature

The (r, θ) signature is an other possible representation of shapes that uses the information contained in the outline of the object and the polar coordinate system (see appendix A.1) to generate a plot that allows the visualization of the radial distance of each outline point and their angle with a reference axis.

Usually, the center of mass or centroid of the shape is used as a reference point to compute the radial distance r. Moreover, the angle θ (bounded between 0 and 2π) is calculated using the positive Cartesian x-axis as reference. As an example, Figure 3.6 shows a circle and its corresponding (r, θ) signature. It is clear that the signature plot is a line, since the distance from every point to the center of the circle is constant.



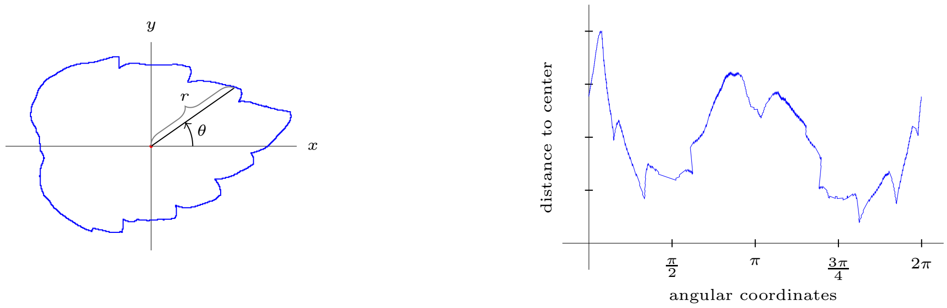
Circle outline

(r, θ) Signature

Figure 3.6: (r, θ) Signature of a circle.

Another example is depicted in Figure 3.7, where the (r, θ) signature of a leaf is shown. In this image, it is possible to see that there are more variations in the radial distance from the center as the angle varies from 0 to 2π. Furthermore, the possible values of the radial distance r are within the interval $[0, \max r]$, where $\max r$ is the maximum radial distance found in the shape.

Computation of the (r, θ) Signature is to some degree very simple. Having an n number of (x, y) points on the shape boundary, one starts by calculating the



Leaf outline

(r, θ) Signature

Figure 3.7: (r, θ) Signature of a leaf.

centroid $(\bar{x}, \bar{y})$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{3.9}$$

the radial distance r_i from each point to the centroid is

$$r_i = \sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2} \tag{3.10}$$

and the corresponding radial angle is

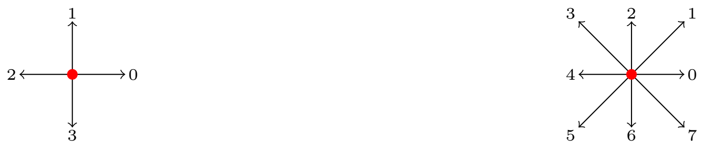
$$\theta_i = \arctan \frac{y_i - \bar{y}}{x_i - \bar{x}} \tag{3.11}$$

The (r, θ) signature is not the only type of its kind, there are similar techniques that use “tangent angle, cumulative angle, curvature, area and chord-length” (Zhang & Lu, 2004, p. 4). The idea behind these types of signatures is to reduce the dimensionality of 2D shapes into 1D functions. In addition, signatures can be size and translation invariant by normalizing and shifting the shape centroid to the origin. In order to achieve rotation invariance, the signatures have to be shifted until the error is minimized when comparing them to a prototype signature, making the method computationally expensive. Other disadvantages of signatures are their sensitivity to noise and to small changes in the outline of the shape (Zhang & Lu, 2004).

Example applications of this technique are: Zion et al. (2007) use shape signatures to sort underwater edible fish species. Blasco et al. (2009a) implemented a system that sorts mandarin oranges (Citrus unshiu). Additionally, Blasco et al. (2009b) created a system that identifies surface defects on the skin of citrus fruits. Mebatsion et al. (2011) also used this technique to model the shape of different fruits.

3.1.7 Chain codes

First proposed by Freeman (1961), chain codes are a simple way to represent shapes contained in binary images. This technique uses the boundary pixel list of a shape to generate a sequence of values that indicates the direction that one has to follow in order to move from one pixel to the next neighboring pixel in the boundary. Usually, a 4-pixel or 8-pixel coding system is used to generate the sequence (see Figure 3.8).

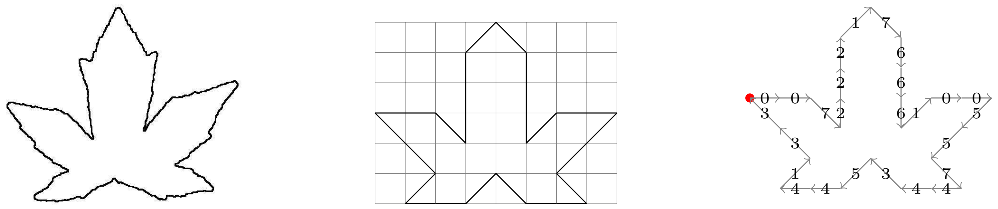


4-direction code

8-direction code

Figure 3.8: Chain code system.

The algorithm starts by arbitrarily choosing a starting point in the boundary sequence. Then, using any one of the above described coding systems (see Figure 3.8), it selects the next pixel in the boundary. Normally the algorithm uses a counterclockwise direction scheme to decide which neighboring pixel follows in the chain code.



(a) Leaf boundary

(b) Pixilated boundary

(c) Chain code

Chain code: {0,0,7,2,2,2,1,7,6,6,6,1,0,0,5,5,7,4,4,3,5,4,4,1,3,3}

Figure 3.9: Example of a leaf chain code. For convenience, the original leaf boundary (a) was pixelated into larger sized pixels (b). This way it is easier to follow the direction that the algorithm takes on every step (c).

Figure 3.9 exemplifies how to extract from a leaf boundary its corresponding chain code. Despite the simplicity of this scheme to represent shapes, it has some disadvantages when using it to identify objects. Just to mention a few: It is very sensitive to changes in the boundary, where small changes generate different chain codes; It is dependant on the scale of the image, so different grid sizes also lead to different chain codes; The extracted chain code depends on the selected starting point.

As application examples, Rath (1997) and Nam et al. (2008) used chain codes to analyze different shapes of leaves.

3.1.8 Polygon approximation

A Polygonal approximation is a way to represent the contour of a shape "using a subset of the original vertices" (Kolesnikov & Fränti, 2007, p. 1282). It is necessary to keep in mind that one can consider the outline of a shape as a set of points (vertex) that are connected by small line segments (edges), which in turn create a polygon. Therefore, polygon approximation finds a polygon with a lower number of edges that represents the original outline polygon.

The reduction in the number of edges is carried out by locating the dominant points or break points that best represent the original boundary shape. There are two main techniques to find the break points: a Merging technique and a Splitting technique (Daeho & Seung-Gwan, 2010; Loncaric, 1998).

The following steps describe the merging technique:

Algorithm 3.1 POLYGON APPROXIMATION MERGING TECHNIQUE (s)

```
Input: The outline points of a shape s.

1: Combine neighbor points along the boundary until the least square error of the line
   that fits these points is greater than a fixed threshold.
2: The two end points of the approximated line are saved as the vertices that join an
   edge of the polygon.
3: The above steps (1 and 2) are repeated until there are no more points to process in
   the analyzed boundary.

Output: $n$ number of edges that best approximate the shape s.
```

As an example, Figure 3.10 depicts how the merging technique works when approximating the shape of a leaf with a polygon. Images 1 to 3 show how the algorithm computes the first 3 edges of the polygon. Finally, the image marked as *k* represents the kth step of the algorithm to compute the last edge.

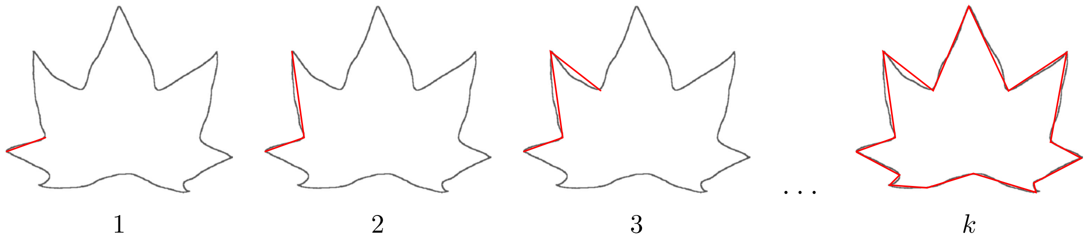


Figure 3.10: Polygon approximation using the merging technique. The series of images represent the n edges the algorithm computed to approximate the polygon.

If the threshold that controls the allowed maximum error is too large, there is the risk that approximated edges are incorrect, see Figure 3.11.

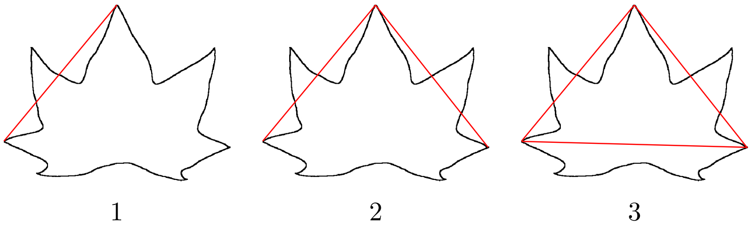


Figure 3.11: Erroneous polygon approximation: merging technique.

The second technique -splitting technique- works by iteratively subdividing the shape boundary into several parts until there are no more break points to extract because a specified criterion was not satisfied. The following steps describe how to use the splitting technique:

Algorithm 3.2 POLYGON APPROXIMATION SPLITTING TECHNIQUE ($\mathbf{s}$)

```
Input: The outline points of a shape s.
1: locate the two points that are farthest away within the boundary.
2: draw a line segment that connects the two points and subdivide the boundary into two segments.
3: for each segment find the point that maximizes the perpendicular distance to the previously calculated line segment.
4: draw the lines joining the maximum perpendicular distance points and the two end points.
5: subdivide the boundary segment into two subsegments.
6: repeat steps 3 to 5 until the perpendicular distance is greater than the specified threshold.
Output: n number of points that best approximate the shape s.
```

Figure 3.12 illustrates the splitting technique for polynomial approximation using the boundary of a leaf. (1) shows the farthest points within the boundary and the line that connects them. (2) shows the points that maximize the perpendicular distance to the line connecting the first two points. Note that each point belongs to one of the two segments created when the the original boundary was subdivided. (3) new lines are drawn between the located points. Additionally, two more boundary segments are created. (4) shows the new maximum perpendicular distance points to each line. (5,...,k) show how the procedure is repeated until no more points are located.

In this approach, errors might occur if the threshold that controls the perpendicular distance to the line connecting two dominant points is too small or too

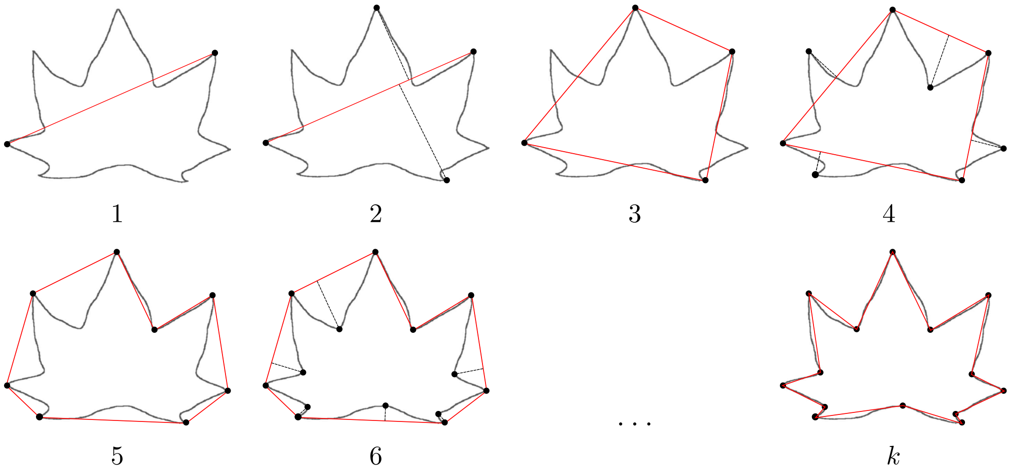


Figure 3.12: Polygon approximation using the splitting technique. The sequence of images $1, 2, \ldots, k$ depict the steps that the method follows to find the break points in the leaf boundary.

large. In the first case, all the points of the shape might appear as dominant points. In the second case, only a few points of the shape might be identified, see Figure 3.13.

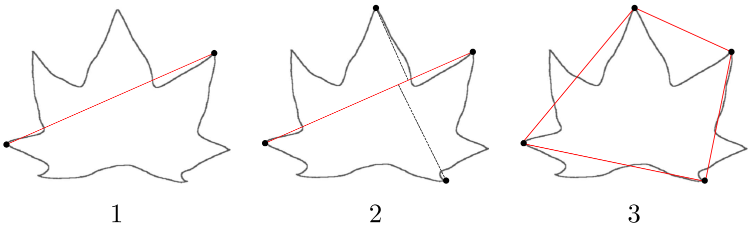


Figure 3.13: Erroneous polygon approximation: splitting technique

After executing one of the above methods to simplify two polygons, it is possible to compare them using the Hausdorff distance, which can be used to calculate the similarity of two sets of points (Atallah, 1983; Helmut et al., 1995). In the case of shape matching, one set of points represents a model $\mathbf{X}$, while a second set $\mathbf{Y}$ contains the information about the object to be recognized. The Hausdorff distance Hd between these polygons is defined as

$$Hd(\mathbf{X}, \mathbf{Y}) = \max\left(hd(\mathbf{X}, \mathbf{Y}), hd(\mathbf{Y}, \mathbf{X})\right) \tag{3.12}$$

the function hd represents the one-sided Hausdorff distance from $\mathbf{X}$ to $\mathbf{Y}$ or from $\mathbf{Y}$ to $\mathbf{X}$, which is

$$\begin{aligned} hd(\mathbf{X}, \mathbf{Y}) &= \max_{x \in \mathbf{X}} \left(\min_{y \in \mathbf{Y}} d(x, y) \right) \\ hd(\mathbf{Y}, \mathbf{X}) &= \max_{y \in \mathbf{Y}} \left(\min_{x \in \mathbf{X}} d(y, x) \right) \end{aligned} \tag{3.13}$$

where d denotes a certain type of metric between points, such as the Euclidean distance. In simple terms, Treiber (2010, p. 51) mentions that "the forward distance $hd(\mathbf{X}, \mathbf{Y})$ can be determined by calculating the distance to the nearest point of $\mathbf{Y}$ for each point of $\mathbf{X}$ and taking the maximum of these distances. The reverse distance $hd(\mathbf{Y}, \mathbf{X})$ is calculated by evaluating the distance to the nearest point of $\mathbf{X}$ for each point of $\mathbf{Y}$ and taking the maximum. Finally, Hd is calculated by taking the maximum of these two values". It is also interesting to note that the Hausdorff distance has the property that the total number of points in the compared polygons can be different, which is a desirable property for real world applications.

For instance, Ray & Ray (1995) and Iñesta et al. (1998) used these techniques to detect dominant points of shapes. In both cases, several shapes are analyzed including the shape of leaves.

3.1.9 Fourier descriptors

It has been well-established by the Fourier theory that a continuous function $f(t)$ can be represented as a summation of a set of sine and cosines of different frequencies by using a transformation process called the Fourier Transform. The Fourier Transform takes as input a function $f(t)$ in the time domain and changes it into a function $F(u)$ in the frequency domain (Yoo, 2001), see Appendix A.9.

In the case of digital images, where the data is not continuous nor infinite, the Fourier Transform has to be modified in order to be able to work with this type of data. This transformation is called the Discrete Fourier Transform

$$F(u) = \frac{1}{n}\sum_{t=0}^{n-1} e^{(-i2\pi ut)/n} f(t), \quad \text{for } u = 0, 1, 2, \ldots, n-1 \tag{3.14}$$

the corresponding Inverse Discrete Fourier Transform

$$f(t) = \sum_{u=0}^{n-1} e^{(i2\pi ut)/n} F(u), \quad \text{for } t = 0, 1, 2, \ldots, n-1 \tag{3.15}$$

Fourier descriptors (FDs) are another method to describe, match and recognize the outline of a shape. This technique was first introduced by Granlund in 1972 when he used them to recognize "hand printed characters". FDs take the advantages of all the Fourier methods for deriving invariant values from the shape boundary that are not affected by common transformations, such as changes in size, position and angular position (Bellmann et al., 2005; Granlund, 1972; Sarfraz et al., 2009).

For example, consider a boundary **B** composed of n Cartesian points $p = (x_k, y_k)$

$$\mathbf{B} = \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{pmatrix}$$

In order to extract the Fourier descriptors, a parametrization of these coordinates has to be carried out. This parametrization step consists of replacing the labels of each axis: where the x values are the real part of the complex number and the y values are the imaginary part (see Appendix A.7). “Although the interpretation of the sequence has been recast, the nature of the boundary itself has not been changed. The advantage of this representation is that it reduces a 2D problem into a 1D problem, i.e. you now have n complex numbers instead of $2n$ real numbers” (Yoo, 2001, p. 12).

$$u_k = x_k + iy_k, \quad \text{for each } k = 1, 2, \ldots, n \tag{3.16}$$

generated vector of complex numbers

$$\mathbf{u} = \begin{pmatrix} x_1 & + & iy_1 \\ x_0 & + & iy_2 \\ & \vdots & \\ x_n & + & iy_n \end{pmatrix}$$

After the generation of the complex vector, the Discrete Fourier Transform is applied to the vector **u**, see equation 3.14. The resulting $F(u)$ values are the *Fourier descriptors* or *Fourier coefficients* of the boundary. Of course, it is possible to recover the original boundary by applying the inverse transformation, see equation 3.15.

One of the important characteristics of this method is that it permits the reduction of the number of descriptors, allowing the possibility of working with approximate shapes (González et al., 2008). Usually, the descriptors at higher frequencies are the ones that are removed, because their contribution to the original shape is negligible.

The identification of objects is carried out by comparing the FDs of two given shapes. If a database of Fourier coefficients is created to represent the objects, then when performing the recognition step, only the one with the most similar FDs has to be found. Naturally, there are certan thresholds that limit the distance between the objects in question (Keyes & Winstanley, 1999).

The following example illustrates how Fourier descriptors are used to represent the boundary of a leaf composed of over 2000 points, see Figure 3.14 (a). Figures 3.14 (b, …, f) show how the original shape would look like if just a certain number of descriptors are used.

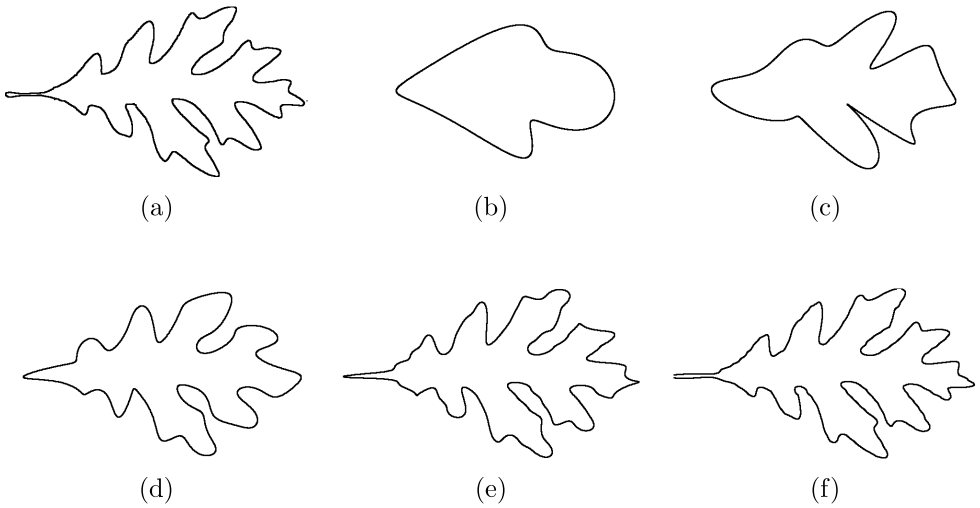


Figure 3.14: Example of shape reconstructions using Fourier descriptors (FDs). (a) The original shape boundary is a leaf composed of over 2000 points. Additionally, shape reconstruction is shown using the most important descriptors (b) 5 FDs, (c) 10 FDs, (d) 25 FDs, (e) 50 FDs and (f) 100 FDs.

The aforementioned images clearly show how FDs are able to reduce the amount of information provided by the original boundary and still recover most of the original form. It is worth noting that the performance of FDs to match shapes is very efficient when there are no irregularities (such as noise and occlusions) that distort the shape boundary; if undesired irregularities are present, the performance of FDs is affected.

Fourier descriptors are a widely used technique to identify shapes, for example, Mebatsion & Paliwal (2011) implemented a system that separates touching kernels, Lootens et al. (2007) porposed a system that recognizes roots of *Chicorium intybus*, Neto et al. (2006) uses this technique to identify the leaves of different plant species, Rocha et al. (2010) proposed a method to classify fruits and vegetables.

3.2 Elliptical representation of shapes

This section briefly describes the 2D ellipse and the 3D ellipsoid. The idea behind presenting these geometric shapes is to review basic geometry concepts that will help in understanding the procedures that are introduced in further chapters.

Ellipses and ellipsoids are used to represent the original shape of leaves and complex amorphous structures. For example, in the 2D space, approximations were carried out because the shape of many leaves can be approximated using ellipses, i.e., elongated leaves or almost circular leaves.

In relation to this subject, Courtney & dePaor (2004, p. 1) mentioned an interesting remark: “Many biological and astronomical forms can be best represented by ellipses. While some more complex curves might represent the shape more accurately, ellipses have the advantage that they are easily parameterised and define the location, orientation and dimensions of the data more clearly.”

3.2.1 The ellipse

The ellipse is a closed curve whose points have the property that the sum of the distances to two fixed points, or foci, is constant. An ellipse can be generated by intersecting a right circular cone with a plane that is not parallel to the axis, base, or generatrix of the intersected cone, see Figure 3.15, and thus, it is considered to be a conic section.

(a) right circular cone

(b) right circular cylinder

Figure 3.15: This figure shows a plane intersecting a right circular cone to create a conic section or ellipse. It also depicts an ellipse generated by the intersection of a right circular cylinder and a plane.

An ellipse has two radii: one called the semimajor axis a and the other called the semiminor axis b, with the major axis being $2a$ and the minor axis $2b$. A characteristic of all ellipses is that they contain fixed points $\tilde{f}_1$ and $\tilde{f}_2$ (called foci), which are located on the major axis and whose sum of the distances $d_1 + d_2$ to a point p on the perimeter remains a constant value ($2a$). The distance between the center and either one of the foci is the value $\tilde{c}$, with the focal length being the distance between the two foci $2\tilde{c}$ (Weisstein, 2010a), see Figure 3.16.

Another interesting concept is the eccentricity, which measures how much an ellipse deviates from being perfectly round, i.e. a circle. The eccentricity is denoted as $\hat{e} = \frac{\tilde{c}}{a}$ with a range of $[0, 1)$, with 0 being perfectly circular and 1 a highly elongated ellipse. A circle, in fact, can be considered an ellipse, where the two radii a and b are equal.

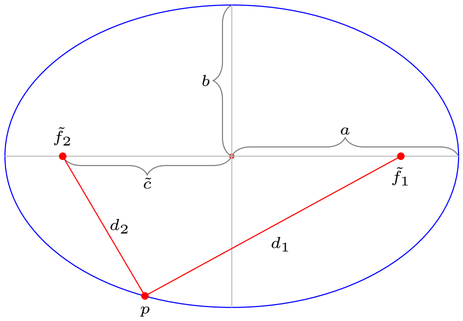


Figure 3.16: Labelled diagram of an ellipse.

3.2.1.1 General equations

In general, conic sections are described by the second degree polynomial shown in equation 3.17. Nevertheless, ellipses have a restriction that constrains the possible values of the constant terms (Spiegel et al., 2010).

$$Ax^2 + By^2 + Cxy + Dx + Ey + F = 0 \tag{3.17}$$

where A, B, C, D, E and F are constant terms with restriction: $C^2 - 4AB < 0$

In its canonical form[1], ellipses have the following Cartesian equation:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad \text{where: } a > 0 \text{ and } b > 0 \tag{3.18}$$

If the center of an ellipse is shifted to the position (x_c, y_c), the equation 3.18 changes to:

$$\frac{(x - x_c)^2}{a^2} + \frac{(y - y_c)^2}{b^2} = 1 \tag{3.19}$$

An ellipse that is not aligned with the axes can be generated by rotating all its points (x, y) by an angle θ in order to obtain corresponding rotated points (x', y'). This is achieved by using the following linear algebra operation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.20}$$

or the equivalent:

$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned} \tag{3.21}$$

---

[1] The canonical form or normal form of an ellipse refers to the set of ellipses with center at the Cartesian origin $(0, 0)$ and aligned with the coordinate axes (Weisstein, 2010a).

If these rotated values are substituted into equation 3.19, the following equation is obtained:

$$\frac{(x\cos(\theta) - y\sin(\theta) + x_c)^2}{a^2} + \frac{(x\sin(\theta) + y\cos(\theta) + y_c)^2}{b^2} = 1 \tag{3.22}$$

Expanding equation 3.22, it is possible to arrive at the equivalent system:

$$\begin{split}(b^2\cos^2(\theta) + a^2\sin^2(\theta))x^2 &+ \\ (b^2\sin^2(\theta) + a^2\cos^2(\theta))y^2 &+ \\ 2\cos(\theta)\sin(\theta)(a^2 - b^2)xy &+ \\ 2(b^2\cos(\theta)x_c + a^2\sin(\theta)y_c)x &+ \\ 2(a^2\cos(\theta)x_c - b^2\sin(\theta)y_c)y &+ \\ (b^2x_c^2 + a^2y_c^2 - a^2b^2) &= 0\end{split} \tag{3.23}$$

After arranging the coefficients of the above equation and looking at the coefficients of the general equation of an ellipse (see equation 3.17), the following associations can be made:

$$\begin{aligned} A &= b^2\cos^2(\theta) + a^2\sin^2(\theta) \\ B &= b^2\sin^2(\theta) + a^2\cos^2(\theta) \\ C &= 2\cos(\theta)\sin(\theta)(a^2 - b^2) \\ D &= 2(b^2\cos(\theta)x_c + a^2\sin(\theta)y_c) \\ E &= 2(a^2\cos(\theta)x_c - b^2\sin(\theta)y_c) \\ F &= b^2x_c^2 + a^2y_c^2 - a^2b^2 \end{aligned} \tag{3.24}$$

It is clear that when a ellipse has its center at the origin $x_c = 0$ and $y_c = 0$ the values of D and E are zero, thus simplifying the general equation to:

$$Ax^2 + By^2 + Cxy + F = 0 \tag{3.25}$$

One can further simplify the above equation by dividing each term in the equation by the constant term F:

$$\tfrac{A}{F}x^2 + \tfrac{B}{F}y^2 + \tfrac{C}{F}xy + 1 = 0 \tag{3.26}$$

It is worth noting that the constant terms of the general quadratic equation (equation 3.17) help to identify the type of the conic section. In this case, it is essential to find out if these terms produce an ellipse or other type of conic section, i.e., parabola or hyperbola. The identification is carried out with the help of the following matrices (Seufer & Späth, 1999; Weisstein, 2010a):

$$\mathbf{M} = \begin{bmatrix} A & \frac{1}{2}C & \frac{1}{2}D \\ \frac{1}{2}C & B & \frac{1}{2}E \\ \frac{1}{2}D & \frac{1}{2}E & F \end{bmatrix} \qquad \mathbf{N} = \begin{bmatrix} A & \frac{1}{2}C \\ \frac{1}{2}C & B \end{bmatrix} \qquad \mathbf{O} = \begin{bmatrix} A+B \end{bmatrix}$$

where A, B, C, D, E, F are the constant terms in equation 3.17.

It is an ellipse if and only if the determinants of the matrices fulfill the following conditions (Weisstein, 2010a):

$$det(\mathbf{M}) \neq 0 \qquad det(\mathbf{N}) > 0 \qquad \frac{det(\mathbf{M})}{\mathbf{O}} < 0$$

After making sure that the polynomial is in fact an ellipse, one can also use the same constant terms A, B, C, D, E, F in the equation to calculate the parameters of the standard equation of an ellipse: semimajor axis a, semiminor axis b, the center (x_c,y_c) and the rotation angle θ (Weisstein, 2010a).

$$\text{rotation angle } \theta = \begin{cases} 0 & \text{if } C = 0 \text{ and } A < B \\ \frac{\pi}{2} & \text{if } C = 0 \text{ and } A > B \\ \frac{1}{2}\arctan\left(\frac{C}{A-B}\right) & \text{if } C \neq 0 \text{ and } A < B \\ \frac{\pi}{2} + \frac{1}{2}\arctan\left(\frac{C}{A-B}\right) & \text{if } C \neq 0 \text{ and } A > B \end{cases}$$

The center (x_c, y_c):

$$x_c = \frac{2BD - CE}{C^2 - 4AB} \qquad y_c = \frac{2AE - CD}{C^2 - 4AB}$$

The semimajor axis a, semiminor axis b:

$$a = \sqrt{\frac{2(AE^2 + BD^2 + FC^2 - 4CDE - 4ABF)}{(C^2 - 4AB)[\sqrt{(A-B)^2 + C^2} - (A+B)]}}$$

$$b = \sqrt{\frac{2(AE^2 + BD^2 + FC^2 - 4CDE - 4ABF)}{(C^2 - 4AB)[-\sqrt{(A-B)^2 + C^2} - (A+B)]}}$$

Having calculated the values of the semimajor axis a and the semiminor axis b, one can proceed to compute the value of the focal distance $\tilde{c}$.

$$\tilde{c} = \sqrt{a^2 - b^2} \tag{3.27}$$

Also note that “the chord through a focus parallel to the conic section directrix of a conic section is called the latus rectum, and half this length is called the semilatus rectum” (Coxeter, 1989, p. 116). Figure 3.17 shows how to locate the latus rectum Ł and the similatus rectum ł.

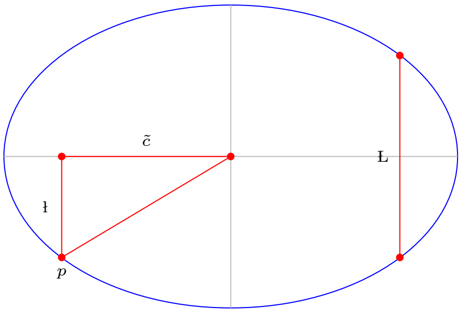


Figure 3.17: Latus rectum Ł and semilatus rectum ł of an ellipse. The distance between the center and either one of the foci is the value $\tilde{c}$. The point p is where the semilatus rectum touches the ellipse.

In order to calculate the value of the semilatus rectum one needs to use equation 3.28, where ł is the semilatus rectum, a semimajor axis, and $\hat{e}$ is the ellipse's eccentricity.

$$\text{ł} = a(1 - \hat{e}^2) \tag{3.28}$$

3.2.1.2 Parametric equations

The general parametric equation of an ellipse with center at (x_c, y_c) is (Weisstein, 2010a)

$$\begin{aligned} x &= x_c + a\cos(\omega) \\ y &= y_c + b\sin(\omega) \end{aligned} \tag{3.29}$$

In order to rotate an ellipse, one can use the rotation operation shown in equation 3.20. In addition, it is worth noting that a and b are interchangeable within the equations. Although, by convention, a is the always the largest value.

3.2.1.3 Ellipse fitting

This subsection attempts to clarify the general idea of fitting an ellipse to a set of points, as is depicted in Figure 3.18.

Assuming that a set of points $\mathbf{P} = \{p_1, p_2, \ldots, p_n\}$ belongs to an ellipse, where $p_i = (x_i, y_i)$ and n is the number of elements in the set, it is possible to find the ellipse that best fits this set of points by using the general quadratic equation of an ellipse and the least squares method (Fitzgibbon et al., 2002).

Basically, all that has to be done is to find the solution to a system of equations of the type:

$$\mathbf{A}\,\boldsymbol{x} = \boldsymbol{b} \tag{3.30}$$

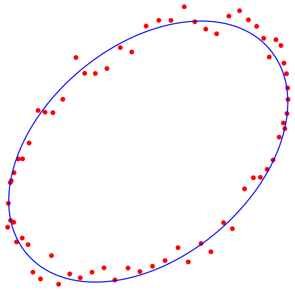


Figure 3.18: Fitting an ellipse to a given set of points.

where $\mathbf{A}$ is the coefficient matrix, $\boldsymbol{x}$ is the solution vector, and $\boldsymbol{b}$ is the constant vector. This system may or may not have a solution, nevertheless, using the least squares approximation (Meyer, 2000, p. 223). one can find a solution vector $\hat{\boldsymbol{x}}$ that minimizes the distance to the vector $\boldsymbol{b}$:

$$\min |\boldsymbol{b} - \mathbf{A}\,\hat{\boldsymbol{x}}| \tag{3.31}$$

In order to create the matrices $\mathbf{A}$, $\hat{\boldsymbol{x}}$ and $\boldsymbol{b}$, it is necessary to recall equation 3.17, where the general polynomial of ellipses is given:

$$\underbrace{\begin{bmatrix} x_1^2, & y_1^2 & x_1y_1 & x_1 & y_1 & 1 \\ x_2^2, & y_2^2 & x_2y_2 & x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2, & y_n^2 & x_ny_n & x_n & y_n & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} A \\ B \\ C \\ D \\ E \\ F \end{bmatrix}}_{\hat{x}} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{b}$$

Fitzgibbon et al. (1998, p. 3) mentions that “in order to avoid the trivial solution $\hat{\boldsymbol{x}}$=0, and recognizing that any multiple of a solution $\hat{\boldsymbol{x}}$ represents the same conic, the parameter vector $\hat{\boldsymbol{x}}$ is constrained in some way.”

Finally, the set of least squares solutions of $\mathbf{A}\boldsymbol{x}$=$\boldsymbol{b}$ is given by the set of all solutions of the normal equation (Meyer, 2000, p. 233)

$$\mathbf{A}^{\mathsf{T}}\mathbf{A}\,\hat{\boldsymbol{x}} = \mathbf{A}^{\mathsf{T}}\boldsymbol{b} \tag{3.32}$$

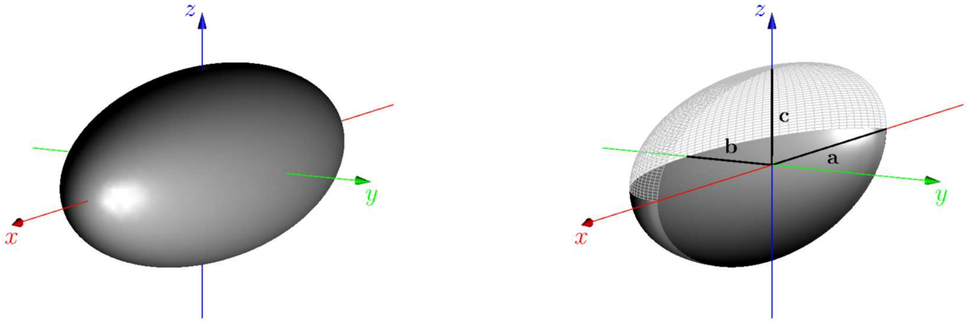
the unknown variables in the $\hat{\boldsymbol{x}}$ solution vector can be computed with

$$\hat{\boldsymbol{x}} = (\mathbf{A}^{\mathsf{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathsf{T}}\boldsymbol{b} \tag{3.33}$$

The matrix $\mathbf{A}^{\mathsf{T}}\mathbf{A}$ is invertible if and only if the columns of $\mathbf{A}$ are linearly independent. If the matrix $\mathbf{A}^{\mathsf{T}}\mathbf{A}$ is invertible, then the singular value decomposition (SVD) method can be used to compute the solution (see Appendix A.6).

3.2.2 The ellipsoid

A three-dimensional ellipse or an ellipsoid is part of the family of quadratic surfaces. This surface is very similar to the two dimensional ellipse. Nevertheless, the ellipsoid has 3 radii (a, b, c), one for every dimension.



(a) Full Ellipsoid

(b) Half ellipsoid

Figure 3.19: This figure depicts a full ellipsoid and the same ellipsoid cut in half in order to show the radii a, b and c.

3.2.2.1 General equations

The following polynomial represents all quadratic surfaces including ellipsoids (Weisstein, 2010b).

$$\begin{gathered} Ax^2 + By^2 + Cz^2 + Dxy + Exz + Fyz + Gx + Hy + Iz + J = 0 \\ \text{where } A, B, C, D, \ldots, J \text{ are constant terms of the equation.} \end{gathered} \tag{3.34}$$

Identifying ellipsoids from the previous general equation is possible by analyzing some properties of the following matrices (Weisstein, 2010b)

$$\mathbf{M} = \begin{bmatrix} A & \frac{1}{2}D & \frac{1}{2}E \\ \frac{1}{2}D & B & \frac{1}{2}F \\ \frac{1}{2}E & \frac{1}{2}F & C \end{bmatrix} \qquad \mathbf{N} = \begin{bmatrix} A & \frac{1}{2}D & \frac{1}{2}E & \frac{1}{2}G \\ \frac{1}{2}D & B & \frac{1}{2}F & \frac{1}{2}H \\ \frac{1}{2}E & \frac{1}{2}F & C & \frac{1}{2}I \\ \frac{1}{2}G & \frac{1}{2}H & \frac{1}{2}I & J \end{bmatrix}$$

where the terms $A, B, C, D, \ldots, J$ are the constant terms in equation 3.34.

A given quadratic polynomial forms an ellipsoid if and only if the ranks of the matrices $\mathbf{M}$ and $\mathbf{N}$ are: $rank(\mathbf{M}) = 3$, $rank(\mathbf{N}) = 4$; the determinant of the matrix $\mathbf{N}$ is negative $det(\mathbf{N}) < 0$; and all the calculated eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$ of the matrix $\mathbf{M}$ have the same sign.

The general Cartesian equation of an ellipsoid with the center shifted to a (x_c, y_c, z_c)

position is (Weisstein, 2010b)

$$\frac{x^2 + x_c}{a^2} + \frac{y^2 + y_c}{b^2} + \frac{z^2 + z_c}{c^2} = 1 \tag{3.35}$$

3.2.2.2 Parametric equation

The parametric equation of ellipsoids with their center at the point (x_c, y_c, z_c) is

$$\begin{aligned} x &= x_c + a\sin(\beta)\cos(\omega) \\ y &= y_c + b\sin(\beta)\sin(\omega) \\ z &= z_c + c\cos(\beta) \end{aligned} \tag{3.36}$$

where the parametric angle ω varies from 0 to 2π and β varies from 0 to π (Weisstein, 2010b).

3.2.2.3 Ellipsoid fitting

Fitting an ellipsoid to a set of (x, y, z) points is done in the same way as in the two dimensional fitting. However, in this case, it is necessary to find the solution using equation 3.34.

3.3 Active Shape Models

The goal of this section is to present Active Shape Models (ASMs), which is the primary identification model used within this work. The basic concepts and mathematical background are introduced, as well as some basic examples that aim to help the reader gain a better understanding of these particular types of models.

ASMs are part of the commonly named Statistical Shape Models because they incorporate the statistical information obtained from a learning set into the model templates. Furthermore, these templates have the capability of being altered to create new shapes which are similar to the ones included in the original learning set. Being able to generate new shapes is a very powerful and desirable characteristic because the deformable templates are not limited to the forms in the learning set, thus allowing the models to be applied in unknown and different circumstances.

Before going any further into the details of ASMs, it is worthwhile to define the terms *statistics* and *statistical inference*. In the dictionary (Farlex, 2010), statistics is defined as “a numerical value, such as standard deviation or mean, that characterizes the sample or population from which it was derived”, while statistical inference is defined as “the theory, methods, and practice of forming judgments about the parameters of a population”.

The aforementioned terms form the basis of a general understanding of how ASMs work. The first requirement is a representative set of similar objects, or population sample, that one wants to identify. This sample will be statistically analyzed in order to identify certain characteristics, which in turn are going to define the model or template that defines the population set. The template can be used to ascertain if other objects can be grouped within the same object set. Cootes et al. (1995), mention that in order for these templates to work, they have to be able to “deform in ways which are characteristic of the objects they represent”, and that the only way to achieve this is by “learning the specific patterns of variability of the structures to be modeled”.

Thus far, it has been mentioned that ASMs need a set of similar objects to generate a model. However, it has not been established the way that these objects need to be represented in order to use this method. The reader should note that ASMs use shape boundaries to perform object matching. Of course, this statistical approach is not limited to shapes, it can also be applied to the texture of objects and other characteristics. In the same way, it is possible to combine different characteristics to create a model - such as combining shape and texture - to produce a type of statistical model called Active Appearance Models or AAMs (Cootes & Taylor, 2004, p. 55).

3.3.1 Shape representation and landmarks

Shape boundaries can be represented as a set of k points p; where $p = (x, y)$ in the 2D space, and $p = (x, y, z)$ in the 3D space. When all points that belong to a specific *shape* $\mathbf{s}$ are assembled as a vector, it is expressed as

$$\mathbf{s} = \{p_1, p_2, p_3, \ldots, p_k\} \tag{3.37}$$

where each point p is a n-tuple $\in \mathbb{R}^n$. Note, however, that the attributes of an object are not limited to its shape and many other characteristics can be combined. Furthermore, a given *set of shapes* $\mathcal{S}$ that includes m single shapes is denoted as

$$\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \ldots, \mathbf{s}_m\} \tag{3.38}$$

ASMs require that all shapes belonging to the same class have the same number of points to represent the characteristics of the outline. On digital images, the size of the extracted contours can vary from image to image, depending on several factors, such as camera position, light conditions, noise, etc. The problematic then arises that one needs to find ways to represent a shape boundary with the same number of points.

The process of finding the points of reference in all shape boundaries is called *landmarking*. Originally, landmarks were used to denote an important point on a geographic area, but in the case of ASMs, they are used to indicate dominant points with similar positions in all of the boundaries of the same object class. In relation to this particular problem, Chetverikov (2003) indicated that “if these characteristic contour points are identified properly, a shape can be represented in an efficient and compact way”.

Essentially, the most desirable attribute when landmarking an object is to locate its main features with a reduced number of landmarks. This processes can be carried out manually or automatically by detecting the highest curvature points in each shape.

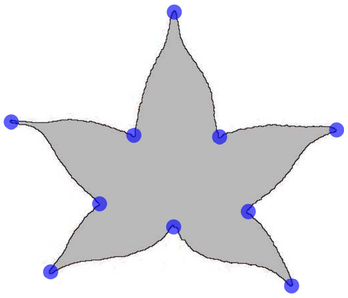


Figure 3.20: Landmarks placed on a leaf of a Sweetgum tree (*Liquidambar styraciflua*).

As an example, Figure 3.20 shows the shape of a leaf that was manually landmarked, where a total of 10 landkmarks were extracted. Note that if a greater number of landkmarks are used, the leaf boundary will show more details. Hence, a trade-off between compactness and detail arises.

3.3.2 The correspondence problem

Each landmark has to be labeled in such a way that each one represents a specific part of the shape. Moreover, when trying to compare two or more objects that belong to the same class, all landmarks with the same label have to correspond to the same physical region.

If the correspondence between landmarks is not well-established, the created model will lead to inappropriate deformations and will compromise the accuracy of the identification process. Therefore, finding the right correspondence between landmarks is of great importance when using ASMs.

The following picture, Figure 3.21, is a continuation of the example shown in Figure 3.20, although this time the chosen landmarks are labeled in order to set a reference point for each particular region of the leaf. In this particular example, the landmark labeled with the number 1 was placed where the petiole touches the leaf. Then, following a counterclockwise direction, the rest of the landmarks were placed where the curvature is very high.

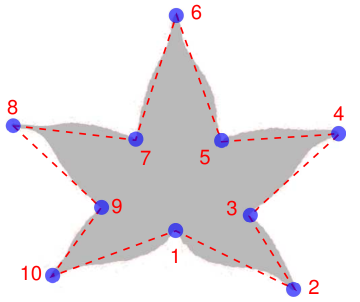


Figure 3.21: Labeling landmarks to establish correspondence.

In order to build a statistical model using a training set of shapes, the same landmarking and labeling scheme has to be followed for all of the samples. Figure 3.22 depicts an example of a training set, which contains different shapes of the Sweetgum leaves.

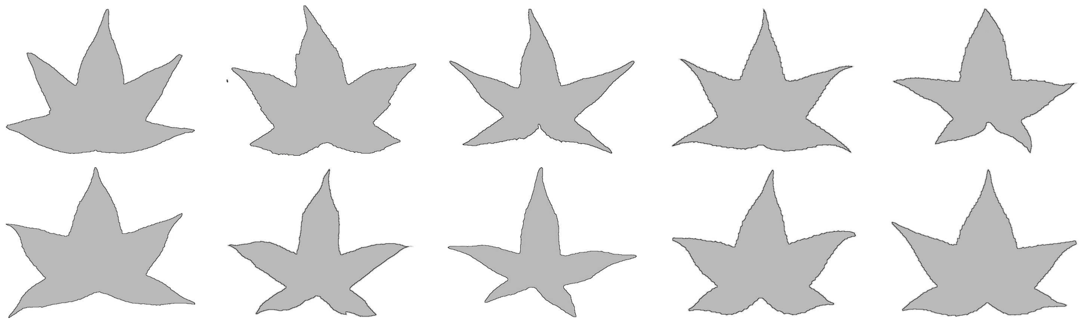


Figure 3.22: ASM training set using Sweetgum leaves.

Moreover, when using the model to perform recognition on shapes not included in the training set, one has to make sure that these landmarks correspond to the same object region as the ones in the model. Otherwise the model will most likely fail to match an object of the same class.

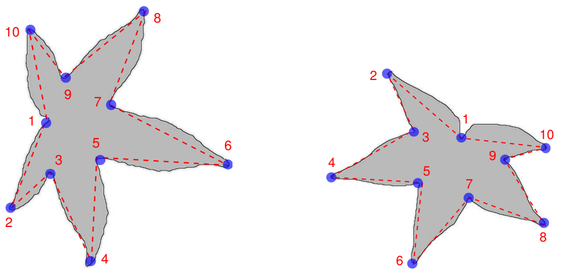


Figure 3.23: Example of a consistent landmarking scheme on shapes that do not belong to the training set from which the model was created.

Figure 3.23 shows how two leaves that do not belong to the model training set were consistently landmarked according to the scheme that was described in the above example (see Figure 3.21). One can see that despite the rotation, translation, and scale aspects, each landmark was placed in their corresponding region.

3.3.3 Shape alignment

The next step in the model generation is to align all the landmarks extracted from the training set in such a way that the distance between them is minimized. This arrangement is required by the statistical method in order to properly perform a distribution analysis on the landmark configurations.

Analyzing aspects of variability between the objects' shapes requires that the affects of geometric transformations on shapes are eliminated.

Procrustes analysis

A popular method to carry out this task is the Procrustes analysis or Procrustes superposition (Algorithm 3.3), extensively studied by Schönemann (1966) and Gower (1975). Below is a detailed description of this algorithm when applied to a given set of shapes (Cootes & Taylor, 2004; Cootes et al., 1995; Davies et al., 2008; Taylor et al., 1992).

Algorithm 3.3 PROCRUSTES ANALYSIS($\mathcal{S}$)

```
Input: A sequence of shapes $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \ldots, \mathbf{s}_m\}$.
Eliminate the translational component in all $m$ shapes by moving the centroid of each shape $\mathbf{s}_i$ to the origin.
Eliminate the scale component by normalizing each shape $\| \mathbf{s}_i \| = 1$.
Select a reference shape within the $m$ shapes in the data set $\mathcal{S}$; set it as the current mean shape $\bar{\mathbf{s}}$.
repeat
    Rotate each landmarked plant $\mathbf{s}_i$ towards the current mean shape $\bar{\mathbf{s}}$ so that the sum of the squared distances between the points of $\mathbf{s}_i$ and $\bar{\mathbf{s}}$ is minimized.
    Re-estimate the mean shape $\bar{\mathbf{s}}$ using all the previously aligned shapes $\mathbf{s}_i$. Normalize the mean shape $\| \bar{\mathbf{s}} \| = 1$.
until there is no change in the new mean shape $\bar{\mathbf{s}}$, or the changes are so small that they can be neglected.
Output: An aligned set of shapes $\mathcal{S}$, the mean shape $\bar{\mathbf{s}}$.
```

Procrustes analysis starts with the elimination of the translational divergences by subtracting from each shape its center of mass. This step is followed by a simple normalization so that all of the shape samples have a standard size (vector length). Finally, an iterative set of steps is carried out that attempt to minimize the rotational component within the sample set.

The rotational step in the above algorithm is executed by computing a rotation matrix using the Singular Value Decomposition (SVD) method. This rotation matrix rotates the vector points around the centroid, which in this case is set to the origin, so as to minimize the sum of the squared distances between the corresponding landmark points in the different objects (see Figure 3.24).

Before any further description, it is important to mention the matrix *singular value decomposition* or SVD. According to Kalman (1996, p. 2), this technique "has interesting and attractive algebraic properties, and conveys important geometrical and theoretical insights about linear transformations." For more information about this method, please refer to Appendix A.6, which is based in part on the recollection of many authors, such as Akritas & Malaschonok (2004); Golub & van Loan (1996); Kalman (1996); Meyer (2000); Moler (2004); Muller et al. (2004); Stewart (1993); Strang (2005).

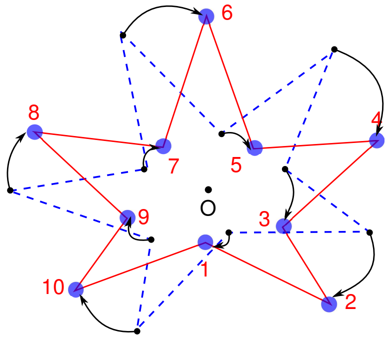


Figure 3.24: Rotation step. This image shows how a shape $\mathbf{s}_i$ (blue dashed outline) is rotated towards the mean shape $\bar{\mathbf{s}}$ (red outline). The point O represents the centroid of both shapes.

Aligning two shapes using SVD first requires the calculation of the covariance matrix. For example, shape $\mathbf{s}_i$ would be rotated to minimize the distance to shape $\mathbf{s}_j$ (it is assumed that the following vectors are centered at the origin and normalized).

$$\mathbf{s}_i = \begin{pmatrix} x_{i,1} & y_{i,1} \\ x_{i,2} & y_{i,2} \\ \vdots & \vdots \\ x_{i,n} & y_{i,n} \end{pmatrix} \qquad \mathbf{s}_j = \begin{pmatrix} x_{j,1} & y_{j,1} \\ x_{j,2} & y_{j,2} \\ \vdots & \vdots \\ x_{j,n} & y_{j,n} \end{pmatrix}$$

The covariance matrix $\mathbf{C}$ of the above vectors is

$$\mathbf{C} = \mathbf{s}_i^{\mathsf{T}} \mathbf{s}_j, \tag{3.39}$$

Using SVD the covariance matrix $\mathbf{C}$ can be factorized as (Muller et al., 2004, p.

523)

$$\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}} \tag{3.40}$$

The rotation matrix $\mathbf{R}$ that would rotate $\mathbf{s}_i$ towards $\mathbf{s}_j$ is, see Muller et al. (2004, p. 512) for a geometric interpretation.

$$\mathbf{R} = \mathbf{V}\mathbf{U}^{\mathsf{T}} \tag{3.41}$$

The rotated vector $\mathbf{r}_i$ is computed with the following operation

$$\mathbf{r}_i = (\mathbf{R}\,\mathbf{s}_i^{\mathsf{T}})^{\mathsf{T}} \tag{3.42}$$

The reader should note that this transformation can alter the landmark correspondence. Precautions should be taken in order to maintain the correspondence between the landmarks in the shape to be rotated and those in the template.

Moving forward with the example of the Sweetgum leaves, the following image (Figure 3.25) presents a plot that shows how the training set of landmarked leaves looks like after running the Procrustes algorithm. This image also depicts the calculated mean shape.

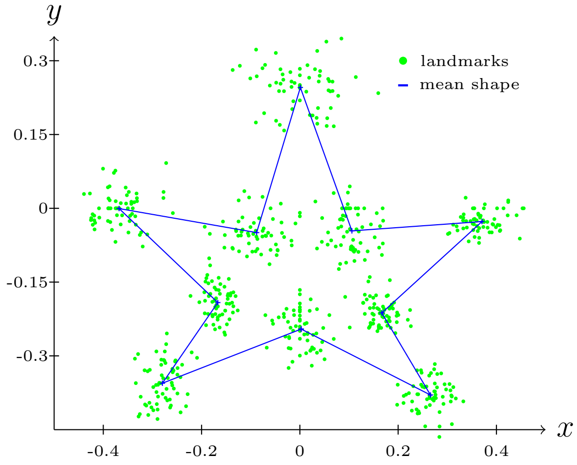


Figure 3.25: Example of aligned landmarks and mean shape. A set of 50 Sweetgum leaves was used to generate the plot presented here.

3.3.4 Shape model statistical analysis

Generating a prototype shape that is able to deform into shapes similar to the ones included in the training set requires an statistical analysis that learns from the variations and similarities within the given learning data. Naturally, having this type of model provides great potential for a pattern recognition tool because in reality, shapes vary from one situation to an other.

The objective of the statistical analysis is to model the variations within the selected data set (in this case landmarks). For instance, in Figure 3.25, it is possible to identify clouds of points around the landmarks that belong to the mean shape. To some extent, each cloud of points represents the area where the model can be deformed. Each one of these areas contains characteristic patterns of variation that have to be incorporated into the shape model after analyzing the data.

Principal Component Analysis (PCA) (Duda et al., 2001, p. 568) is a great statistical tool that allows the characterization of existent patterns in a given population group, it also permits the identification of noise and enables dimensionality reduction. PCA works hand in hand with SVD to find these main characteristics or principal components so that each member in the data set can be approximated with a model that has less parameters than in the original data set.

In order to compute the principal components of a set $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \ldots, \mathbf{s}_m\}$ of landmarked shapes, a series of steps are required. It is assumed that the set $\mathcal{S}$ was previously aligned and normalized using a method, such as the one described in Algorithm 3.3.

Algorithm 3.4 PRINCIPAL COMPONENT ANALYSIS ($\mathcal{S}$)

Input: A sequence of shapes $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \ldots, \mathbf{s}_m\}$.

1: The proper arrangement of a matrix is necessary, which contains the data of each shape in the form of a row matrix. One has to make sure that the d-dimensions (where $d > 0$) of each shape are correctly included.

For example, all 2-dimensional shapes $\mathbf{s}_i$ have to be put into a specific order, so that all the data from their n landmarks are included in a row matrix with a total of $2n$ elements

$$\mathbf{s}_i = (x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) \tag{3.43}$$

for 3-dimensional shapes $\mathbf{s}_i$ with n landmarks the row matrix can be arranged as follows

$$\mathbf{s}_i = (x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n, z_1, z_2, \ldots, z_n) \tag{3.44}$$

this new arrangement is used to create a matrix $\mathbf{A}$ of size $m \times dn$, where d is the number of dimensions, m is the number of shapes in the training set, and n is the number of landmarks. Each of the dn columns of $\mathbf{A}$ represent specific attributes of each shape.

$$\mathbf{A}_{m,dn} = \begin{array}{c} \longleftarrow \text{attributes} \longrightarrow \\ \left(\begin{array}{ccccc} \delta_{1,1} & \delta_{1,2} & \delta_{1,3} & \cdots & \delta_{1,dn} \\ \delta_{2,1} & \delta_{2,2} & \delta_{2,3} & \cdots & \delta_{2,dn} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \delta_{m,1} & \delta_{m,2} & \delta_{m,3} & \cdots & \delta_{m,dn} \end{array}\right) \end{array} \begin{array}{c} \\ \text{shape}_1 \\ \text{shape}_2 \\ \vdots \\ \text{shape}_m \end{array} \tag{3.45}$$

2: The mean vector $\boldsymbol{\mu} = \{\mu_1, \mu_2, \ldots, \mu_{dn}\}$ has to be calculated in order to move the center of each attribute to the origin. Equation 3.46 shows how to calculate each μ_j, which after their computation have to be subtracted from their corresponding column

$$\mu_j = \frac{1}{m} \sum_{i=1}^{m} \delta_{ij}, \quad \text{for each } \{j = 1, \ldots, dn\} \tag{3.46}$$

3: Compute the covariance matrix.

$$\mathbf{C}_{dn,dn} = \frac{1}{m} \mathbf{A}^{\mathsf{T}} \mathbf{A} \tag{3.47}$$

Remember that the covariance matrix contains the variance measures for each variable in the data. In this case the matrix is square, which has in its principal diagonal the variance of each variable, and in the off-diagonal, it contains the pairwise covariance measures.

4: Compute the eigenvectors and eigenvalues of the covariance matrix. The eigen-decomposition of the covariance matrix can be carried out using several methods, for instance, SVD (see Appendix A.6).

$$\mathbf{C}_{dn,dn} = \boldsymbol{\Phi} \, \boldsymbol{\lambda} \, \boldsymbol{\Phi}^{\mathsf{T}} \tag{3.48}$$

where the matrix $\boldsymbol{\lambda}$ is a diagonal matrix that contains all the eigenvalues λ_i. Furthermore, matrix $\boldsymbol{\Phi}$ contains eigenvectors $\boldsymbol{\phi}_i$ that correspond to a single eigenvalue λ_i. Eigenvectors describe the principal directions of variation of the data. This means that a single eigenvector indicates just one direction component, and its corresponding eigenvalue express the amount of variance explained by this vector in its corresponding direction. Most of the computational packages arrange the eigenvalues and eigenvectors according to the variance they represent, this means that λ_1 has the largest eigenvalue, λ_2 the second largest, etc. The following matrices depict an example of this arrangement

$$\overset{\boldsymbol{\lambda}}{\begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}} \qquad \overset{\boldsymbol{\Phi}}{\begin{pmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \\ \vdots \\ \boldsymbol{\phi}_n \end{pmatrix}}$$

where n is the total number of eigenvalues and eigenvectors.

6: It was already mentioned that each eigenvalue expresses the amount of variance explained by its corresponding eigenvector: the larger the eigenvalue is, the most important it is when recovering the data in the whole learning set. Furthermore, the overall sum of eigenvalues λ_i denotes the total variance in the set. Therefore, it is possible to express the eigenvalues as a percentage fraction of the total variance:

$$\begin{array}{cc} \text{Total variance } V_T & \text{Percentages of variance } v_i \\ V_T = \sum_{i=1}^{n} \lambda_i & v_i = \frac{\lambda_i}{V_T} \cdot 100 \end{array} \tag{3.49}$$

In order to perform dimensionality reduction, one just needs to select a k number of λ_i and their corresponding $\boldsymbol{\phi}_i$ until a desired percentage of variance V_K is reached, see equation 3.50. For instance, Cootes & Taylor (2004) suggest that a 98% of total variance is an acceptable value to represent the data. However, the user should decide this according to their specific needs.

$$V_K = \sum_{i=1}^{k} v_i \tag{3.50}$$

If dimensionality reduction is not desired, this step can be skipped.

Output: At this point, the statistical analysis is finished and the model is ready to be tested. A matrix $\mathbf{\Phi}$ contains a total of k eigenvectors that correspond to the k largest eigenvalues, which are stored in the diagonal matrix $\boldsymbol{\lambda}$. Furthermore, these elements can explain a V_K percentage of the total variance. It should be pointed out that if the dimensionality reduction step was not executed, k is still equal to the total number of eigenvalues and eigenvectors calculated by SVD.

Finally, each shape $\mathbf{s}$ in the training data set can be approximated by

$$\mathbf{s} \approx \boldsymbol{\mu} + \mathbf{\Phi}\,\mathbf{b} \tag{3.51}$$

where the vector $\mathbf{b} = \{b_1, b_2, \ldots, b_k\}$ is constituted by the k parameters that will deform the model mean shape $\boldsymbol{\mu}$. Moreover, the parameters of this vector $\mathbf{b}$, which approximate a shape $\mathbf{s}$ in the training data set is computed as follows

$$\mathbf{b} = \mathbf{\Phi}^{\mathsf{T}} (\mathbf{s} - \boldsymbol{\mu}) \tag{3.52}$$

A parameter b_i indicates how much an eigenvector $\boldsymbol{\phi}_i$ has to be scaled/streched to approximate one of the points of a given shape. This means that, in order to generate different shapes, it is just necessary to scale the eigenvectors with different b_i values.

3.3.5 Shape model deformation

Being able to create different shapes by just changing scalar values in the vector $\mathbf{b}$ provides great flexibility for the shape model. Nevertheless, limits to these scalar values have to be provided so that the newly generated shapes remain consistent to the data in the learning set.

Cootes et. al. (2004; 1995) suggests that the deformation values b_i should be bounded by the interval $[-3\sqrt{\lambda_i}, 3\sqrt{\lambda_i}]$. This means that for the previously mentioned corresponding pair $(\lambda_i, \boldsymbol{\phi}_i)$, there is a deformation value b_i bounded within the interval $[-3\sqrt{\lambda_i}, 3\sqrt{\lambda_i}]$.

$$\begin{gathered}
\{-3\sqrt{\lambda_1} \leq b_1 \leq 3\sqrt{\lambda_1}\} \\
\{-3\sqrt{\lambda_2} \leq b_1 \leq 3\sqrt{\lambda_2}\} \\
\vdots \\
\{-3\sqrt{\lambda_k} \leq b_k \leq 3\sqrt{\lambda_k}\}
\end{gathered}$$

An interesting remark about the λ_i was made by Muller et al. (2004, p. 524), “in probability theory the singular values are known as the *standard deviations* (σ).” This implies that the above ranges will only allow variations within $\pm 3\sigma$, which include 99.7% of the data.

For instance, when looking at the statistical model created for the sweetgum leaves presented in this section, new leaf shapes are generated by varying the model deformation values using equation 3.51. Figure 3.26 shows how the model responds to these changes. Note that in order to create the shapes shown in this figure, the deformation vector **b** only deforms one eigenvector at a time and sets the other vectors to zero. For example, the following vector $\mathbf{b} = [b_1, 0, 0, 0, \ldots, 0, 0, 0]$ only deforms the mean shape using the fist eigenvector $\boldsymbol{\phi}_1$, $\mathbf{b} = [0, b_2, 0, 0, 0, \ldots, 0, 0, 0]$ only deforms the mean shape using the second eigenvector $\boldsymbol{\phi}_2$, etc.

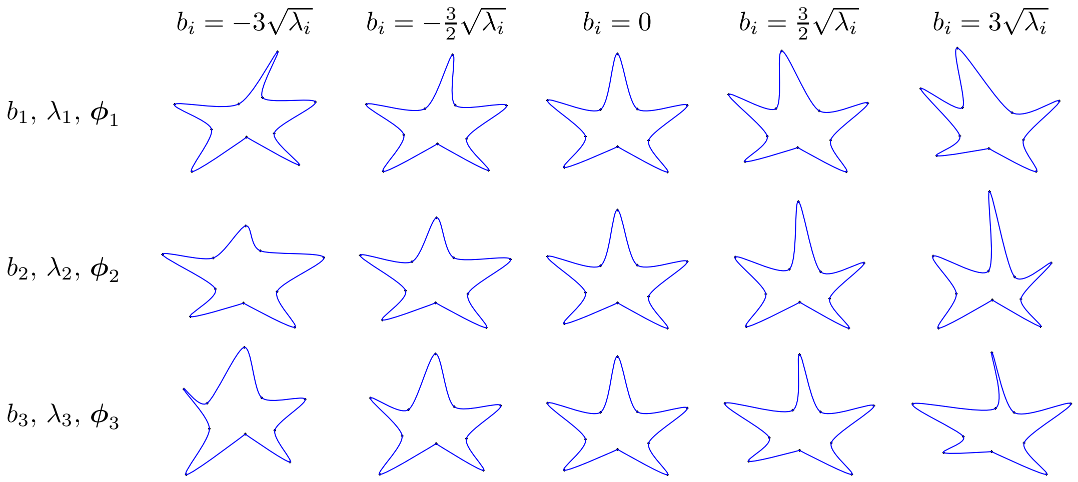

$b_i = -3\sqrt{\lambda_i}$
$b_i = -\frac{3}{2}\sqrt{\lambda_i}$
$b_i = 0$
$b_i = \frac{3}{2}\sqrt{\lambda_i}$
$b_i = 3\sqrt{\lambda_i}$
$b_1, \lambda_1, \boldsymbol{\phi}_1$
$b_2, \lambda_2, \boldsymbol{\phi}_2$
$b_3, \lambda_3, \boldsymbol{\phi}_3$

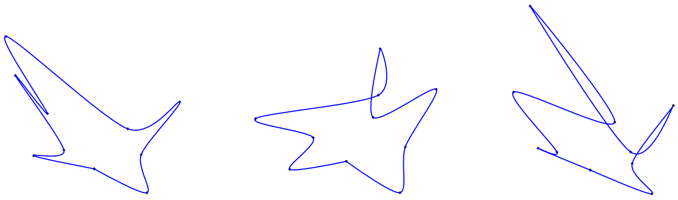
Figure 3.26: Sweetgum leaf model deformation. The deformation values b_i scale its corresponding eigenvector in order to change the mean shape.

In order to best clarify the deformations presented on Figure 3.26, Table 3.1 presents the numerical values of the 3 most important eigenvalues. This way, one can interpret the deformations of the shape model.

Eigenvalue	Value	Percentage of variance
λ_1	0.0045597	17.6489 %
λ_2	0.0032384	12.5345 %
λ_3	0.0027777	10.7515 %

Table 3.1: Numerical values of the 3 most important eigenvalues of the deformation predented in Figure 3.26.

If the deformation intervals are not considered, the model would generate shapes that are totally inconsistent with the training shapes. An example of this situation is depicted in Figure 3.27.



(a) (b) (c)

Figure 3.27: Sweetgum leaf model with wrong deformations: this image depicts two shapes that were generated using deformation values outside of the deformation model intervals. These three cases were generated using the following values: (a) $b_1 = 7\sqrt{\lambda_1}$. (b) $b_1 = 4.5\sqrt{\lambda_1}$, $b_2 = 4.5\sqrt{\lambda_2}$, $b_3 = 4.5\sqrt{\lambda_3}$. (c) $b_1 = -4.5\sqrt{\lambda_1}$, $b_2 = -4.5\sqrt{\lambda_2}$, $b_3 = -4.5\sqrt{\lambda_3}$. In each case, the values that are not indicated are set to 0.

3.3.6 Applications of ASMs

Active Shape Models have been extensively used in different areas of study, such as medical image analysis, face recognition, 2D and 3D object tracking, character recognition, etc. (Cootes et al., 1995; Davies et al., 2008).

Certainly, ASMs find most of its applications in the field of medical image interpretation, i.e., inspection of anatomic structures, the planning of surgical interventions, the examination of the progress of a disease, the examination of the progress of a healing process. Just to mention some examples: Cootes et al. (1994) proposed a model to analyze the human heart ventricle for the early detection of potential heart problems. Smyth et al. (1997) designed a model that identifies the shape of human vertebrae, this with the purpose of early detection of osteoporosis. Alien et al. (2007) also tried to detect osteoporosis by analyzing dental radiographies. Huysmans et al. (2005) generated a model to detect scoliotic deformities on humans. Toth et al. (2011) presented a method to calculate the volume of the prostate on patients using MRI images. Gregory et al. (2008) implemented a model that predicts if patients with osteoarthritis need total hip replacement. Bhat & Savvides (2008) developed models that “classify eye shapes and identify whether the images belong from left or right irises”.

Face detection is one field where ASMs are also used (Cootes & Taylor, 2004; Edwards et al., 1998; Milborrow & Nicolls, 2008). Models are created using hundreds of sample images to recognize facial features. This information can be used, on the one hand, to recognize a generic human face, and, on the other hand, to recognize a particular individual by their facial features.

Another example of the use of ASMs, this time in the agricultural field, is the positioning and tracking of animals in a farm. Onyango et al. (1995) and Tillett et al. (1997) proposed a model that locates pigs in a farm by identifying their back-ridges.

3.3.7 Applications of ASMs in plant science

In the area of plant science ASMs also have applications, such as produce grading, plant identification, etc. For instance: Søgaard & Heisel (2002) used different shape models to identify and classify weed plantlets. Moeslund et al. (2005) also employed them to identify the shape of cactus leaves. Persson & Åstrand (2008) developed shape models to classify crops and weeds on open field conditions. Kaewapichai et al. (2007) implemented grading models to detect the stage of maturity of pineapples. Srivastava et al. (2005) developed shape models to identify different objects, including apples, pears and tomatoes.

CHAPTER 4

Shape analysis using 2D ASMs for overlapping problems in plant science

4.1 Introduction to the overlapping problem

Before describing the problem at hand, it is important to define the word *overlap*. According to Farlex (2010), overlap is defined as "To lie over and partly cover something; to have an area or range in common with; to lie or extend over and cover part of".

From the above definition, it is clear that if one looks at overlapping objects, it is only possible to see a part or parts of them. One only needs to look around to find examples of things in the environment that partially occlude the view of other objects. A simple example using three geometric shapes is shown in Figure 4.1, where a square overlaps a circle and a triangle.

People commonly face these types of situations in their daily activities, nevertheless, the human vision system is so efficient that it is capable of solving such problems without us even realizing the great amount of information that has to be processed in order to do so. Scientists estimate that almost half of the cerebral cortex is used to process around 10^8 bits/s of visual data (Koch et al., 2006; Millner & Goodale, 1996; Patla, 1997).

Understanding how humans process visual information is an important step in the development of a computerized vision system. Although visual perception

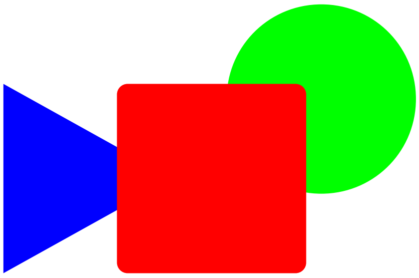


Figure 4.1: Simple overlapping example.

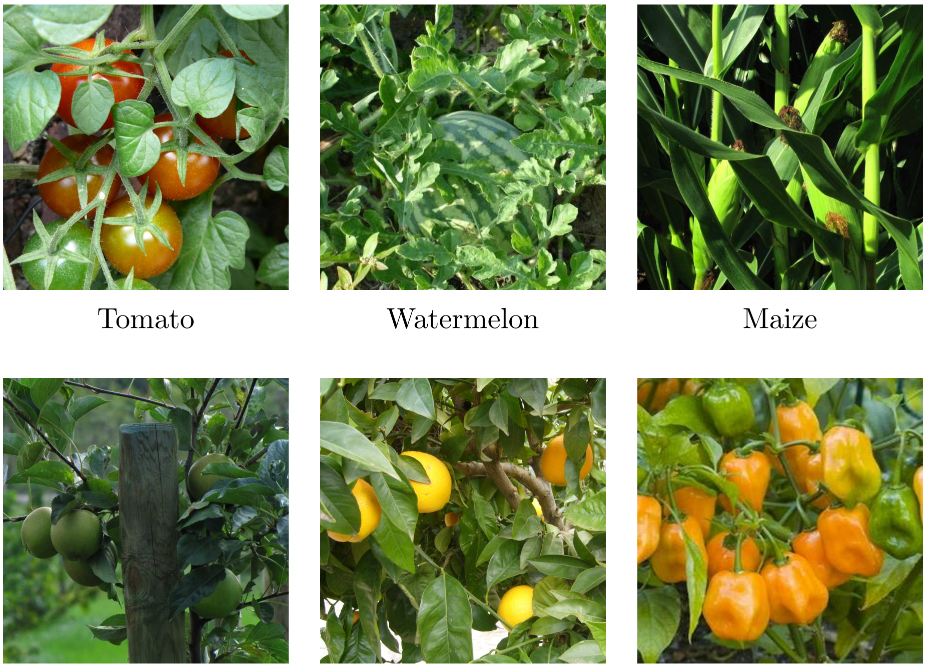
has been extensively studied, there are still various areas that remain unexplained (Culham et al., 2001; Diamant, 2008; Kellman & Spelke, 1983; Marchant & Sistler, 1993). For example, in psychology, there are several conflicting theories as to how to the human brain develops the ability to perceive individual objects in the surrounding environment. One theory is that infants in their first 2 years of life learn to identify objects by interacting (i.e. seeing, touching) with them. Based on memories of an object when seen in its entirety, the brain can then use this stored information to recreate similar objects when they are partially hidden from view. On the other hand, it has been put forward that humans are born with an inherent general conception of the physical world as a set of objects, which are spatially related but can move independently of each other (Diamant, 2008, pp. 484 -486).

Computers, however, have no inherent knowledge of the world around them. Thus, when developing a computerized vision system, the programmer has to provide a way for the computer to learn and identify individual objects and then use this information to match similar objects when partially occluded. This is not an easy task if one considers how complex and dynamic the world is. Lighting conditions, angle of view, distance from the sensor, shape and texture variance, and other factors can drastically affect the performance of such systems.

4.2 Related work

In recent years there has been a widespread interest in developing intelligent computer vision systems that will help in the automation of tedious and labor-intensive tasks. There are many different engineering disciplines that have contributed methods for solving visual recognition problems related to their specific field of study. These solutions, however, are not totally exclusive and the methods can be adapted to solve similar problems in other research fields. This also holds true for resolving the overlapping problem in biosystems and agricultural research areas.

In order to clarify the problematic, a simple example is presented below. Imagine that a robotic machine is required to help harvest crops (ex. Maize, Apples, Pepper, Oranges, Tomatoes, etc). Moreover, this machine will be equipped with a vision system as a means of tracking the goods to be harvested. While navigating through the fields, the robot has to detect the fruit and vegetables, which may not be fully in view, as the ones shown in Figure 4.2. This very fact generates a series of difficulties in the processes of recognition, positioning and manipulation, and challenge the effectiveness of the recognition system.


Tomato Watermelon Maize

Apple Orange Pepper

Figure 4.2: Examples of overlapping situations in Biosystems. It is possible to see that some of the crops are overlapped by other crops or leaves.

Machine vision has many potential uses not only within harvesting activities but also in biorobotics in general. Apart from crop localization, it can also be applied to detect weeds, to sort the products and evaluate their quality, etc. When working with living organisms and open environments, the uncertainties increase, because the information is constantly changing, and sometimes it is partially or completely hidden. Despite this, scientists are still working on finding methods to achieve the desired automation.

Like many other vision systems, the one presented in this work uses color segmentation as a means to separate regions of interest (RoI) - in this case the green of the plants or specific color of produce to be harvested - in order to identify the areas that might have some information useful for the vision recognition system. There are many methods to perform the segmentation process, just to mention some: histogram thresholding, feature space clustering, region based edge detection, fuzzy logic, neural network approaches, etc. Each strategy has its pros and cons, and depending on the characteristics of the problem, one has to adopt the most suitable method (Busin et al., 2008; Cheng et al., 2001; Comaniciu & Meer, 2002; Huang et al., 2005; Juang & Wu, 2010).

After identifying the region(s) that might contain the object(s) that one is looking for, the following step is to search for patterns that would lead to the proper identification of objects. It is in this stage where techniques, such as the ones presented in the introductory chapters, are used to recognize the shape of the objects.

One of the pioneer studies that focused on the proper identification of horticultural objects using their shape was carried out by Rath (1997). In his research, he developed and evaluated several techniques with the purpose of identifying shape characteristics of different type of plants.

However, like in many other research areas, the identification of objects is challenged by the fact that the objects that one wants to recognize are normally overlapped by other objects. This means that the identified RoIs contain information that is incomplete, and somehow the selected pattern recognition technique has to find the way to work around missing features and still be able to recognize the desired object(s).

For example, Hemming (2000), in his attempt to differentiate between crops and weeds in open field situations used geometric primitives to differentiate between plant types. Nevertheless, the performance of his approach was compromised when plants are overlapping each other. Also, Pérez et al. (2000) and Philipp (2004) were faced with the difficulty of plant mutual overlapping in their efforts to identify weeds in cereal and sugar beet fields.

When looking at harvesting robots, e.g., strawberry (Hayashi et al., 2010), cherry (Tanigaki et al., 2008), flowers (Kawollek, 2006), cucumbers (Henten et al., 2003), apples (De-An et al., 2011), etc., there is a common problematic: fruits or flowers partially occluded. This means that there are cases when the objects are not recognized, because there is missing visual information.

Other examples include the inspection and grading of agricultural products. Brosnan & Sun (2002) mention that it is possible to improve the quality of foods, fruits, vegetables and grains by means of vision systems that grade the agricultural products. Nonetheless, all the methods mentioned in his review also have the problematic that in order to cope with the production necessities, they have to be less sensitive to partly occluded objects.

Therefore the overlapping problem constitutes on of the basic problems of all biorobotic systems, and currently has not been completely solved.

4.3 Objectives

The goal of this investigation is to develop a vision system using Active Shape Models that is capable of identifying small plants growing under overlapping conditions.

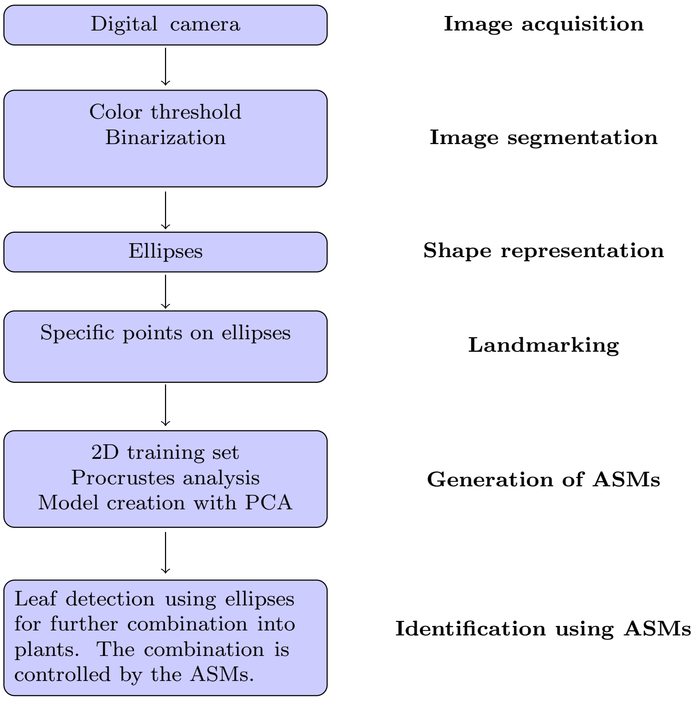
The presented thesis is concerned with plants that are composed of a set of leaves where a single leaf has, to a certain degree, an elliptical form. In this case, it is possible to present the assumption that these plants can be seen as a group of ellipses.

This technique identifies plants by first detecting individual leaves (i.e. ellipses) for further grouping into plantlets using deformable templates that contain statistical information about their main features.

Shape models of experimental plants with 2, 3 and 4 leaves were tested separately and then overlapped to analyze the ability of the method to overcome the overlapping problem.

4.4 General idea on how solve the overlaping problem

The following list of mentions the steps that this work will follow order to solve the problematic at hand. Naturally, each procedure will be described in detail in further sections.



Before going to the next section, it is important to describe its structure. There are two main topics: Experimental setup and New and adapted algorithms for plant detection. The first one will describe the elements and equipment to carry out the investigation. The second one will mention all the methods (new or adapted) used to solve the problem.

Experimental setup

Model plant
Image acquisition
Comparison with human recognition

New and adapted algorithms for plant detection

Image segmentation - RGB plant color extraction
Ellipse detection using a clustering algorithm
Ellipse detection using arcs
Landmarking procedure
Deformable plant templates
Neighboring leaves combination
Energy heuristic to match real plants

4.5 Materials and methods

4.5.1 Model plant

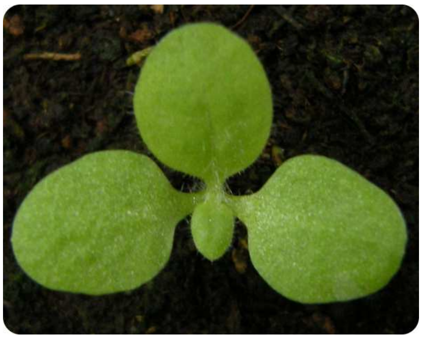
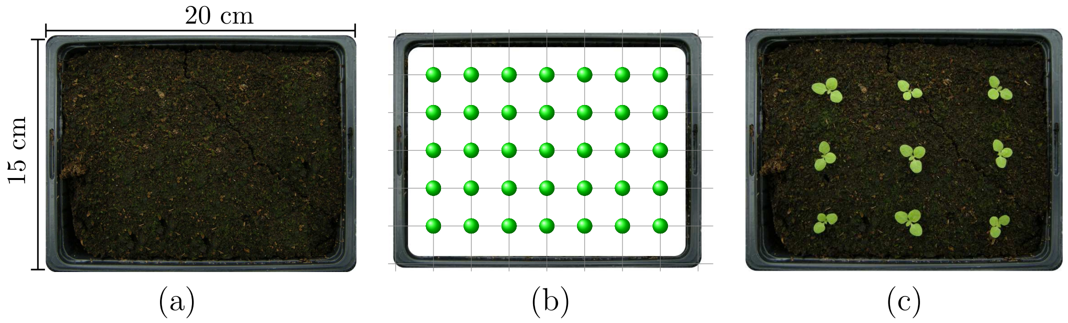


Figure 4.3: Model plant *Nicotiana tabacum*

The plant selected as a model during the experimental phase was the tobacco plant *Nicotiana tabacum*. This plant was chosen because the plantlets are fast growing, allowing a series of photos to be taken in a short period of time, with the purpose of documenting their growth. The plantlets were grown in a greenhouse with a temperature of 23 ± 2°C, in a peat-based substrate, without the use of pesticides.

The seeds were planted in a container measuring 20 cm by 15 cm, as shown in Figure 4.4a. The plants used to create the statistical model were planted in rows of 3 by 3, see Figure 4.4c, creating a non-overlapping condition area. Other trays were prepared with the seeds planted closer together, for instance in rows of 7 by 5 plants (see Figure 4.4b) or in random patterns, in order to create plants growing in overlapping conditions.



(a) (b) (c)

Figure 4.4: Seeding the tobacco plantlets. (a) Trays used. (b) Example of a seeding pattern used to create overlapping. (c) Example of plants grown without overlapping.

Table 4.1 presents a description of the arrangements used to sow the seeds of tobacco.

Number of trays	Tray description
6	Model plants, only 9 plants per tray, no mutual overlapping
3	Overlapping plants, in rows of 6 by 5 plants
3	Overlapping plants, in rows of 6 by 4 plants
2	Overlapping plants, in rows of 5 by 5 plants
2	Overlapping plants, in rows of 5 by 4 plants
2	Overlapping plants, randomly spread over the sowing area

Table 4.1: Arrangement of plants within the tray area

4.5.2 Comparison with human recognition

The results of the plant detection algorithm are compared with the human ability to identify plantlets under overlapping situations. The experiments were carried out in June 2011 using two groups of persons: *Experts* and *Non-experts*. The first group (experts) was composed of 10 Ph.D. students from the Biosystems and Horticultural Engineering Section (BGT), Leibniz University of Hannover. The second group was composed of 8 trade apprentices (non-experts) between the ages of 15 and 16 years old. The experiments with this group were carried out at the beginning of the class session under the supervision of the teacher.

In order to carry out the experiments each person was given a printed version of the overlapping plant situations that the ASM system tried to solve. Note that the information presented to each person was in a black and white format (1 situation per sheet of paper), something very similar to the cases shown in Figure 4.6, which depicts cases of varying degrees of difficulty. Then, 3 color images of individual plants (2 , 3 and 4 leaves) were shown to the participants so that they were able to identify the plants, see Figure 4.5. Finally, the participants were asked to find as many plants as they could as fast as possible, but without a specific time limit. The comparison and results are presented at the end of this chapter.

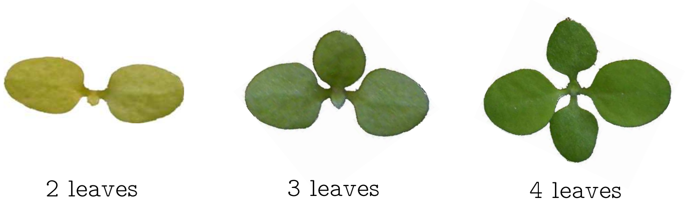


Figure 4.5: Images presented to the participants before they started solving the identification cases.

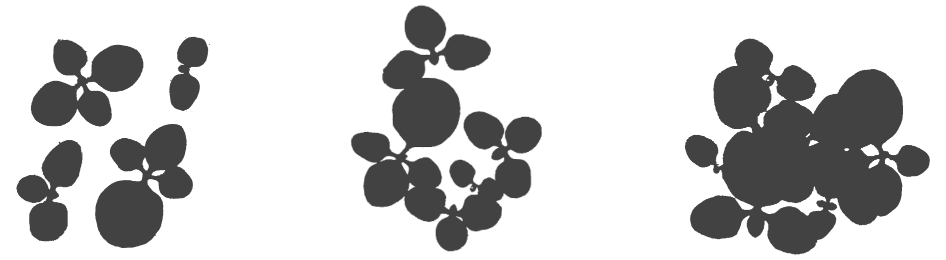

(a) No overlapping (b) 0.1 % overlapping (c) 32 % overlapping

Figure 4.6: Example image showing the type of cases presented to the participants in the comparison experiment.

4.5.3 Image acquisition

The image acquisition was done in a very simple manner: a stand and a commercial camera (Nikon Coolpix P5000) were used to take the pictures of the plantlets, see Figure 4.7a-b. In this way, the series of photos were always taken from the same position, i.e. directly above the plantlet tray (see Figure 4.7c). After germination, the plantlets were photographed every day over a period of 2 weeks, in a phytochamber with constant light conditions (7 kilolux).

The previously mentioned camera uses the RGB color space to represent colors. This color space has 3 different channels: (R) red, (G) green and (B) blue, which

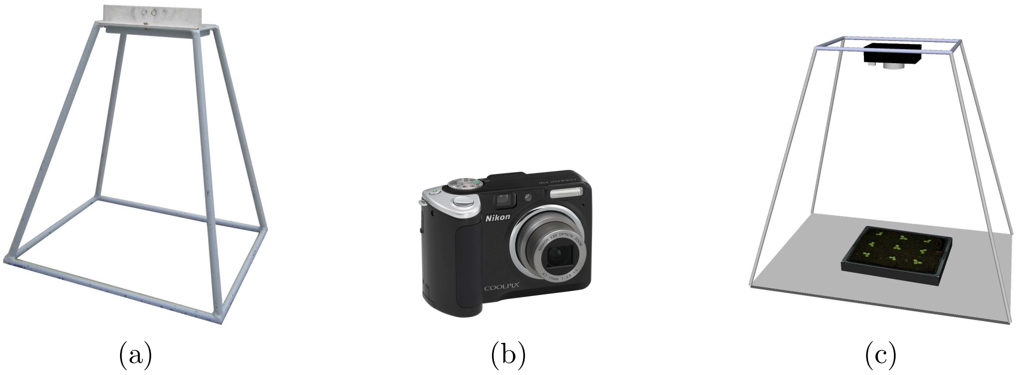

(a) (b) (c)

Figure 4.7: Image acquisition. (a) Stand for the camera. (b) Nikon Coolpix camera. (c) Image showing how the pictures of the plants were taken.

combined in different proportions create new colors. Additionally, the representation used to encode a pixel is a 24-bit RGB, where each channel takes 8-bits. The white balance was set to automatic and no flash was used.

4.5.4 Image segmentation - RGB plant color extraction

Much of the success or failure of every vision recognition system depends on the image segmentation, and so special attention is required. The first step in the analysis is the division of the image into parts or regions. In this particular case, there are just two identifiable regions: plants and soil. Naturally, the part that contains the data to be analyzed (the region of interest, RoI) is the green area pertaining to the plants, see Figure 4.8a-b (input).

It is important to mention that the images used in this work were captured in a chamber that has optimal and constant light conditions. Due to these conditions, there was a high contrast between the color of the trays, soil and plants, thus facilitating the segmentation process.

In order to separate the soil from the plants, an enhancement step and a color filter were used. This method is an adaptation from the excess green index technique proposed by Woebbecke et al. (1995), which performs the enhancement on the green channel of the image. Finally, a threshold filter only allows regions of the green channel to pass through the filter for further binarization. Details of this segmentation procedure is shown in algorithm 4.1. In addition, its effect on two sample plants is depicted in Figure 4.8a-b(output).

Algorithm 4.1 RGB-GREEN EXTRACTION(I, w_f)

```
Input: An image $I$ with $n$ pixels $\mathbf{p}^i = (p_R^i, p_G^i, p_B^i)$, where $(1 \leq i \leq n)$, each
    pixel contains the information for each color channel. A weighting factor $w_f$
    within the interval (0,1].
Compute the minimum and maximum green enhanced values within all pixels
    $\min p_G = \min_{1 \leq i \leq n} [(p_G^i - p_R^i) + (p_G^i - p_B^i)]$
    $\max p_G = \max_{1 \leq i \leq n} [(p_G^i - p_R^i) + (p_G^i - p_B^i)]$
    $\mu = (\max p_G + \min p_G) \cdot w_f$
    $thr = \frac{\mu - \min p_G}{\max p_G - \min p_G}$
for all pixels $\mathbf{p}^i = (p_R^i, p_G^i, p_B^i)$, where $(1 \leq i \leq n)$
    $p_G^* = \frac{p_G^i - \min p_G}{\max p_G - \min p_G}$
    $p_G^i = \begin{cases} 1 & \text{if } p_G^* \geq thr \\ 0 & \text{if } p_G^* < thr \end{cases}$
    $p_R^i = p_B^i = 0$
Output: Image $I$ with a binarized green channel
```

During the experimental phase the value of the weighting factor w_f was set to 0.5.

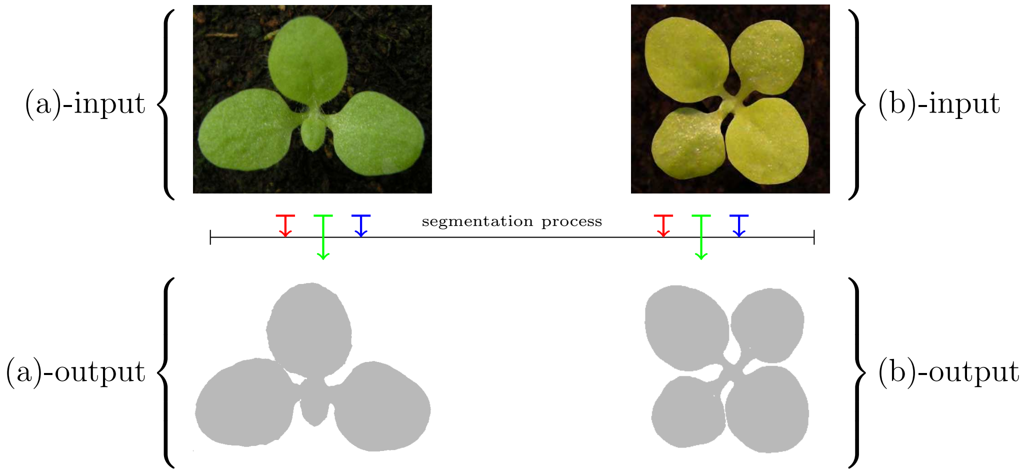


Figure 4.8: Effect of the color segmentation on two sample images containing plantlets. (input) Original images showing single plantlets. (output) Effect of the segmentation process.

Immediately after binarization, a border extraction step is carried out. The algorithm employed to extract the outlines from the binary images was proposed by

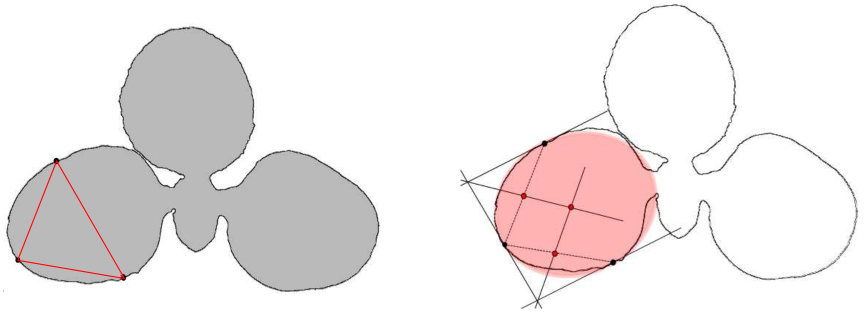
Suzuki & Abe (1985). This algorithm performs a topological analysis to produce a collection of contours and store them in a hierarchical tree data structure, where the exterior contours are at the top of the tree branches and the interior contours (holes) are stored deeper in the branch hierarchy.

Figure 4.9: Example of the border extraction algorithm when applied to a binary region (Suzuki & Abe, 1985).

4.5.5 Ellipse detection using a clustering algorithm

This section concentrates on modeling leaves as ellipses for further grouping into small plantlets. Taking into consideration that many leaves have an elliptical shape allows the use of different methods to match ellipses and to forget for a while about the individual complexity of the leaf shape.

The detection method presented here is and adapted version of the technique proposed by Yuen et al. (1989). This approach iteratively takes 3 random points along the shape border and generates possible ellipses that may match the leaf's area (Figure 4.10a-b). This technique was also used by Mclaughlin (1998) to detect elliptical shapes. This procedure has several advantages, a very important one is that it can identify ellipses that are overlapping one another. Nevertheless, it also generates a large amount of information, including faulty detections.



(a) shape, border and convex lines (b) generation of a possible ellipse

Figure 4.10: Example of how to generate possible ellipses with 3 randomly selected border points. (a) Image showing the segmented shape of Figure 4.3 with the shape border and 3 randomly selected border points . (b) Image with the generated ellipse (red) using the 3 border points and the method proposed by Yuen et al. (1989).

Algorithm 4.2 list the steps that this algorithm follows in order to perform ellipse detection on a given shape contour.

Algorithm 4.2 ELLIPSE DETECTION USING MEAN SHIFT CLUSTERING($\mathbf{s}, wf$)

```
Input: the outline of a shape s that contains $k$ points, and a weight factor $wf$
1: for a $(wf \cdot k)$ number of times attempt to generate ellipses
       a: generate 3 random points
       b: store the generated ellipse in a list
   end for
2: perform mean shift clustering step on the list of ellipses
3: keep the clusters with the highest densities, the ones with the most number
   of elements
4: ensure that the list has valid ellipses
Output: an $n$ number of detected ellipses
```

The weight factor wf used in presented approach was 1, which means that the algorithm repeats the first loop k number of times (the same number of points that are contained in the outline). With respect to the total number of attempts that the method uses to create ellipses, there are two situations that might affect its performance: if the procedure is repeated a small number of times, it is probable that not all the ellipses are going to be detected; if the procedure is repeated too many times, the amount of generated data would be too large to analyze.

After selecting 3 random points, the tangent lines of each selected point has to be computed. This is done using some of its neighboring points (pixels also belonging to the contour) within a selected window size and generating a single straight line that best represents them, see Figure 4.11. After collecting all of the points in the neighboring region, a tangent line can be approximated using linear least squares.

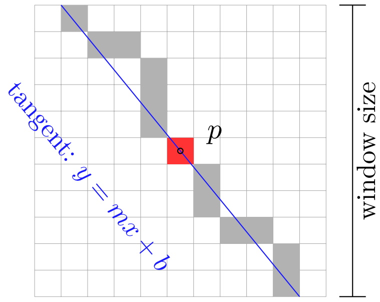
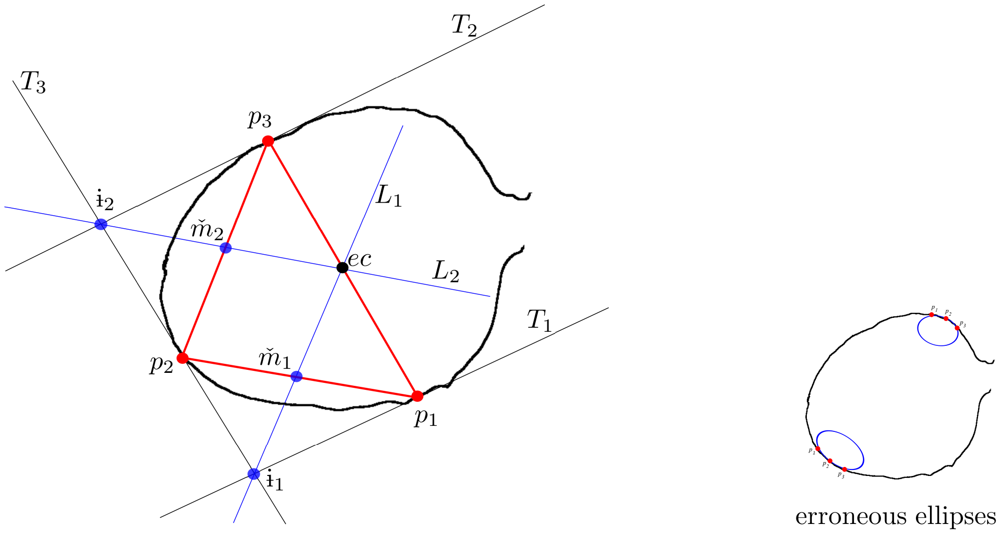


Figure 4.11: Approximating a tangent line to the random point p. The number of neighbor pixels that are taken into account depends on the selected window size.

One of the problems when approximating the tangent line to a selected point is the number of neighboring points that are going to be taken into account. If too few are selected, the resulting line might not be properly aligned, thus leading to erroneous ellipses. If too many are used, the computation of the line takes more time and still does not ensure that the approximated line is correct. The correct number of neighboring points also depends on the size of the shape that is to be analyzed.

With the help of Figure 4.12, the following list of steps demonstrate how to calculate the center of one possible ellipse using 3 random points along the border p_1, p_2, p_3 (Yuen et al., 1989). Note that the 3 random points are the same as the ones shown in Figure 4.10a-b:

1. generate the tangents T_1, T_2, T_3 to the random points p_1, p_2, p_3 as described above.
2. calculate the intersection points $ɨ_1$ and $ɨ_2$ (if they exist) of the tangent lines T_1-T_2 and T_2-T_3 respectively.
3. compute the middle points $\check{m}_1$ between (p_1, p_2) and $\check{m}_2$ between (p_2, p_3).
4. generate the lines L_1, L_2 using the points $(ɨ_1, \check{m}_1)$ and $(ɨ_2, \check{m}_2)$ respectively.
5. the center ec of the ellipse is given by the intersection of the lines L_1-L_2.



erroneous ellipses

Figure 4.12: This image shows in detail how to compute the center of an ellipse using 3 border points. If the chosen random points are too close together, the computed ellipses might be erroneous. An example of this situation is depicted on in the image to the right.

Placing the center of an ellipse with coordinates (x_c, y_c) at the origin $(0, 0)$ allows the simplification of the general ellipse equation (see the following equations: 3.17, 3.24, 3.25 and 3.26) . In order to calculate the rest of the parameters A, B

and C of the simplified equation and to generate the ellipse, the following linear system has to be solved

$$\begin{aligned} Ax_1^2 + Cx_1y_1 + By_1^2 = 1 \\ Ax_2^2 + Cx_2y_2 + By_2^2 = 1 \\ Ax_3^2 + Cx_3y_3 + By_3^2 = 1 \end{aligned} \tag{4.1}$$

where the x_i and y_i values correspond to the coordinates of the 3 selected random points. After the constant terms of the quadratic equation have been calculated, one proceeds to compute the terms of the standard equation of an ellipse (see procedure on page 24). A generalized Bresenham algorithm for conic sections was used to draw the ellipses (Bresenham, 1977; Kennedy, 2009; van Aken, 1984).

Despite simplicity of the method, it generates a large amount of possible ellipses, making it necessary to create groups of ellipses that represent one single leaf. The following image, Figure 4.13, shows some of the possible ellipses after running the method using the shape and border depicted in Figure 4.10a. In this image, one can clearly see patches of several ellipses grouped around specific areas; the groups composed of a larger number of ellipses most likely represent a single leaf.

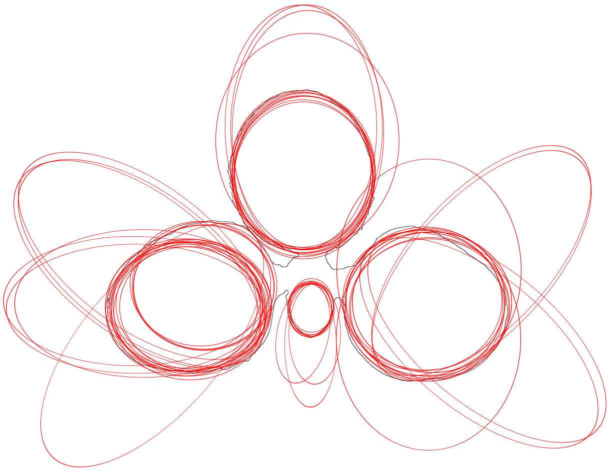


Figure 4.13: Possible ellipses generated with the 3 random points method.

Filtering out false ellipses is necessary because faulty leaves (ellipses) can confuse the plant detection algorithm. This is why the system has to make sure that the selected points form a convex set where a straight line joining any two points of the set must be completely contained within the shape, see Figure 4.10a. The previously mentioned lines are created with the Bresenham algorithm (Angel & Morrison, 1991). Whenever a pixel of these lines is outside of the region of interest, then it is probable that a faulty ellipse will be created, and 3 new random points are generated along the outline of the shape.

In addition to the amount of probable ellipses generated by this method, the number of leaves within a region of interest is unknown and variable. Therefore, it is necessary to integrate an approach that is able to successfully find groups of similar ellipses without any previous knowledge of the data.

A potential problem when clustering ellipses is that circles have equal major and minor axes ($a = b$), so the rotation angle θ plays no role. The problem arises when the algorithm tries to group circles with different rotation angles, which obviously belong to the same group, but the method places them into different ones. In order to solve this problem, a roundness factor was introduced $\xi = (1 - b/a)$. This factor allows θ to be set to a constant value if ξ is close or equal to zero.

The mean shift algorithm, developed by Comaniciu & Meer (2002), is a clustering technique that belongs to the so-called non-parametric methods, which does not require prior knowledge of the data. The algorithm starts by looking for clusters in a set of ellipses and iteratively moves the center of the window towards the local density maximum until convergence. The algorithm uses the concept of a kernel density estimator (kde), also known as Parzen Windows (Duda et al., 2001). This procedure is able to estimate the density of a given data set with n elements that belong to a d-dimensional space $\mathbb{R}^d$ using a kde $K(\mathbf{u})$, where K is a kernel function.

The kernel used in this research is the Gaussian function with zero mean:

$$K(\mathbf{u}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{u^2}{2\sigma^2}} \tag{4.2}$$

The space dimensionality of the problem at hand is five: one dimension for each necessary parameter of the ellipse parametric function $\mathbf{u}_i = \{x_c, y_c, a, b, \theta\}$, where x_c and y_c are the coordinates of the center, a and b are the radii of the major and minor axes, and θ is the rotation angle. In order to look for groups, the clustering algorithm takes each one of the elements in the data set and calculates, several times, a so-called mean shift vector . The mean shift vector gives the direction and new position of the center of the window, which always shifts towards the spots with the highest density. The algorithm will converge when the aforementioned center does not change or the changes are considered negligible . Equation 4.3 sums up the procedure of how to compute the mean shift vector $m_s(\mathbf{u})$, where K is the kernel function, $\bar{\mathbf{u}}$ is the current mean vector, $\mathbf{u}_i$ is a vector in the data set and h is the smoothing parameter (Comaniciu & Meer, 2002, p. 3).

$$m_s(\mathbf{u}) = \frac{\sum\limits_{i=1}^{n} \mathbf{u}_i K\left(\| \ \frac{\bar{\mathbf{u}} - \mathbf{u}_i}{h} \ \|^2\right)}{\sum\limits_{i=1}^{n} K\left(\| \ \frac{\bar{\mathbf{u}} - \mathbf{u}_i}{h} \ \|^2\right)} - \bar{\mathbf{u}} \tag{4.3}$$

In regard to the Gaussian kernel, the value of σ (standard deviation) indicates how the values are spread around the current mean vector. It is assumed that σ is the same for all five ellipse parameters. This assumption enables the use of a control variable ϖ_s (window size), which is computed as follows: $\varpi_s = \varphi\sigma$, where φ is the size of the interval between the z-scores $[\mu \pm \frac{\varphi}{2}\sigma]$ of the normal distribution. Depending on how confident one wants the window range to be,

φ will be greater or smaller but always $\varphi > 0$. Note that a value of $\varphi = 6$ is sufficient because it reaches 99.7 % confidence (assuming the data is normally distributed).

After the clustering process, the list containing all the clusters and their evaluated density (the number of elements) has to be sorted according to their density values for a further trimming procedure. This process will eliminate the clusters which are not above a certain density threshold, and will ensure that only ellipses with a high certainty of being real leaves are analyzed. Since every situation is different and the number of clusters differ every time the procedure is executed, the threshold has to change according to these circumstances. In order to face this problematic, the trimming threshold is set to a value of 10 % of the density of the cluster with maximum density. The groups of ellipses with less members than the threshold are eliminated. To a certain extent, this value might appear too low. However, if the threshold is too high, clusters that represent small leaves will be deleted, and that is not a desirable situation.

The last step in the detection procedure is a discrimination step, where a distinction in favor of the ellipses with the best information takes place. It is considered that one ellipse offers more information than another when its shape covers more area within the region of interest (RoI), and, at the same time, contains minimal area outside the same analyzed region. In order to avoid ellipses with excessive data outside the RoI, a threshold was set to restrict how much area of the ellipse is allowed to be outside. Additionally, two ellipses are considered to be different when they cover different areas, offering new information. If the surface area of one ellipse is already taken, it will only be considered to be existent if it offers new information above a certain threshold. In order to best clarify this situation the following terms are introduced:

a) The *ellipse overlapping area* refers to the ratio between number of pixels that the ellipse and the RoI (pixels pertaining to the image of the real plant) have in common and the total number of pixels in the ellipse.

b) The *new information area* refers to the ratio between the number of pixels in the overlapping area that are not marked as occupied and the total number of pixels in the ellipse. These values are nothing else than the percentage of ellipse area inside the RoI and the area of the ellipse which is not occupied by another ellipse.

$$\text{ellipse overlapping area} = \frac{\text{number of pixels (Ellipse} \cap \text{RoI)}}{\text{number of pixels (Ellipse)}}$$

$$\text{new information area} = \frac{\text{pixels not marked as occupied (Ellipse} \cap \text{RoI)}}{\text{number of pixels (Ellipse)}} \tag{4.4}$$

Figure 4.14 depicts an example of two ellipses that are being analyzed to determine if they provide enough information to be considered as possible leaves.

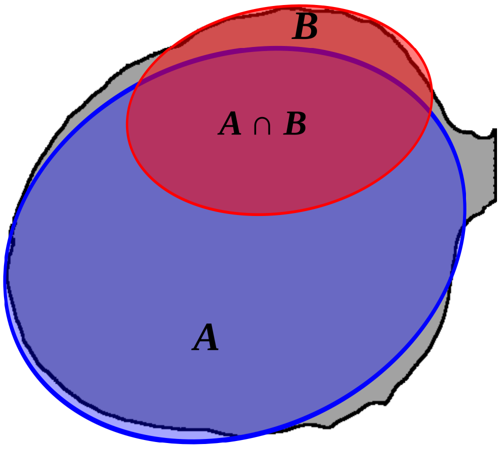


Figure 4.14: Example of overlapping ellipses. This image shows two ellipses A and B, in which ellipse A is already a detected leaf and ellipse B is being analyzed to see if it can be considered a leaf. In this case, the new information offered by ellipse B are the pixels within the RoI (plant pixels in gray) minus those within $A \cap B$.

This procedure starts with the clusters with maximum density, thus ensuring that the best ellipses will occupy its corresponding area at the beginning of the process and make the other ellipses with less density compete for a part of the surface within the region of interest.

4.5.6 Ellipse detection using arcs

The above described algorithm was able to detect the ellipses, however, an extreme amount of time was needed to analyze the large amount of data that it produces. Therefore, a second method based on the detection and combination of arc sections was developed. The general idea behind using arcs is to combine contour sections to approximate the ellipses that best superimpose the area of a given shape. For example, Figure 4.15 depicts arcs labeled as arc-A, arc-B, arc-C and arc-D. It can be seen that the information provided by several arcs can be used to generate ellipses that coincide with the outline of the object. For example, if arc-A and arc-B are combined, a correct approximation is produced; if arc-B and arc-C or arc-A and arc-D are combined, a faulty ellipse is generated; if arc-C and arc-D are combined, a good approximation is also found. Clearly, combining the correct arcs creates the best ellipse for the provided contour.

It is important to mention that the contour of a shape is composed of a sequence of points, where every two neighboring points are connected, generating a path-oriented structure. For example: the set $\mathbf{s} = \{p_1, p_2, \ldots, p_n\}$, composed of n coordinate points $p_i = (x_i, y_i)$, form the following connected path $(p_1, p_2), (p_2, p_3), \ldots, (p_{n-1}, p_n)$.

The developed algorithm provides a simple way to find arc sections of a given con-

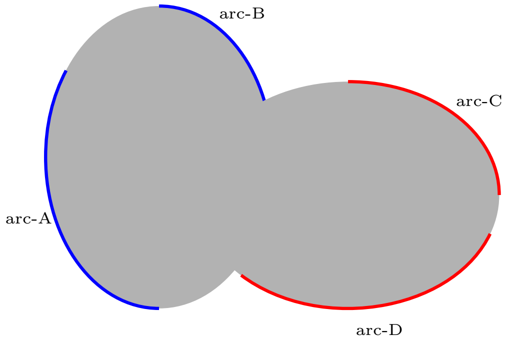


Figure 4.15: Basic arc-combination.

tour for further combination into an ellipse that best overlays the original shape. Finding the arcs is executed by using an operator that takes into consideration intrinsic characteristics of round shapes, including ellipses.

In the polar coordinate system, a reference axis (polar axis) and reference point (pole) are needed in order to calculate the angle between two points. The distance r from the pole to a point is called radial distance, and the angle θ between the polar axis and the line segment connecting the pole and the point is called the angular coordinate, see Appendix A.1.

If the angles of multiple points need to be compared to one another, the polar axis and pole must remain constant. It is well-established that if the pole is placed at the center of a circle, all points on the perimeter will be equidistant from the pole. An ellipse however, has variable radial distance to its center, always varying between the value of the semimajor axis (a) and the semiminor axis (b).

The previous statement is illustrated using the following example: The (r, θ) signature of one circle and one ellipse (centered at the origin) are generated using the approach described in section 3.1.6 (also see Appendix A.1). Furthermore, for the sake of convenience, both objects have the same number n of coordinates in their border path. This means that there are n radial distances r and n angular coordinates θ for each sequence.

Sequence	Radial distance	Angular values
$p_1 = (x_1, y_1)$	r_1	θ_1
$p_2 = (x_2, y_2)$	r_2	θ_2
$\vdots$	$\vdots$	$\vdots$
$p_n = (x_n, y_n)$	r_n	θ_n

Figure 4.16 plots the corresponding (r, θ) signatures of a circle with radius r=2, and an ellipse with parameters a=3 and b=1. It also shows how the radial distance to the center is constant in circles, but bounces back and forth between the values of a and b for ellipses.

By following the previous example, it is possible to present an additional charac-

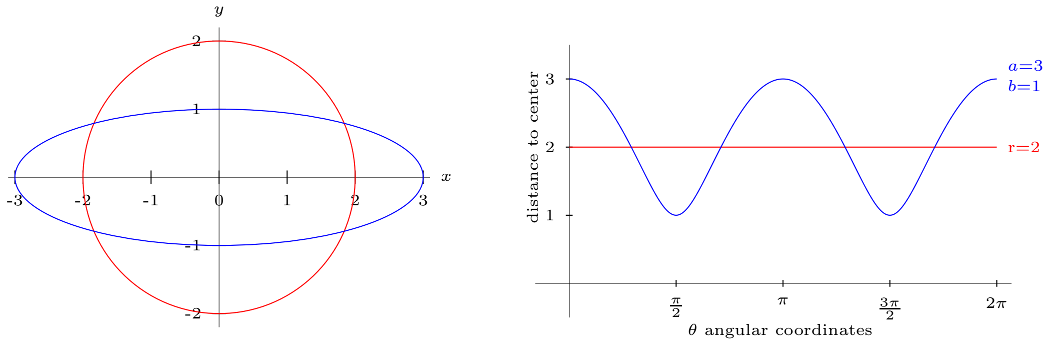


Figure 4.16: Behavior of the (r, θ) signature using an ellipse and a circle.

teristic concerning the behavior of the radial angle. If ellipses are considered to be to some extent one big arc, it is possible to say that all arcs have the same behavior: Their θ_i values are always increasing or decreasing in the form of a *wave*. In this manner, a basic arc extraction can be performed, just by looking at the increasing or decreasing changes in the θ_i values of a given contour. In other words, all the points that belong to one arc can be collected by checking if there is a $(+)$ positive change in the angular value from θ_i to θ_{i+1}. The accumulation of points stops when the change in the angular value is zero or $(-)$ negative. Clearly, the grouping of points can also be done by following $(-)$ negative changes and stopping when the change becomes zero or $(+)$ positive (an ascending or descending value analysis)

However, in the case of circles the (r, θ) signature is a line and the analysis of the ascending or descending values of θ is not possible. In order to carry out such an analysis, a new variable α is incorporated into the calculations to introduce the *wave* behavior for circles. This wave-operator computes an α_i value using the previously calculated (r_i, θ_i) signature values.

$$\alpha_i = r_i \sin(\theta_i) \cos(\theta_i) \quad \text{where } 1 \leq i \leq n \tag{4.5}$$

In Figure 4.17 there are two curves which illustrate the effect of applying the above operator on the already calculated r_i and θ_i sets. It can be seen that both curves oscillate between different intervals $[-\alpha_{min}, \alpha_{max}]$, which depend on their respective semimajor and semiminor axes.

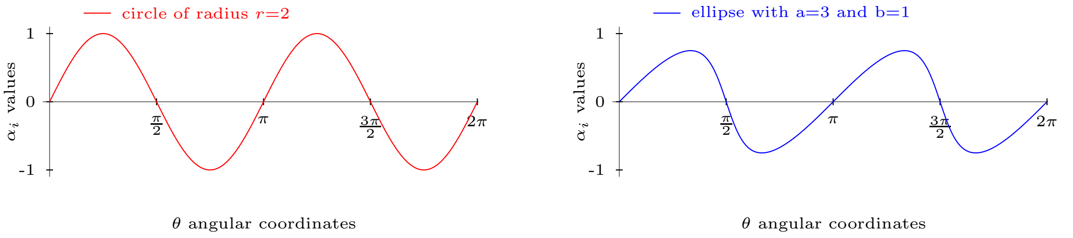


Figure 4.17: Behavior of α values after applying Equation 4.5.

The introduction of the wave operator to compute the α_i values enables the arc extraction by performing an analysis of the ascending or descending behavior of the same α_i values, which in turn have a corresponding (x_i, y_i) coordinate on the contour of the shape being analyzed. Groups of points are created when the α_i values have $(+)$ positive changes (ascending values), and different groups of points are created when the α_i values have $(-)$ negative changes (descending values). The sections where there are no changes in neighboring α_i values (zero-change) are not taken into account.

Figures 4.18 and 4.19 show the results of the arc extraction scheme when applied to the circle and the ellipse of Figure 4.16. One can see that both contours were grouped into 5 arcs. It is worth mentioning that if one starts grouping arcs at one of the lowest points or one of the highest points of the curve, the contour will be subdivided into 4 arcs, where arc 1 and arc 5 are joined together.

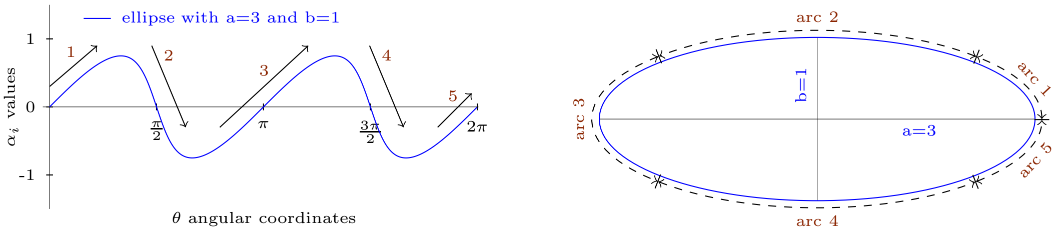


Figure 4.18: Ellipse arc-extraction using ascending descending analysis of the α_i values. Left side image shows the α_i analysis of the outline. Right side image shows the input ellipse and detected arcs.

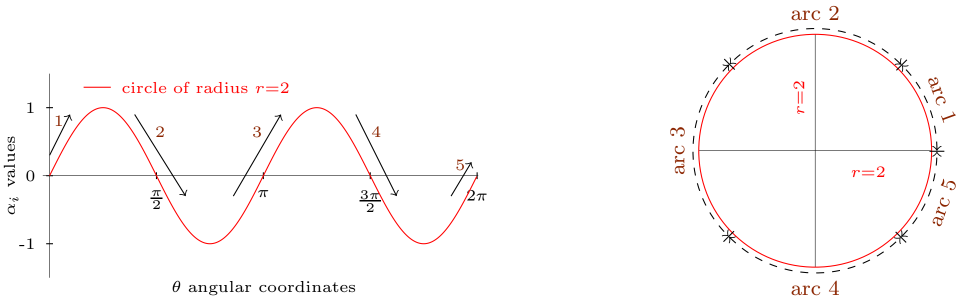


Figure 4.19: Circle arc-extraction using ascending descending analysis of the α_i values. Left side image shows the α_i analysis of the outline. Right side image shows the input circle and detected arcs.

The following algorithm summarizes the arc extraction procedure:

Algorithm 4.3 ARC-EXTRACTION($\mathbf{s}$)

```
Input: a sequence of $n$ points $\mathbf{s} = \{p_1, p_2, \ldots, p_n\}$ forming a continuous path,
    each $p_i = (x_i, y_i)$ is a coordinate, $i \in \{1, \ldots, n\}$.
1: Compute the $r_i$, $\theta_i$, $\alpha_i$ values for each coordinate point.
2: $k \leftarrow 0$. Counter of arc-sequences.
3: $i \leftarrow 1$. Start evaluating at the beginning of the sequence.
4: Evaluate the state of $\alpha_i$ and $\alpha_{i+1}$
       $\Delta \leftarrow \begin{cases} \text{positive change } \textbf{if } (\alpha_i < \alpha_{i+1}) \\ \text{negative change } \textbf{if } (\alpha_i > \alpha_{i+1}) \\ \text{zero change } \textbf{if } (\alpha_i = \alpha_{i+1}) \end{cases}$
   repeat the evaluation until a change in $\Delta$ is found ($i \leftarrow i + 1$) or until $i = n$
5: Create a new arc sequence.
       • Eliminate zero-chage sequences.
       • Store positive or negative change sequences.
       $k \leftarrow k + 1$
       $\mathbf{arc}_k \leftarrow$ {all coordinates found before the change in $\Delta$}
6: Go to step 4 or end if $i = n$.
7: After subdividing the shape contour into $k$ arcs, execute the following:
       • Sort the $k$ arc sequences according to their length (descending order).
       • Apply length threshold
       $\underset{\text{for all } i \in \{1, \ldots, k\}}{\mathbf{arc}_i} \begin{cases} \text{keep } \textbf{if } \text{length} \geq \text{threshold-length} \\ \text{delete } \textbf{if } \text{length} < \text{threshold-length} \end{cases}$
Output: arc sequences sorted according to their length.
```

Algorithm 4.3 first subdivides the shape outline into several arc sections, then it sorts them according to length in order to identify which arcs provide the greatest amount of information about the shape. Finally, the algorithm applies a threshold that removes arc sections that are too small and can be considered negligible. Figure 4.20 depicts the result of applying the arc detection algorithm to a plant contour (see Figures 4.9a-input and 4.9a-output for the original plant shape and plant contour). After executing the arc detection on the plant outline, a threshold was used to eliminate arcs that contained less than 20 coordinate points (note that this threshold was only used to generate this example figure). Furthermore, one can also see that the larger arcs provide more information about the location of individual leaves.

The next step in the ellipse identification process is to combine neighboring arc sections to approximate the ellipse that best represents the area where the arcs are situated. In order to carry out the combination, the *middle point* of each arc section must be found. Given that the arc points are arranged in a linear sequence, the middle point of each arc is the point located in the middle of the

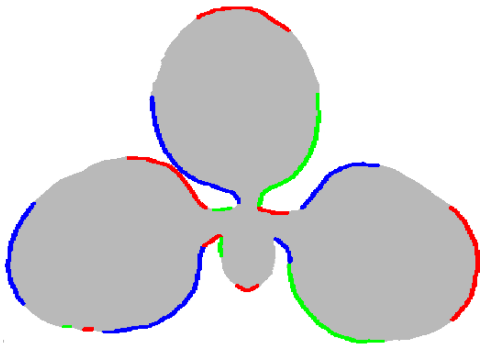


Figure 4.20: Arc extraction on a real plant contour. This example image only shows the arc sections composed of more than 20 pixels.

arc sequence; one should not confuse it with the mean-point. Figure 4.21 shows three arcs from the previous image with their corresponding middle points.

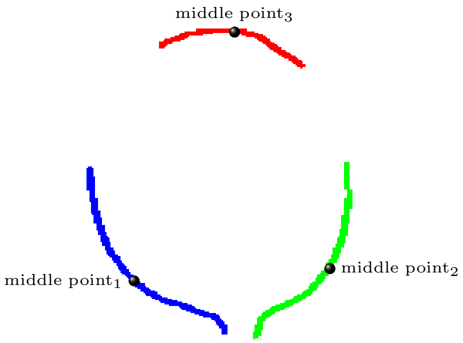


Figure 4.21: Middle points of arc sequences.

Having located all middle points, one can start with the combination of arc sections to find the ellipses that best superimpose the shape area. It was previously mentioned that the **arc** sections are sorted according to their length; this gives the advantage that the longest arc sections are selected first for the ellipse fitting. The reason to do this is to use the arcs with the most information about the outline before the small arcs that contain less information are taken into consideration. However, before combining arcs, one has to make certain that they are real neighbors.

For instance, assume that there are a total of n **arc** sequences after applying the sorting and the length threshold steps. In order to discover neighboring arcs around a selected **arc**, a series of steps have to be followed:

Algorithm 4.4 Discover true neighboring arcs of $\mathbf{arc}_{top}$(n arcs)

Input: A set of n arc sequences sorted according to their length

```
From the $n$ arc sequences, select the arc that contains the greatest number of points $\mathbf{arc}_{top}$.
By using the Euclidean distance between arc middle points, calculate the $m$ nearest arc neighbors to the selected $\mathbf{arc}_{top}$.
Within these $m$ nearest arc neighbors find the true arc neighbors of $\mathbf{arc}_{top}$. Two arc sections are considered to be neighbors when the line segment connecting their middle points are fully contained in the shape area (region of interest). Whenever a couple of pixels of the line segment are outside of the region of interest, the arc sections are not recognized as true neighbors [1].
```

Output: A list with all the *true arc neighbors* of $\mathbf{arc}_{top}$.

The following image, Figure 4.22, shows an example of how to identify the real arc neighbors of a given arc section. In this image, it is possible to see that from the arc section identified as $\mathbf{arc}_{top}$, lines are drawn to the other neighbor arcs. The lines that are convex to the shape identify the real neighbors, while the lines not convex to the shape (dashed lines) identify spurious neighbors.

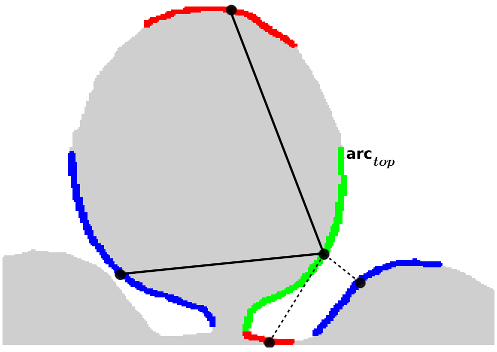


Figure 4.22: Identifiying real arc neighbors. Drawing lines on the RoI: Solid lines identify real neighbors, whereas dashed lines identify spurious neighbors.

After finding the real arc neighbors of a given arc section, one can start trying to combine them in order to find the ellipse that best fits the area where the arcs are found. For example, the arc depicted in Figure 4.22 ($\mathbf{arc}_{top}$) is combined with two other arc sections to find the ellipse that best fit that area.

Algorithm 4.5 clarifies how the combination of arcs is executed. Assume that a given $\mathbf{arc}_{top}$ was selected to be analyzed. Furthermore, its k real arc neighbor sequences were found and stored in a list. It also computes the best ellipse by combining a set of neighboring arcs. To certain extent, one can think of this

---

[1] The lines that connect two middle points are drawn using the Bresenham algorithm (Angel & Morrison, 1991).

algorithm as an arc growing approach, because with each new combination, the method tries to use other border sections to best fit an area. Moreover, this algorithm uses the overlapping area between the generated ellipse and the shape to test if the combination of two set of points approximate a suitable ellipse (of course, suitable according to the given region of interest), see equation 4.4 and Figure 4.14.

Algorithm 4.5 ARC COMBINATION($\mathbf{arc}_{top}$, k neighbor sequences)

```
Input: an $\mathbf{arc}_{top}$ sequence and its $k$ real arc neighbors ($\mathcal{N}$)
for each neighbor sequence $\mathcal{N}_i$, where $i \in \{1, \dots, k\}$.
    • Fit an ellipse $\mathbf{E}$ using the data points of $\mathbf{arc}_{top}$ and $\mathcal{N}_i$, see section
      3.2.1.3 for more details on ellipse fitting.
    • Draw $\mathbf{E}$ on top of the RoI and apply different thresholds (see equation
      4.4 and Figure 4.14)
      $\mathbf{E}\begin{cases} \text{keep } \textbf{if} \text{ overlapping area} \geq \text{threshold-area} \\ \text{delete } \textbf{if} \text{ overlapping area} < \text{threshold-area} \end{cases}$
      $\mathbf{E}\begin{cases} \text{keep } \textbf{if} \text{ new information area} \geq \text{threshold-information} \\ \text{delete } \textbf{if} \text{ new information area} < \text{threshold-information} \end{cases}$
    • if ellipse $\mathbf{E}$ passes all the thresholds, then $\mathbf{arc}_{top}$ inherits all the points
      of $\mathcal{N}_i$.
    • if ellipse $\mathbf{E}$ does not pass all the thresholds, then $\mathcal{N}_i$ is not considered
      a viable arc combination and goes back to the list of all arc sections.
      $\mathbf{arc}_{top}$ stays the same and awaits another combination.
If $\mathbf{arc}_{top}$ did not combine with any other arc(s), then fit an ellipse $\mathbf{E}$ using
  only its points and test the above thresholds.
Stop execution and return null If ellipse $\mathbf{E}$ did not pass all the requirements.
Mark the area that the ellipse and the shape have in common.
    $\text{occupied area} = \mathbf{E} \cap \text{RoI}$
Output: an ellipse $\mathbf{E}$ and the combined $\mathbf{arc}_{top}$.
```

During this research, the *overlapping area* threshold was set to 97 % of the total number of pixels in the ellipse. Likewise, the *new information area* threshold was set to a 20 %, this means that an ellipse has to have no less than 20% of its total number of pixels unoccupied by another ellipse.

Up to this point, it has been explained how to generate the best overlapping ellipse using one arc section and its neighbors. However, if one recalls, the output of the arc extraction algorithm (Algorithm 4.3) was a list of arcs that was sorted according to their length.

To complete the process of computing the best ellipses that fit the original shape, it is necessary to repeat the previously mentioned steps with all the arc sections found by the algorithm. The following set of instructions summarize how the iteration is executed:

Algorithm 4.6 DISCOVER ALL ELLIPSES WITHIN A ROI (n arc sequences)

```
Input: $n$ arc sequences

1: Assume that there are a total of $n$ arc sequences already sorted according to
   their length. Those sequences with the greatest length are at the beginning of
   the list.
2: for all $\mathbf{arc}_i$ sequences in the list, where $i \in \{1, \dots, n\}$.
     • Find the true arc neighbors $\mathcal{N}$ of $\mathbf{arc}_i$ (see Algorithm 4.5).
     • Find the ellipse $\mathbf{E}$ that $\mathbf{arc}_i$ and its neighbors fit. If an ellipse was found,
       store it in a list of viable ellipses (which represent viable leaves).
     • Mark all the arc sections that were combined in the previous step. If an
       arc was already combined, it cannot be used again.

Output: A list that contains $k$ ellipses $\mathbf{E}$
```

The list that contains all the computed ellipses will be used in the leaf clustering step, which combines different ellipses to find plant shapes. Also note that the output of this ellipse detection method are the ellipses that best superimpose the area of interest. If one recalls the first ellipse detection method (see section 4.5.5) it produces an output that has to be further analyzed in order to discriminate the faulty detected ellipses, see Figure 4.13.

4.5.7 Landmarking procedure

In order to generate the deformable template, it is essential to extract information about the general shape of the selected model plants. In the case of an ASM, it is necessary to find points along the outline, called landmarks, that will denote the shape of the plant. The landmarking procedure starts by running the ellipse detection procedure to find every leaf on each model plant (see section 4.5.6) . If the detection was successful, each plant can be considered as a set of ellipses, as the ones shown in Figure 4.23.

The proposed landmarking procedure computes 5 points for each detected ellipse and 1 extra point for the center of the plant, for a total of $5n$ landmarks, depending on the n number of leaves of each plant model. It is worth noting that the plant center is only used as the anchor point to shift the plant to the origin, it is not used as a landmark point in the models. Additionally, it is assumed that all plantlets have either 2, 3 or 4 leaves. Thus, $n \in \{2, 3, 4\}$.

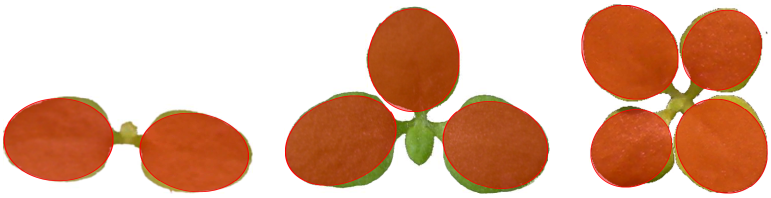


Figure 4.23: Representing the leaves of a plantlet with ellipses.

Suppose that a plantlet with n leaves (ellipses) is to be landmarked. After running the ellipse detection procedure, which successfully detected all n ellipses of the given plant, one can start placing the landmark points. Algorithm 4.7 lists the steps that are followed to place the landmark points on each ellipse.

For the purpose of visual clarification, each step of the landmarking process will be shown on an example plantlet.

• The first landmarks to be extracted are the ellipse centers (ec) of each detected leaf. In this case, a total of $n = 2$ centers.

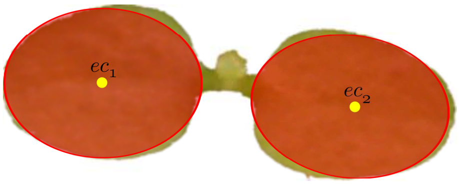


Figure 4.24: Extracting landmarks step 1: finding the ellipse center.

• In order to compute the rest of the landmarks, an iterative search procedure is executed to locate the plant center, which will group the leaves into an individual plantlet:

1. The procedure starts by placing some k number of points (i.e. $k = 360$) around the perimeter of every ellipse. Depending on how accurate or fast one wants the calculations to be, the value of k can be increased or decreased arbitrarily.
2. Then, the first tentative plant center pc is computed by calculating the mean point of the n ellipse centers.
3. For each of the n ellipses, find the closest perimeter point cp to the center pc. Then update the set that stores the n closest perimeter points $\{cp_1, \ldots, cp_n\}$.
4. Calculate the new plant center pc_{new} by computing the mean point of the n closest perimeter points.
5. Compare the plant centers pc_{new} and pc. Stop the procedure if there are no changes or if the difference is so small that it is considered to be negligible. Otherwise, replace pc with pc_{new} and return to step 3.

Thus far, the landmarking procedure has extracted a total of n ellipse centers (ec), n closest perimeter points (cp) and 1 plant center (pc). This step is an important part of the landmarking process, during which the position of each petiole is determined, i.e. the points where each leaf attaches to the plant stem. For instance, the plantlet that is in the process of being landmarked looks like the image depicted in Figure 4.25. The petiole of the plant should be located where the line segments connect points cp with the point pc.

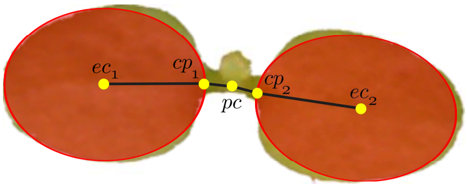


Figure 4.25: Extracting landmarks step 2.

- The procedure continues with the extraction of landmarks, but now focuses on individual ellipses. The next step is to find, within the corresponding perimeter points, the furthest point fp to the plant center pc.

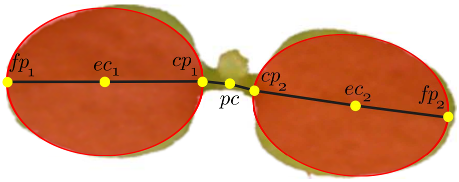


Figure 4.26: Extracting landmarks step 3.

- Finally, the last landmarks to be found are the ones that represent the breadth of each leaf. In order to find these, the procedure generates the line that passes through the points fp and cp of each ellipse. Every line divides each ellipse into two parts that contain one landmark each: wa and wb. These landmark points are the ones that maximize the orthogonal distance to the line dividing each ellipse.

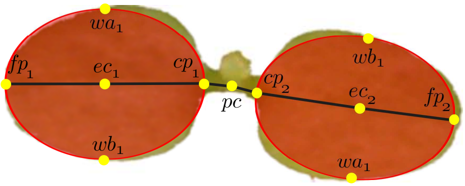


Figure 4.27: Extracting landmarks step 4.

The presented procedure is an automated way of landmarking each plantlet. With a small number of points, this method is able to represent the whole shape of the plant. Figure 4.28 shows a landmarked version of the plants presented in Figure

4.23. One is able to see the skeleton of each plant, the breadth of each leaf, the petioles connecting all leaves, and the center of the plant.

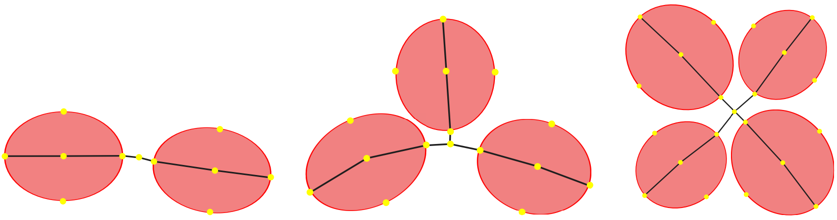


Figure 4.28: Examples of fully landmarked plants.

Algorithm 4.7 PLANT LANDMARKING PROCEDURE (n ellipses)

Input: n ellipses, where $n \in \{2, 3, 4\}$.

```
1: Store the centers $ec_i$ of the $n$ ellipses
2: Using the ellipses centers iteratively compute the center of the plant $pc$
3: Calculate the closest points $cp_i$ from each ellipse to the plant center
4: Compute the furthest points $fp_i$ from each ellipse to the plant center
5: For each ellipse compute the points $wa$ and $wb$, which represent the width of
   each leaf.
```

Output: $5n$ landmark points

4.5.8 Deformable plant templates

This section will introduce the deformable models created to recognize tobacco plants with 2, 3 and 4 leaves. It was previously mentioned that the model plants were seeded with sufficient distance from each other, so that there was no overlapping between them.

The model plants were individually landmarked (see section 4.5.7) with the purpose of creating a training set that allowed the generation of the ASM. Table 4.2 indicates the number of sample images that were used to create each shape model.

Plant model	Number of training images
2 leaves	70
3 leaves	70
4 leaves	70

Table 4.2: Individual plant images used to train each deformable model.

The ASMs presented here were created using the procedure introduced in section 3.3, and the landmarking for each training plant was carried out automatically.

The only information provided for the landmarking procedure was the real number of leaves that the plant had.

Finally, after collecting all the landmarks, groups of 2 leaves, 3 leaves and 4 leaves were generated in order to statistically analyze them, and to formulate the deformable models. Each generated ASM is composed of a set of *eigenvectors* $\mathbf{\Phi}$, *eigenvalues* $\boldsymbol{\lambda}$ and a *mean shape* $\boldsymbol{\mu}$.

For the purpose of clarification, the information from each model was gathered and plotted. Figures 4.29, 4.30, 4.31 show the *deformable templates* that resulted from applying the ASM learning procedure to the extracted landmarks. One should note that these images are aligned and normalized as described in sections 3.3.3 and 3.3.4.

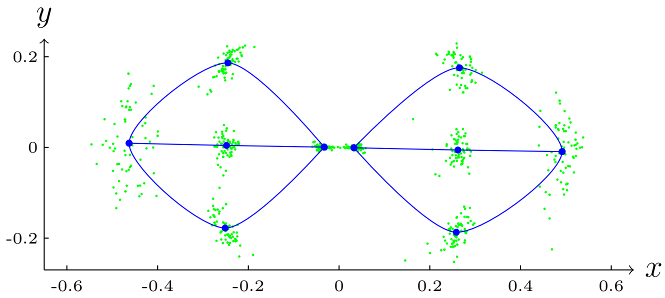


Figure 4.29: Deformable template for tobacco plants with 2 leaves. Each green point represents an individual landmark from the training set, the blue points and lines represent the deformable template or mean shape $\boldsymbol{\mu}$.

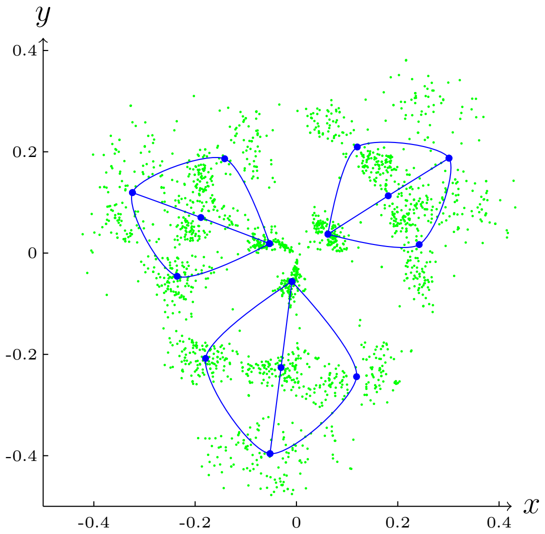


Figure 4.30: Deformable template for tobacco plants with 3 leaves.

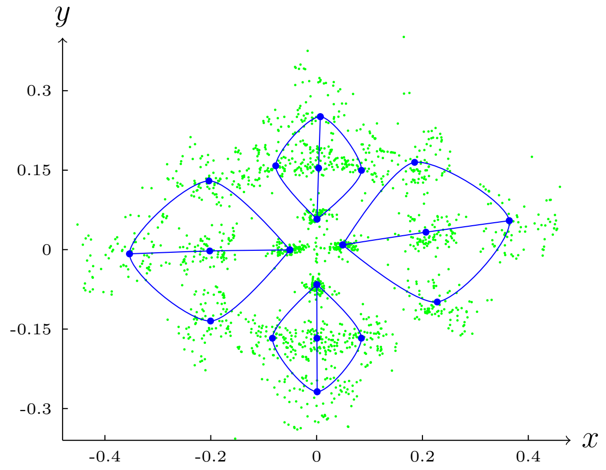


Figure 4.31: Deformable template for tobacco plants with 4 leaves.

If one recalls from section 3.3.4 (see Algorithm 3.4), it is possible to arrange the eigenvalues λ as a percentage fraction of the total variance that they represent. This provides the possibility of performing dimensionality reduction on the models by just using the most important eigenvalues and their corresponding eigenvectors.

That being said, it is interesting to plot the eigenvalues of each shape model to analyze their behavior, see Figures 4.32, 4.33, 4.34. The following plots present each one of the eigenvalues after the dimensionality reduction, shown as a percentage fraction of the variance they represent.

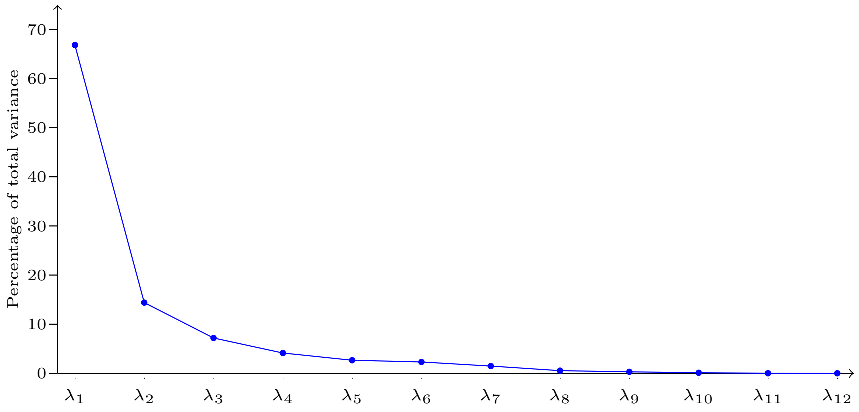


Figure 4.32: Plot of eigenvalues from the 2 leaves model.

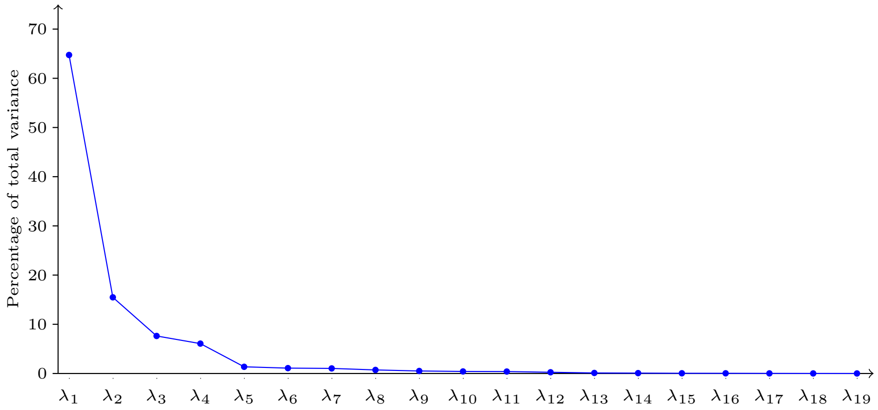


Figure 4.33: Plot of eigenvalues from the 3 leaves model.

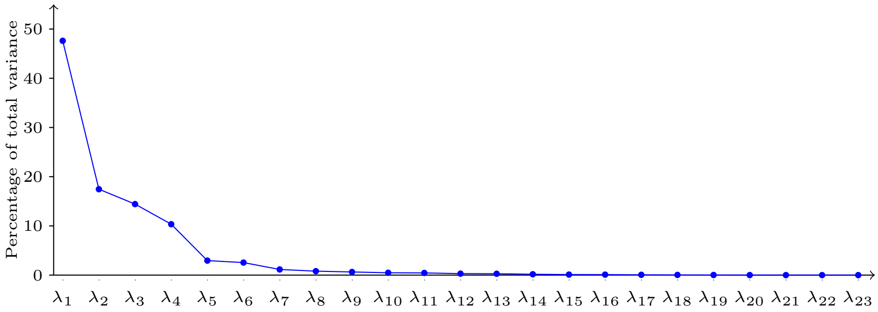


Figure 4.34: Plot of eigenvalues from the 4 leaves model.

After observing the plots, one can realize that further dimensionality reduction in the models is possible, since almost all of the information was contained within the first 10 eigenvalues. However, during this research for the sake of accuracy, only a reduction to 99.9% of the total variance was carried out (i.e. all of eigenvalues plotted in the above graphs).

4.5.9 Neighboring leaves combination

Having described the landmarking approach and the plant models that are going to be used in the identification process, it is necessary to formalize the use of these deformable templates in a real situation.

Imagine that the detection of the plantlets in Figure 4.35a is desired. Moreover, assume that the ellipse detection algorithm accurately localized all leaves in the image, as depicted in Figure 4.35b.

In reality, the collected information, after running the leaves detection algorithm, is a list of ellipses. Visually, this list represents a set of neighboring ellipses, such as the ones depicted in Figure 4.35c. If the plant identification is performed using only the plot of ellipses, one will realize that the task is very cumbersome.

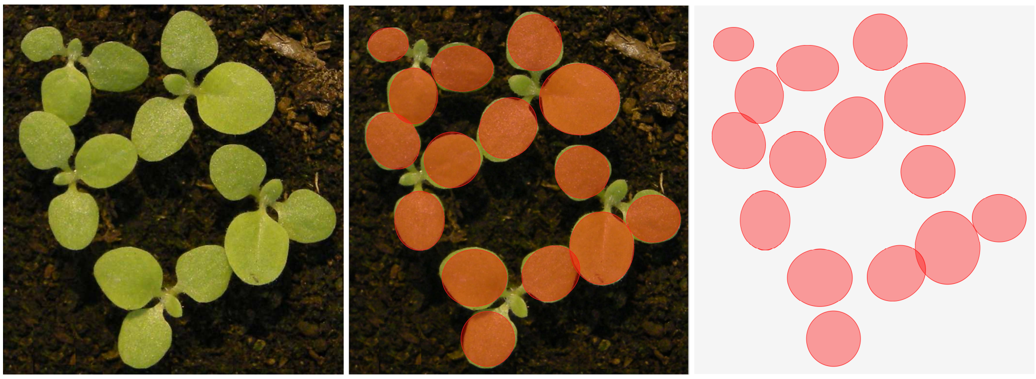


Figure 4.35: Example of real plant detection. (a) Image showing tobacco plantlets. (b) Image after detecting all leaves with the arc ellipse detection approach. (c) Plot of neighboring ellipses.

However, this is the only information that the ASM uses to identify each plant. To some extent, the ASM performs the job of ellipse grouping, and always tries to create the best model of a small-sized plant. Additionally, there might be falsely detected ellipses (leaves). Despite this fact, the shape model has to be able to find the correct group of ellipses that represent the actual real-life plant.

The steps of the method used to match small plants by combining neighboring ellipses are listed in Algorithm 4.8. After presenting the general idea of the procedure, each step is described in detail.

Algorithm 4.8 ELLIPSE COMBINATION PROCEDURE (**L**)

```
Input: A list L containing the data of each detected ellipse is required. Suppose that this list has a total of $l$ ellipses $\{\mathbf{E}_1, \mathbf{E}_2, \dots, \mathbf{E}_l\}$, where $\mathbf{E} = \{x_c, y_c, a, b, \theta\}$.

1: Extract an ellipse $\mathbf{E}_i$ from the list, where $i \in \{1, \dots, l\}$.
2: Find the $m$ closest neighboring ellipses to ellipse $\mathbf{E}_i$ and form a set of $n$ members, where $n = m + 1$.
3: In order to find the best plant, test all possible combination of the previously generated set using the deformable template (ASM). See Algorithm 4.9.
4: If a plant is found store its leaves (ellipses) and remove them from L. In other case, remove ellipse $\mathbf{E}_i$ from the list.
5: Go to step 1 until there are no more ellipses $\mathbf{E}_i$ in the list.

Output: a total of $k$ plants found by the ASMs.
```

• The first steps in the algorithm are to extract an ellipse $\mathbf{E}_i$ from the list, and then to find the m (i.e $m = 10$) closest ellipses within the $l-1$ elements of the list **L**. The idea of identifying the nearest ellipses is to reduce the data set that the deformable model has to test when looking for a combination of leaves that would produce a plant. Figure 4.36 illustrates the case of an ellipse when looking for the m ellipses that minimize the distance d between their centers.

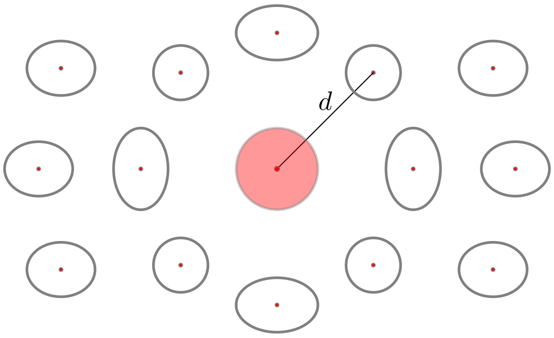


Figure 4.36: Locating neighboring ellipses. d denotes the Euclidean distance between two center points.

- The following steps are to use ellipse $\mathbf{E}_i$ and the other m neighboring ellipses to create a set of n elements, where $n = m + 1$. This set will be used to generate several different combinations in search of the one combination that best approximates a plant. For the moment, it is assumed that the best combination is going to be selected. Please refer to section 4.5.10 to find out more about this selection scheme.

The combinations of the n ellipses taking k distinct elements at a time without any repetitions is equal to the binomial coefficient

$$ {}_nC_k = \binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{4.6}$$

where $0 \leq k \leq n$. Also, the value of k is determined by the *number of leaves* that the deformable template contains. For instance, use $k = 2$ when searching for plants with 2 leaves, $k = 3$ when trying to match plants with 3 leaves, etc.

Each k-subset will be landmarked and tested with the ASM to check how well its ellipses deform into a plantlet. Certainly, as is to be expected, the greater the size of the combination set, the greater the number of k-subsets that the method has to check, thus increasing the computational time.

After checking the viability of all k-subsets (possible plant or not a plant), the best fit will be selected as a recognized plant in the image and its elements will be extracted from the list $\mathbf{L}$ that contains all of the ellipses. Naturally, the policy of always selecting the best k-subset can rule out the ellipse $\mathbf{E}_i$. For this reason, whenever the extracted ellipse $\mathbf{E}_i$ is part of a combination subset, one has to check if it forms a possible plant. In the case that $\mathbf{E}_i$ is part of a possible plant, but is not in the final selection, insert $\mathbf{E}_i$ back in the list $\mathbf{L}$. If, after checking all possible combinations, $\mathbf{E}_i$ is not part of any possible plant, one can delete it and treat it like a faulty ellipse detection.

- The above mentioned steps of the combination procedure have to be repeated until there are no ellipses to combine in the list $\mathbf{L}$.

In order to improve the plant identification process, a restriction was made during the landmarking step of every k-subset. This restriction stipulates that at least one pixel of the 3×3 square of the calculated plant center pc has to be within the region of interest. If the center is not within the region of interest, the combination subset will be treated as a none viable plant shape. The following image shows how the plant center is placed in this square in order to compare it to the interest region.

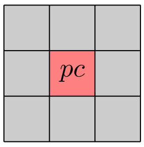


Figure 4.37: Placing calculated plant center in a 3×3 square

In the case that several models are being used to match plants at the same time (i.e. the 2, 3 and 4 leaves models), each model has to be tested using all the nearest neighbors, as described above. However, a small modification has to be made.

Imagine that step 2 of the neighboring leaves combination is about to be executed, where a group with n ellipses will be combined to find the best plant (ellipse $\mathbf{E}_i$ and its m ellipse neighbors).

1. Combine $\binom{n}{2}$, all possible 2-subsets in the search of the best plant with 2 leaves.
2. Combine $\binom{n}{3}$, all possible 3-subsets in the search of the best plant with 3 leaves.
3. Combine $\binom{n}{4}$, all possible 4-subsets in the search of the best plant with 4 leaves.
4. Select the best plant (within the plants with 2,3, or 4 leaves) and remove the ellipses that belong to this plant from the list of ellipses $\mathbf{L}$, see section 4.5.10 to learn about the plant selection scheme.
5. If no viable plant was found, eliminate $\mathbf{E}_i$ from the list of ellipses $\mathbf{L}$.
6. If $\mathbf{E}_i$ is part of a viable plant, but it is not part of the final k-subset, put it back into the list of ellipses $\mathbf{L}$.

The only thing that changed from the first description is that, in this case, the combination subsets with different k values have to be analyzed and finally compete for the best deformed plant shape.

4.5.10 Energy heuristic to match real plants

This section will clarify the heuristic approach which decides if a certain combination of ellipses is a plant, and also how to decide that one combination is better than another. Algorithm 4.9 describes the procedure to test if certain combination of ellipses (k-subset) is a real plant.

In order to test if the shape $\mathbf{s}$, extracted from a k-subset of ellipses, passes the deformable conditions of the ASM, one has to compute the $\boldsymbol{\beta}$ vector (see section 3.3.5 and algorithm 3.4) using the current shape $\mathbf{s}$, and the corresponding model for 2, 3 or 4 leaves $(\boldsymbol{\lambda}, \boldsymbol{\Phi}, \boldsymbol{\mu})$.

$$\boldsymbol{\beta} = \boldsymbol{\Phi}^{\mathsf{T}} (\mathbf{s} - \boldsymbol{\mu}) \tag{4.7}$$

At this point, the deformation values β_i are checked. They have to be within their corresponding intervals $[-3\sqrt{\lambda_i}, 3\sqrt{\lambda_i}]$. One has to remember that each β_i explains the deformation that its corresponding ϕ_i will be subjected to. If the β_i values are out of range of the intervals, then the shape $\mathbf{s}$ is taken as a non-viable plant and is discarded.

Algorithm 4.9 MATCH REAL PLANTS WITH ASM (k-subset of ellipses)

Input: A k-subset of ellipses, where $k \in \{2, 3, 4\}$

1. Test if the center is part of the RoI. If the center is not within the RoI, this is not a plant and another k-subset must be tested.
2. If the center is within the RoI.

   a: Extract the landmarks from the set of ellipses to form the shape $\mathbf{s}$.
   b: Shift the landmarks, using the center of the plant pc as the origin.
   c: Normalize the shape, $\| \mathbf{s} \| = 1$.
   d: Rotate $\mathbf{s}$ towards the mean shape $\boldsymbol{\mu}$ of the ASM, see section 3.3.3.
   e: Test if the mean shape $\boldsymbol{\mu}$ can be deformed into the shape $\mathbf{s}$. If the shape meets the conditions to be a plant, this set of ellipses is consider a real plant. In other case, this combination is not consider a feasible plant.

Output: A positive or a negative answer after testing if a k-subset can be deformed into a real plant.

In the case that the shape $\mathbf{s}$ passed all the restrictions, compute its *deformation energy*, which is the sum of all the deformation values:

$$de = \sum_{i=1}^{n} |\beta_i| \tag{4.8}$$

Store all the information corresponding to the shape $\mathbf{s}$: ellipses that generate the shape, the landmarks, and its *deformation energy*.

After testing all the possible shapes with the same number of leaves (k-subsets with the same k), the ellipse combination that is considered the best deformation above all other m plausible deformations is the one that has the minimum deformation energy de.

$$\text{best plant} = \min_{de_i}\{de_1, de_2, \ldots, de_m\} \tag{4.9}$$

In the case when the ASMs for 2, 3 and 4 leaves are being used simultaneously, the best plant that deforms each model has to be found. In order to decide which plant to choose, a weighting factor is used. This factor is computed as follows:

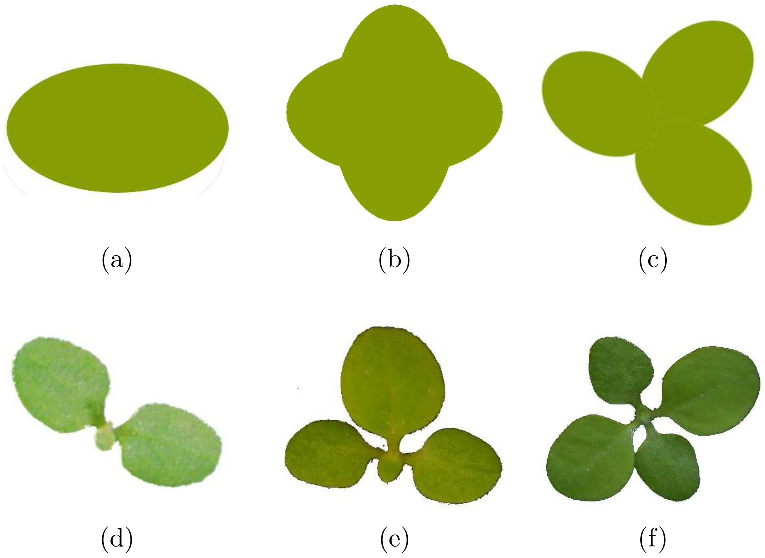
$$de_{new} = \frac{1}{k} e^{-de} \tag{4.10}$$

where k is the number of leaves that each best possible plant has. Finally, the plant to select would be the one with the minimum de_{new}. Clearly, this weighting scheme favors the plants with more leaves; this was done in order to give preference to the more complex shapes that require more deformation energy.

4.6 Results

4.6.1 Comparison of ellipse detection methods

During this chapter two methods to detect ellipses were presented: one procedure uses three random points to generate ellipses (3Point method - see section 4.5.5); the other method extracts the arcs of a contour, and then uses them to fit ellipses (Arc method - see section 4.5.6).


(a) (b) (c)

(d) (e) (f)

Figure 4.38: Images used to compare the ellipse detection methods.

A comparison between these two methods was carried out as follows: Six images containing ellipses and real plants were selected in order to input them into the detection programs, see Figures 4.38(a to f). Then, the detected number of ellipses and the elapsed time were collected. It is worth noting that these experiments were executed using the same computer: Intel(R) Pentium(R) D CPU 2.66GHz with 1 gigabyte of RAM.

Furthermore, both of these methods have parameters that can affect their performance, for example: minimum arc length, number of random 3-point trials to test, area of each ellipse that must be included in RoI, etc. In order to compare both methods, these parameters were fixed and did not change while performing the comparison.

In the case of the 3Point method: The parameter that defines how many sets of 3 random border points to analyze was set to 50 % of the total length of the shape outline. The tangent approximation window size was set to 5 pixels, and the area of each ellipse that must be included in RoI was set to 97 %.

In the case of the Arc method: The minimum arc length was set to 8 pixels, and the area of each ellipse that must be included in RoI was set to 97 %.

Image	Detected ellipses		Execution time (s)	
	3Points	Arcs	3Points	Arcs
a	6	1	22.44	0.17
b	9	2	12.89	0.21
c	8	3	10.17	0.32
d	2	3	1.13	0.06
e	7	4	18.91	0.22
f	6	7	3.79	0.31

Table 4.3: Results of comparing the two ellipse detection methods. 3Points: method that generates ellipses using 3 random points. Arcs: method that generates ellipses using arcs.

In the above table, it is possible to see that the method that uses arcs needs less computational time to detect ellipses than the 3Points method. This factor was decisive in the selection of the Arc detection method over the other technique. Moreover, ellipse detection using arcs is much more accurate when detecting overlapped and non-overlapped ellipses. For instance, if one looks at the image of the single ellipse (Figure 4.38a), the 3Points method computes 6 ellipses where there is clearly just 1, which the Arc method correctly found. The second image (Figure 4.38b) depicts 2 ellipses that are overlapping each other, the Arc method was able to find these 2 ellipses, whereas the 3Points method found 9 ellipses.

The third image (Figure 4.38c) shows 3 ellipses overlapping one another: again, the Arc detection method was able to find the correct number and position of the ellipses, while the 3Point method detected 8 ellipses. The aforementioned examples, whose ellipse detections are depicted in Figure 4.39, represent perfectly elliptical shapes.

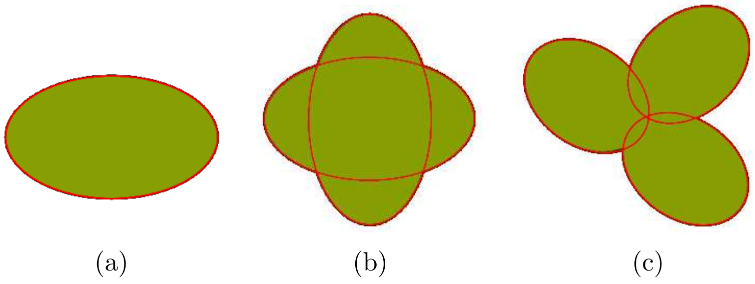


Figure 4.39: Output of the Arc ellipse detection using ellipses as input. Images a, b and c show the detected ellipses for each case.

In the case of plant leaf detection (fitting ellipses over leaves), one has to consider that leaves do not have perfectly elliptical shapes and more than one ellipse can fit within the leaf area.

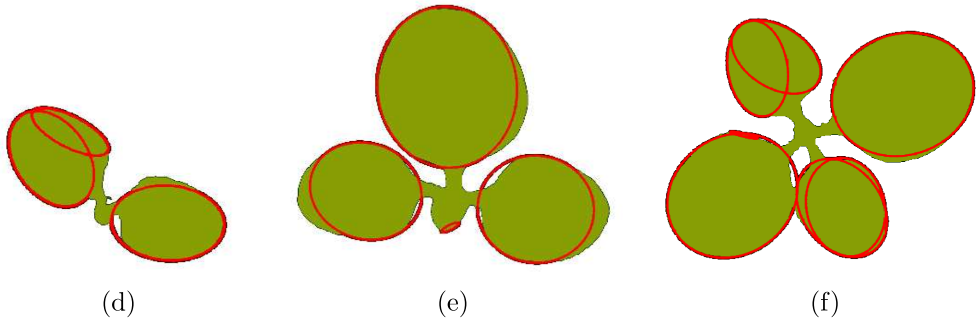


Figure 4.40: Output of the Arc ellipse detection using plants as input. Images d, e and f show individual plantlets and their corresponding leaf detection.

When comparing the two ellipse fitting techniques using single plants as input, the Arc detection method again performed much better than the 3Point technique. For example, the Arc detection method found 3 ellipses for the 2-leaf plantlet shown in Figure 4.38d. If one looks at the output shown in Figure 4.40d, the detection of two ellipses for one leaf is due to the large amount of deviation of the plant leaf from an elliptical shape. For the 3-leaf plantlet depicted in Figure 4.38e, the Arc detection method was able to identify 4 ellipses, see Figure 4.40e. Certainly, this plant has 3 notable leaves, however, the method also detected a 4th small leaf that is still growing. For the 4-leaf plantlet in Figure 4.38f, the Arc detection method found 7 ellipses that best adjust to the contour of the shape, see Figure 4.40f. If one looks carefully, one can see that this is a similar case to the 2-leaf plant previously mentioned because the ellipse fitting technique found more than one ellipse per leaf.

The 3Point ellipse detection is a method that is able to match ellipses within a given shape. However, it takes a significant amount of time because it has to examine all possible ellipses and group them in to clusters. Furthermore, one has to take into account that this technique produces random number of possible ellipses and every time different results are produced. Additionally, when big ellipses and small ellipses are in the same contour, the 3Point algorithm will find the big ellipses first and might not even detect the small ones, because the number of 3-point random trials is not large enough. Thus making it necessary to calculate more 3-point random trials, which in turn produces more information that has to be analyzed.

After considering the various pros and cons of each ellipse detection algorithm, it was decided that the appropriate decision was to use the Arc detection method in the experimental phase of the plant detection problem.

4.6.2 *Nicotiana tabacum*: leaf detection evaluation and sensitivity analysis

This section presents the evaluation of the Arc detection method for matching tobacco leaves (*Nicotiana tabacum*). Different circumstances and different parameters are presented in order to give a general idea of the behavior of this algorithm. The two main parameters in the evaluated technique are: (1) minimum arc length and (2) the area of each ellipse that must be included in RoI. The first parameter has the purpose of eliminating the small arc sections that might generate faulty ellipses. The second parameter controls the quality of the ellipses generated by the combination of two arc sections. For example, if two arc sections are combined and produce an ellipse that has more than a defined x percentage of its pixels outside the RoI, then these arc sections produce an ellipse that does not fit to the examined shape.

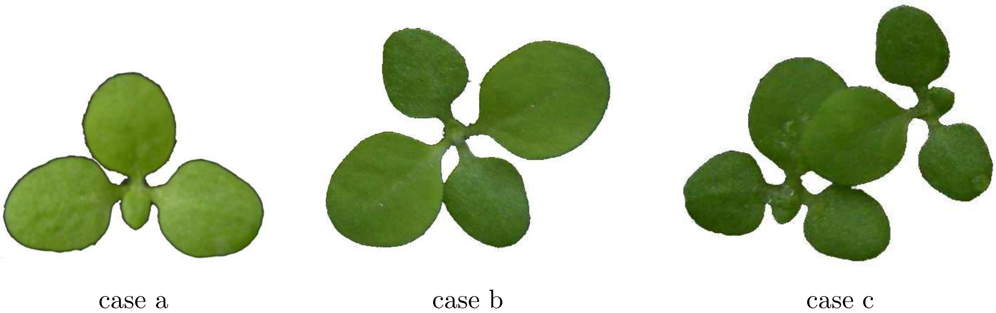

case a case b case c

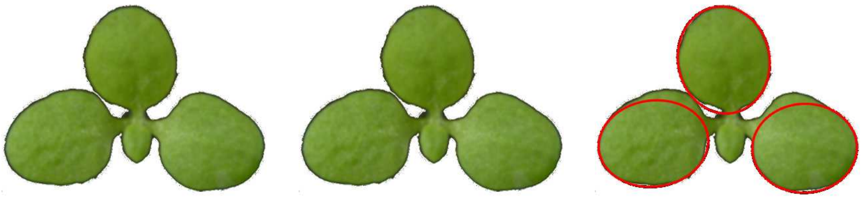
Figure 4.41: Cases used to evaluate the Arc ellipse detection.

In order to perform the parameter evaluation of the detection method, one of the parameters has to be fixed, while the other is changed and studied. First the evaluation of the arc length will be carried out. For this case, the total number of pixels of each ellipse that must be contained in the RoI is set to 97 %, this means that only a small number of pixels is allowed to be outside of the region of interest (3 %). Figure 4.41 shows the three cases that will be used to execute the evaluation.

The evaluations were carried out for the following arc sizes: 200, 100, 50, 25, 12 and 6 pixels. The results of the evaluations are presented in a series of images that depict the detections, the time required to find the ellipses and the total number of ellipses detected. Again, the experiments were executed using a computer with the following core: Intel(R) Pentium(R) D CPU 2.66GHz with 1 gigabyte of RAM.

Figure 4.42: Case a: analysis of a contour that contains 1309 pixels.

min. Arc length	200	100	50
Detected ellipses	0	0	3
Time (s)	0.03	0.04	0.12



min. Arc length	25	12	6
Detected ellipses	5	5	6
Time (s)	0.27	0.33	0.53

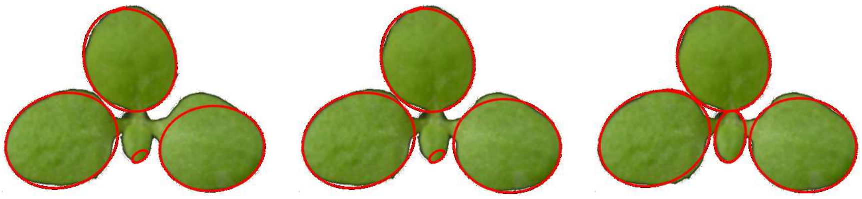


Figure 4.43: Case b: analysis of a contour that contains 1692 pixels.

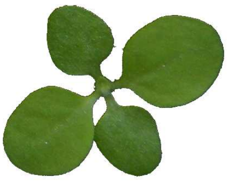
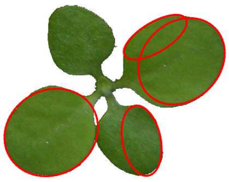
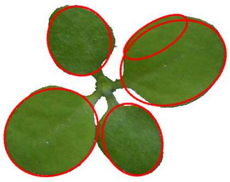
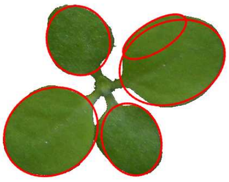
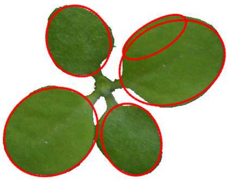
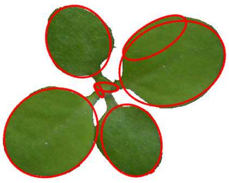
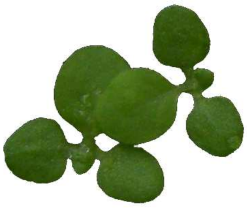
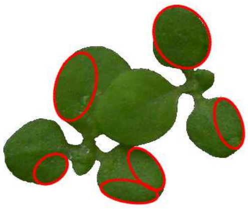
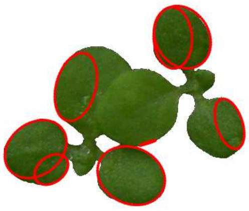
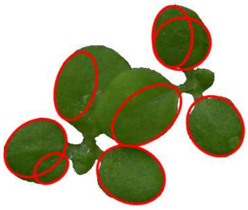
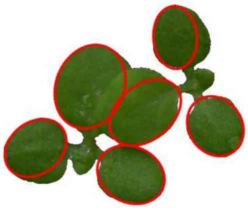
min. Arc length	200	100	50
Detected ellipses	0	4	5
Time (s)	0.04	0.16	0.22
			
min. Arc length	25	12	6
Detected ellipses	5	5	9
Time (s)	0.33	0.54	0.73
			

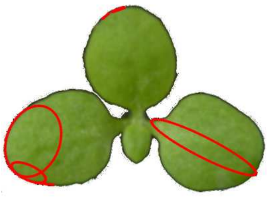
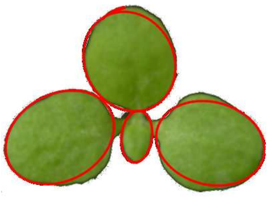
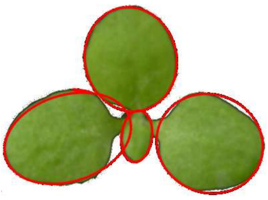
Figure 4.44: Case c: analysis of a contour that contains 1556 pixels.

min. Arc length	200	100	50
Detected ellipses	0	0	6
Time (s)	0.04	0.06	0.24
			
min. Arc length	25	12	6
Detected ellipses	8	8	6
Time (s)	0.36	0.44	0.55
			

In all cases of the above evaluation, the minimum arc size of 200 pixels proved to be a very high threshold. This situation resulted with zero detected ellipses, because none of the above cases had at least one arc with 200 pixels or more. When the minimum arc length was reduced to 100, 4 ellipses were detected in case b. This means that there were several arc sections with 100 or more pixels. With a minimum length of 50 pixels, more ellipses superimposing the leaves can be seen. The same effect happens when the minimum length is reduced to 25, 12 and 6 pixels. Every time the arc size is reduced, the number of arcs increases, along with the number of combinations that the algorithm has to perform in order to find the best ellipse combinations. If small arcs are allowed to be part of the search for ellipses, one can expect that these small arcs complement the information of bigger arcs, thus improving the accuracy. However, if these small arcs cannot be combined with other arcs, they might produce spurious ellipses that will affect the leaf detection.

The evaluation of the second parameter (area of each ellipse that must be included in RoI) was carried out using the same computer with a fixed minimum arc size of 6 pixels. The cases that were analyzed are: 100 %, 98 %, 93 %, 88 %, 83 % and 78 % of pixel area included in the RoI. Furthermore, the same plant cases as in the previous parameter evaluation were used, see Figure 4.41.

Figure 4.45: Case a: analysis of a contour that contains 1309 pixels.

Area in RoI	100 %	98 %	93 %
Detected ellipses	6	6	5
Time (s)	1.39	0.69	0.43
			

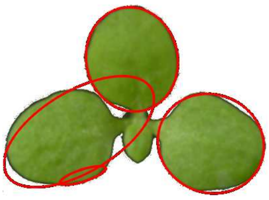
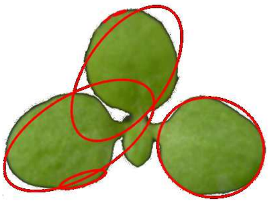
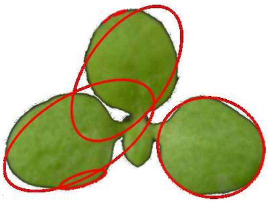
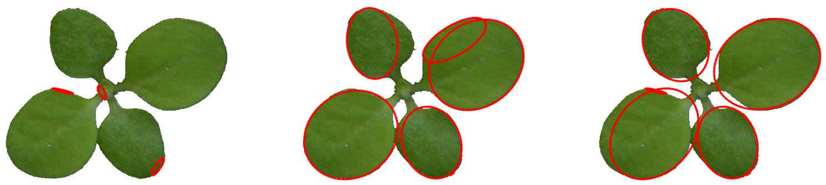
Area in RoI	88 %	83 %	78 %
Detected ellipses	4	6	6
Time (s)	0.36	0.36	0.35
			

Figure 4.46: Case b: analysis of a contour that contains 1692 pixels.

Area in RoI	100 %	98 %	93 %
Detected ellipses	3	6	6
Time (s)	1.13	0.8	0.46



Area in RoI	88 %	83 %	78 %
Detected ellipses	6	6	6
Time (s)	0.45	0.43	0.38

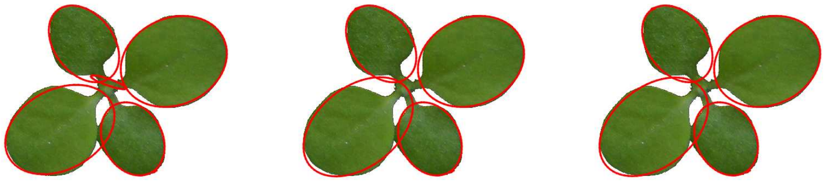
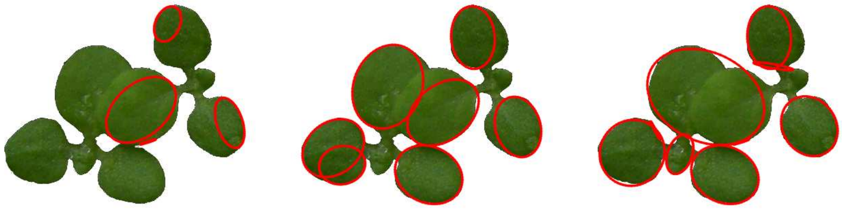
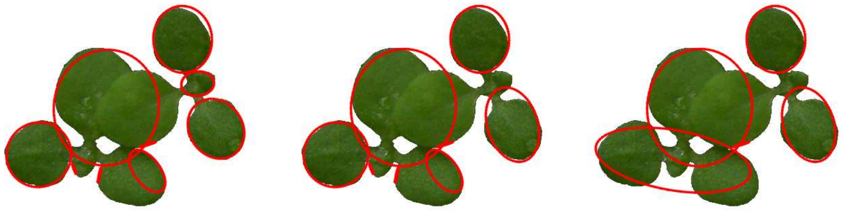


Figure 4.47: Case c: analysis of a contour that contains 1556 pixels.

Area in RoI	100 %	98 %	93 %
Detected ellipses	4	7	9
Time (s)	0.81	0.57	0.46



Area in RoI	88 %	83 %	78 %
Detected ellipses	9	8	5
Time (s)	0.32	0.31	0.32



From the three above evaluated cases, one can see that the algorithm has a poor performance when 100 % area must be included in the RoI. This is due to the fact that the leaves are not perfectly elliptical and this value is too rigid. Additionally, the time used to compute the result was the longest (in each case) because there are no arc combinations and every arc is treated as a new possible ellipse. When the parameter is changed to 98 %, a notable change can be seen in the detections: the leaves are correctly superimposed by ellipses. In this case, by just allowing an error of 2 %, the arc combination was successfully executed in all three cases. Of course, one can also observe that the time of execution is the second longest (in each case). For the rest of the parameter values (93 %, 88 %, 83 % and 78 %) one can see that if too much area is allowed to be outside of the RoI, erroneous ellipses are detected. This effect is caused by the incorrect combination of arcs, because the parameter value is too flexible. It was observed that good values for this parameter are within the range [96 %, 99 %] depending on how elliptical the shape is that one is detecting and how rigorous one wants the parameters to be.

It is worth mentioning that the parameter values used during this research are: (1) Minimum arc size = 6 pixels. (2) Minimum area in RoI = 97 %. The value of 6 pixels (or points) are the minimum necessary to approximate an ellipse. This means that the smallest possible arc sections are taken into consideration for the combinations. 97 % of area on the RoI is a value that was especially fitting for the detection of tobacco plantlets. The estimation of this value was carried out by running the ellipse detection algorithm on several different cases.

4.6.3 Example detections of other type of leaves

In this section, several examples of leaf detection on other types of plants are shown. The purpose of executing leaf detection on plants other than the model (*Nicotiana tabacum*), is to demonstrate that the previously mentioned method to detect ellipses also works on other plant varieties.

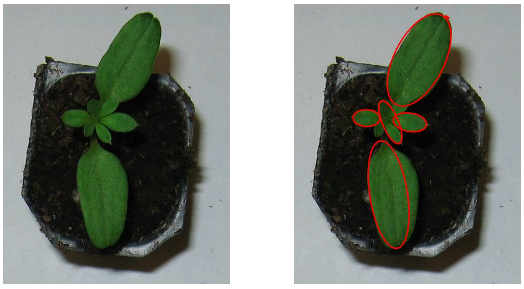


Figure 4.48: Leaf identification on *Galium aparine*

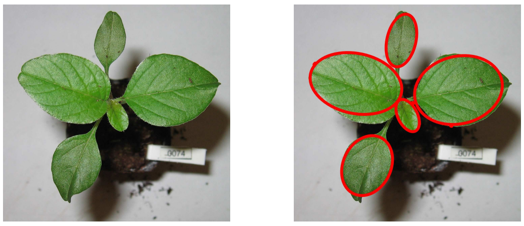


Figure 4.49: Leaf identification on *Amaranthus retroflexus*

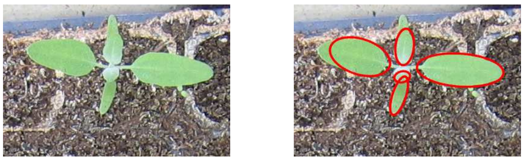


Figure 4.50: Leaf identification on *Chenopodium album*

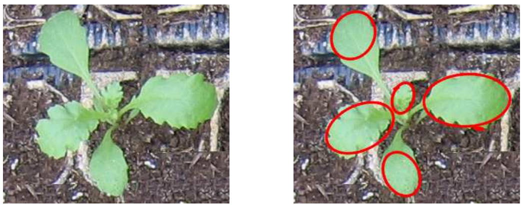


Figure 4.51: Leaf identification on *Senecio vulgaris*

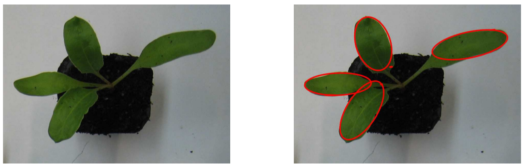


Figure 4.52: Leaf identification on *Beta vulgaris*

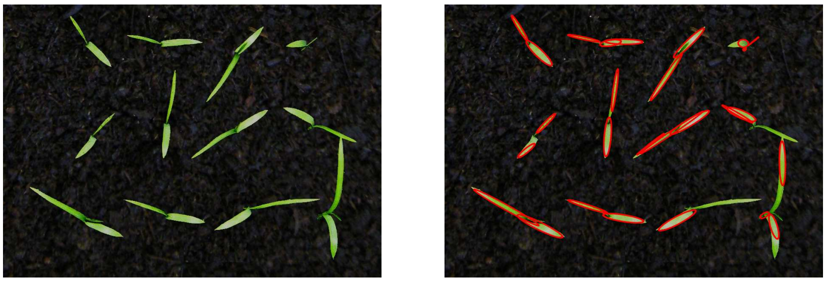


Figure 4.53: Leaf identification using *Apera spica-venti*

From the above images, it is possible to see that the proposed method to identify leaves works on plants different to the model plant, the present identification includes the two major groups of flowering plants: monocotyledons and dicotyledons. Naturally, this opens the door to many applications in the field of precision agriculture. It is worth noting that the parameters used to compute these detections are the same as the ones mentioned in the previous section, with the exception of the color segmentation parameters that had to be adjusted due to the different lighting conditions in which the images were taken.

4.6.4 Experimental results

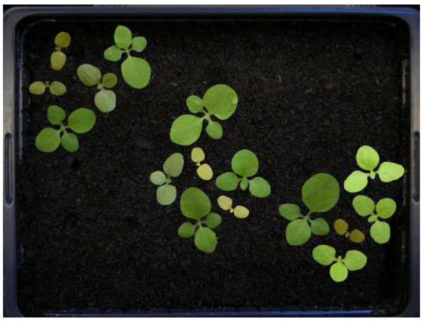
This section presents the results of the ASM *Nicotiana tabacum* plantlet detection using the previously described algorithm. Several case studies were analyzed, using images sets with groups of plantlets with 0 %, 0.5 %, 1 %, 2 %, 4 %, 8 %, 16 %, and 32 % of overlapping area.

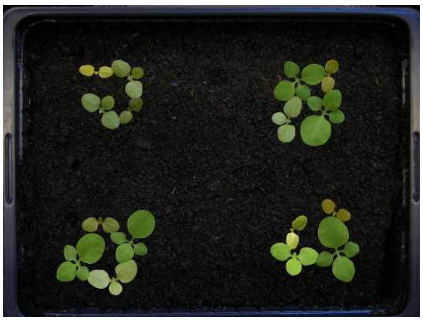
These image sets were generated using image processing software that allowed taking different pictures of plantlets that were stored in a database and combining them to create groups with a similar amount of overlapping area. In order to create an image series, a set of ca. 16 plantlets was created, these same plantlets were used in all the images that belong to a particular image set. An image set contains 8 images, one image for each amount of overlapping.

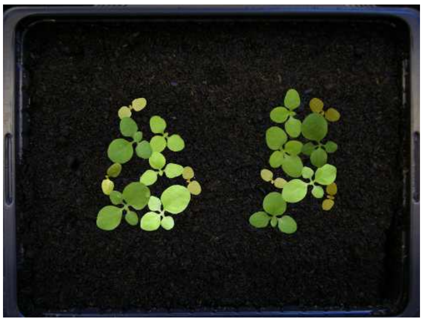
A single image can contain several different groups of plantlets, but each of these groups (within a single image) must contain approximately the same amount of occlusion. Moreover, the image sets must contain the same number of groups. During the experimental phase, images containing 2, 3, 4 and 5 groups of plantlets were analyzed.

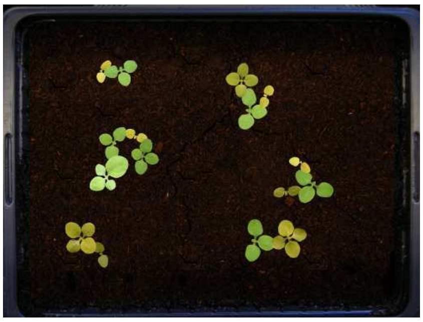
To create an overlapping group of plants, first the *total pixel area* of the non-overlapping plantlets that will form the set is computed. For instance, if a group of four plants is to be created where plantlet *a* contains 200 pixels, plantlet *b* 350 pixels, plantlet *c* 400 pixels and plantlet *d* 250 pixels, then the total pixel area is the combined sum of these individual plantlets, in this case 1200 pixels. Then the plantlets were positioned so that desired percentage of the total pixel area is occluded. In the above described set of four plantlets with 0 % overlapping, no plantlet touches another. With 1 % overlapping area, 12 pixels have to be occluded. This overlapping has to be spread over all the members of the group, i.e. each plantlet has to be overlapped by at least one other, but not necessarily by the same amount. The same process is executed for each group of plantlets in the image, until ca. 16 plantlets are present. The number of plantlets in each group within a single image can vary but can contain a maximum of 8 plantlets.

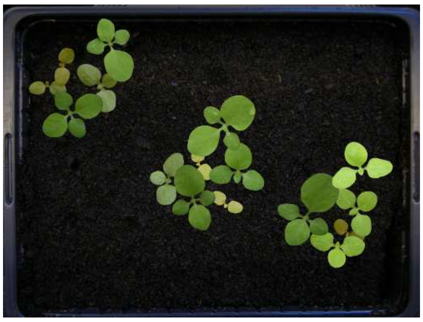
Below, some example images from the case study are depicted:

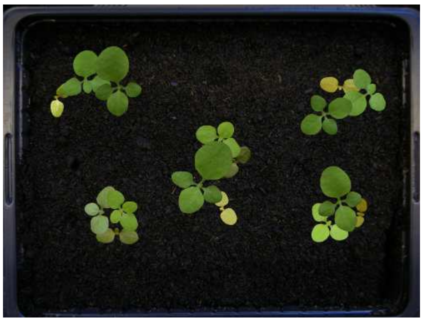

0 % overlapping example

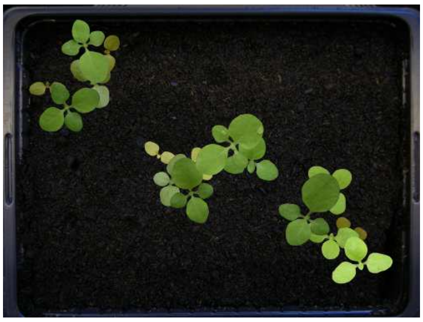

0.5 % overlapping example

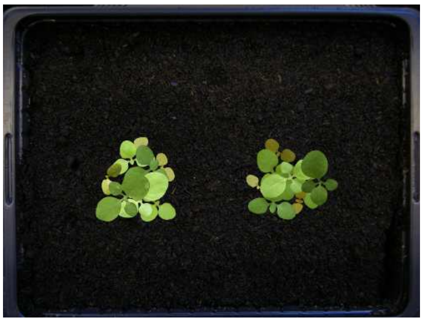

1 % overlapping example


2 % overlapping example


4 % overlapping example


8 % overlapping example


16 % overlapping example


32 % overlapping example

The amount of overlapping area increases until a maximum of 32 % is reached. It was decided to stop the analysis at this point because the information that one can extract about the occluded area is insufficient due to extreme overlapping of the plant parts. It is worth noting that this method only uses the region of interest extracted with the segmentation procedure shown in section 4.5.4. This is much less information than the human brain can interpret from the same image.

Figure 4.54, demonstrates the type of visual information (RoI) which is used by the algorithm to detect the plants. Even for the human brain, such an image with limited data can be quite confusing.

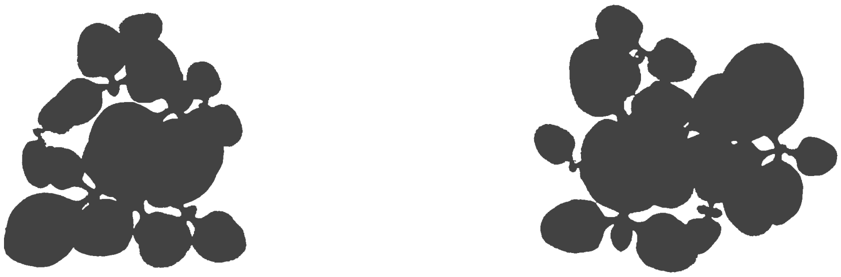


Figure 4.54: Examples of the visual information that the plant recognition algorithm uses to match plantlets. These 2 regions contain tobacco plantlets with 32 % of overlapping area.

4.6.4.1 Evaluation of the identification Algorithm

After executing the detection algorithm using all of the created images, a straightforward evaluation of the detections was used. This evaluation system provides a percentage of the effectiveness of the presented algorithm: It compares the number of actual leaves with the detected ones, and how they were grouped into plants.

1. Count all the leaves within the image of 0 % overlapping. This number will be used to evaluate the image sets that contain these plants.
2. Run the detection program using each image within a set.
3. Evaluate each detected plant as follows:
   (a) if the plant and all of its leaves was correctly detected, one point is awarded for each leaf of the plant.
   (b) if a leaf is identified but not used to create a plant, no point is awarded for that leaf.
   (c) if a leaf is not identified, no point is awarded for that leaf.
   (d) if there is a faulty detection of the leaves and/or plants, then negative points are given.

   i. If all leaves are correctly identified but assigned to the wrong plant then one point is awarded for each correct leaf while a negative point is awarded for each leaf that is placed with the wrong plant.
   ii. If multiple detections of a single leaf are found (more than one ellipse per real life leaf) and this leads to the use of a single leaf in multiple plant detections, then a minus point is awarded for each incorrect plant allocation.
   iii. If multiple ellipses are found for a single leaf but only one ellipse is used to create a correct plant identification, then no negative points are awarded.

4. The sum of all positive and negative points is divided by the total amount of leaves in the image, thus quantifying the effectiveness of the program.

4.6.4.2 Results using artificially generated situations

Figure 4.55 depicts the results after evaluating the actual detections of the ASM algorithm. As one can see, with no overlapping, the method is able to detect plants with 2, 3 and 4 leaves with almost 100 % accuracy. However, with increasing overlapping, the performance of the method decreases until only ca. 21 % of the plants are recognizable with 32 % of overlapping area. The greatest amount of difficulty was encountered when analyzing overlapping plantlets in one large group. This is because there is a greater number of leaf combinations that can lead to the identification of faulty plants. When smaller groups of overlapping plants are analyzed, the algorithm was more readily able to correctly identify the leaves and match them to the corresponding plant.

One more aspect to mention is that much of the success or failure of this plant detection technique depends on proper leaf identification. If the outline of a shape did not provide enough information to perform a correct ellipse detection, there will most likely be a poor outcome from the plant detection algorithm. Naturally, when the plantlets start overlapping each other, the information that the shape border supplies becomes insufficient.

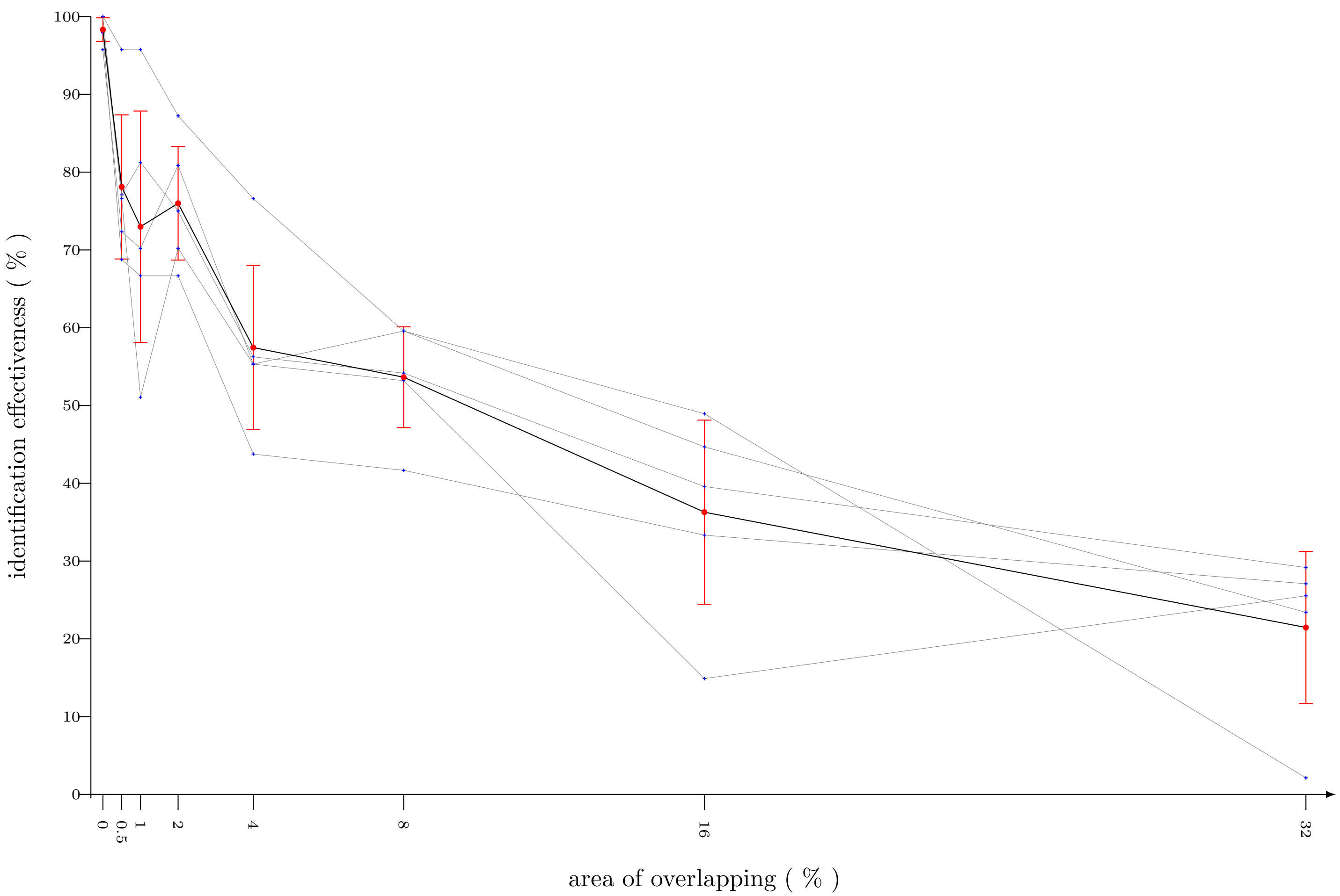


Figure 4.55: Results after evaluating the ASM recognition system. The thin gray lines represent analyses of individual image sets. The thick black line and red dots represent the mean value. Also, the vertical red lines depict the standard deviation.

4.6.4.3 Comparison with human recognition

Since the ultimate goal of machine vision is the creation of systems that perform as well as the human vision or better, it was decided to test the human recognition system and compare it to the ASM identification system presented in this thesis.

In section 4.5.2 it was mentioned that two groups of people (*Experts* and *Non-experts*) are used to make the comparison. This sections also specifies how the experiments are carried out.

The answers presented by each person were analyzed in the same manner as for the ASM system (see section 4.6.4.1). The results of the experiment are shown in Figure 4.56.

With regard to the Expert group, it can be observed that this group was able to solve, almost with a 100 % accuracy, all cases with up to 8 % overlapping. When the overlapping reaches 16 % of the total area, the human identification system started failing, however, it still managed to reach a 93 % accuracy. In the cases with 32 % overlapping. the general reaction from the participants when they first looked at the situations was "It is not possible to identify a plantlet here!". Nevertheless, when they paid more attention, they were able to solve the task with a 69 % accuracy.

The group of Non-experts had a much lower identification performance. In the cases with 0 % overlapping this group started with almost a 94 % accuracy, and then the ability to identify the plants started dropping until it reached ca. 34 % accuracy for the cases with 32 % overlapping.

At the end of the experiment, the participants said that without the color information, it is very cumbersome to perform the identification. Moreover, the participants also admitted that for the cases with heavy overlapping, they had to guess in order to properly identify some of the plantlets.

Figure 4.56 compares the results of the ASM plant detection scheme and the human recognition system. With this image one can see that adult humans are very well trained to solve these situations. There is an inherent learning ability in every human that allows them to process and use all the available visual information to perform excellent pattern recognition. Despite this fact, sometimes the proposed ASM algorithm has better results than the non-expert group.

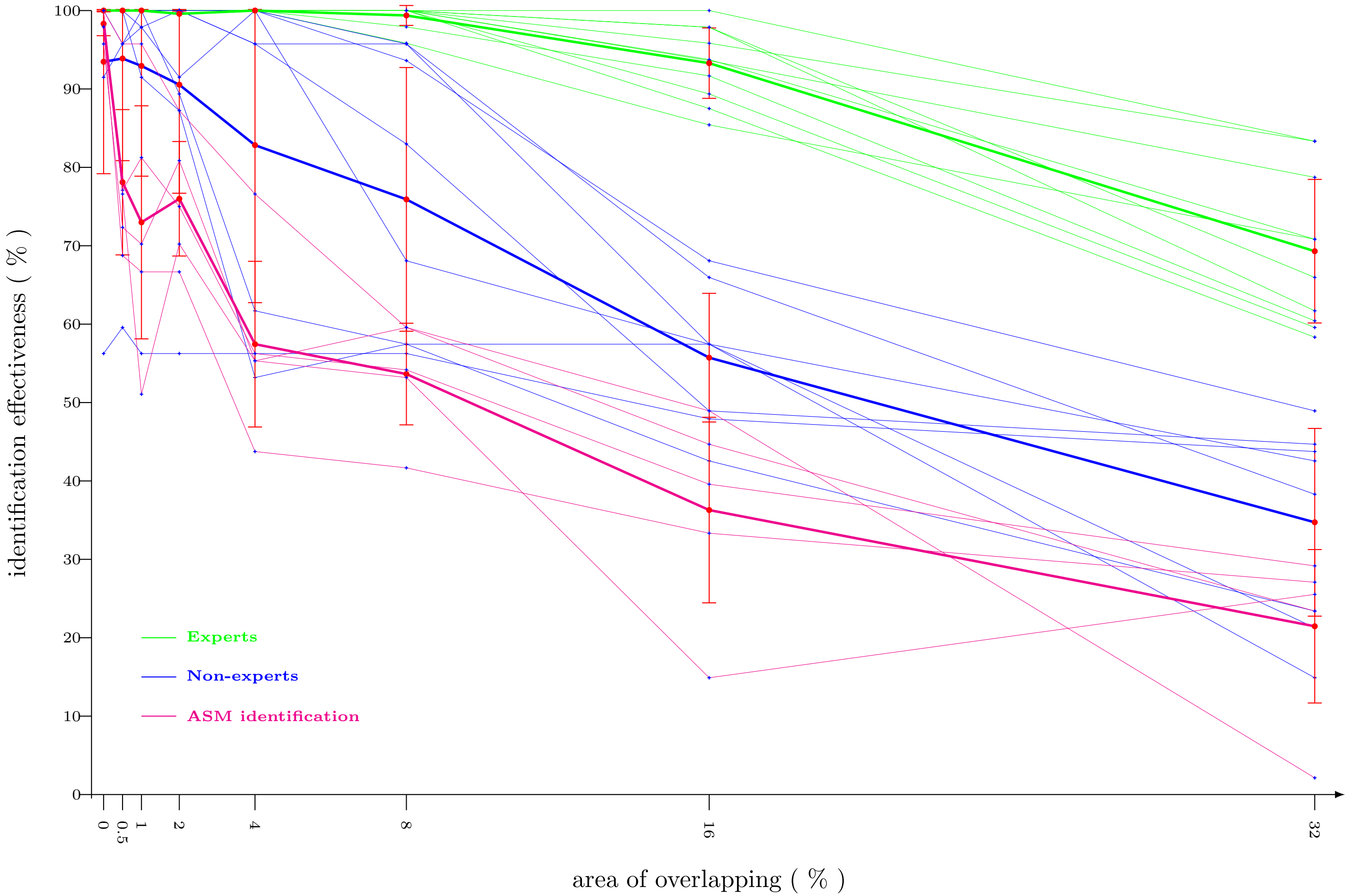


Figure 4.56: Results after evaluating human recognition of plants under overlapping situations. The gray thin lines represent the recognition performance of a single person. The colored thick lines represent the mean performance of the different groups of persons (Experts and Non-experts), as well as the ASM identification algorithm.

4.6.4.4 Results using real situations

This section presents the results after running the *Nicotiana tabacum* detection algorithm on plants growing in rectangular containers (see section 4.5.1). The presented cases are a series of images taken over a period of two weeks after the germination of the tobacco seeds. A total of 6 cases, with different planting arrangements (see Table 4.4) will be described, and finally a graph containing the results of the detection will be presented.

Case	Arrangement	Number of germinated plants
	columns×rows	
R$_A$	5×5	32
R$_B$	6×6	36
R$_C$	6×4	21
R$_D$	5×6	28
Rnd$_1$	random	17
Rnd$_2$	random	31

Table 4.4: Description of the trays with real plants.

Each case is presented as a series of 7 images, taken two days apart from each other, making a total of 12 days. The first images depict small plants without any overlapping; as these plantlets grow, they will start mutually overlapping. As would be expected, each arrangement represents a different problematic, where the hidden areas are different and unknown. Figure 4.57 presents 3 of the previously mentioned study cases.

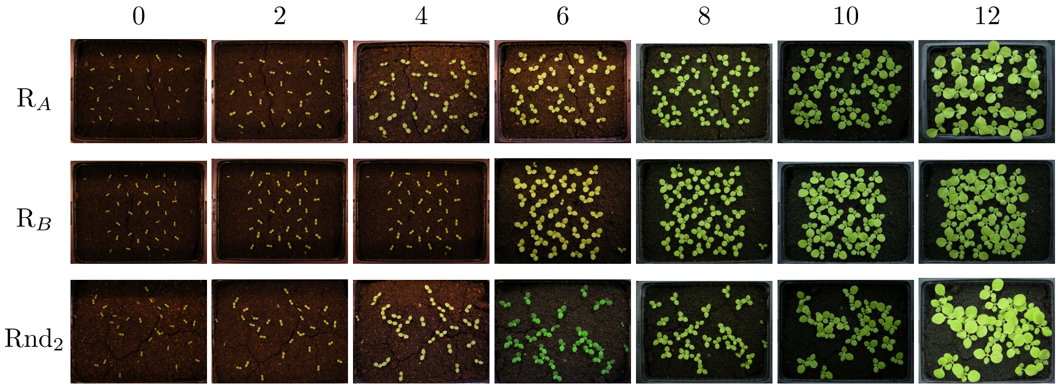


Figure 4.57: Example of a series of images showing trays with real plants over time (days). This images depict the growing stages of the tobacco plantlets within a 12 day period of time. See Table 4.4 to find out more about the example cases R$_A$, R$_B$ and Rnd$_2$.

Figure 4.58 shows an example of the detection carried out on one of the sample images.

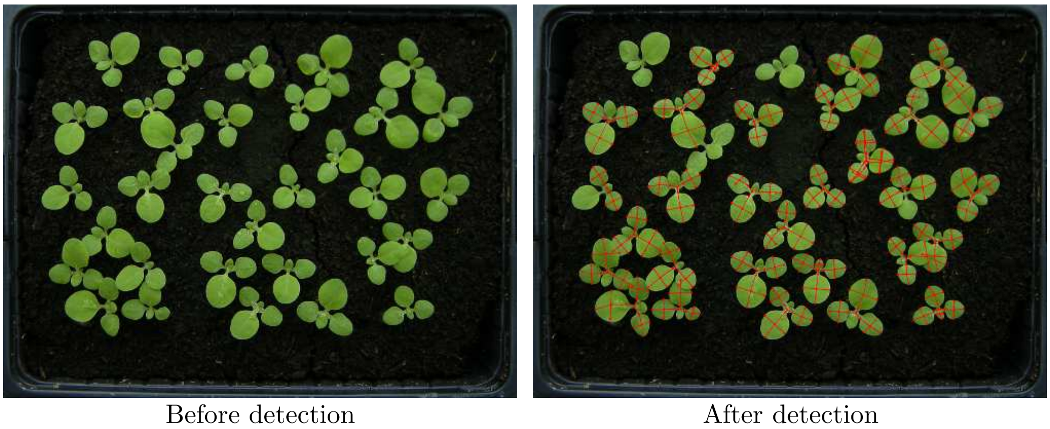

Before detection After detection

Figure 4.58: *Nicotiana tabacum* detection example.

The following plot shows the result obtained after running the ASM identification algorithm on each of the 6 cases introduced in Table 4.4.

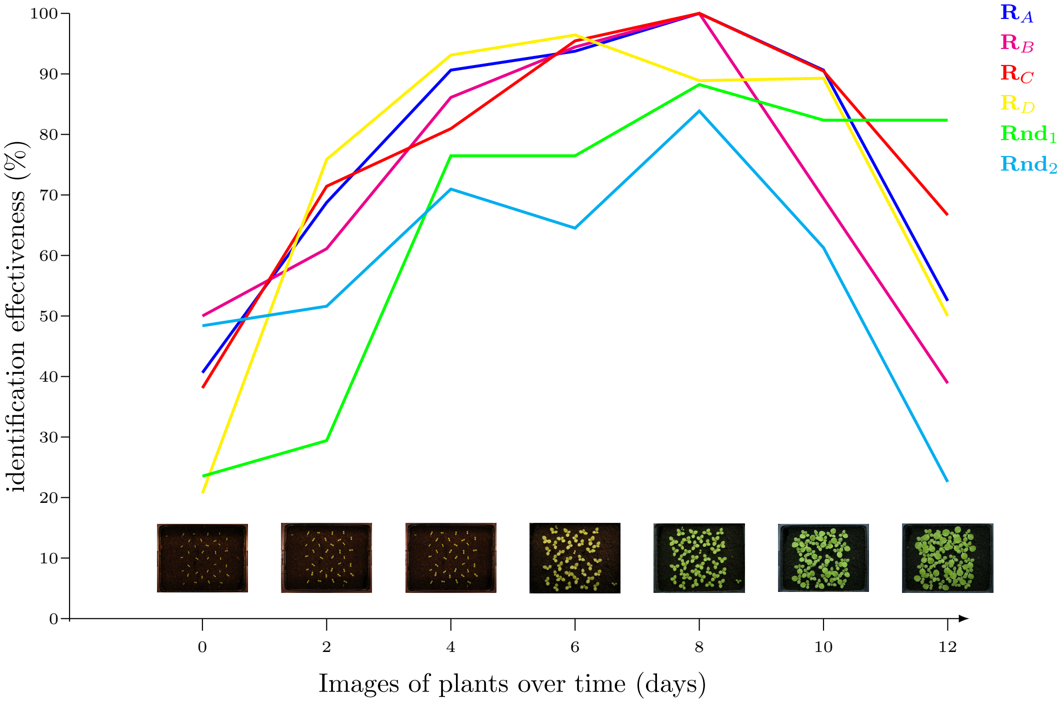


Figure 4.59: Plant recognition on real cases of *Nicotiana tabacum*.

From the above plot, one can see that in the early early stages of growth, when the tobacco plantlets are not fully developed, the ASM algorithm has a low performance in detecting these small plants. This is because the shapes included in the training sets were of older, more mature plants (see section 4.5.1). As the plants grow, the performance of the detection algorithm starts increasing. Likewise, the plant overlapping also starts increasing until it stars affecting the accuracy of the detection method. It should also be noted that in order to be able to compare the results, all the images were taken with the same resolution, something that could lead to resolution problems at the germination stages because of the small size of the plants.

CHAPTER 5

Taking the ASM to the 3D space

5.1 Overlapping in the 3D space

The previous chapter presented the problem of identifying plants under overlapping situations using 2D images. In actuality, these plants are objects that exist in the 3-dimensional space. Occlusions on 2D images and/or overlapping situations occur when perspective views of 3D objects are projected onto a 2-dimensional space, mapping some of the information on the same position and generating information overlap. The same problem is observed when 2D objects are projected onto a 1-dimensional space. Figure 5.1 shows how a cube is projected onto a 2-dimensional space, where the result is a square.

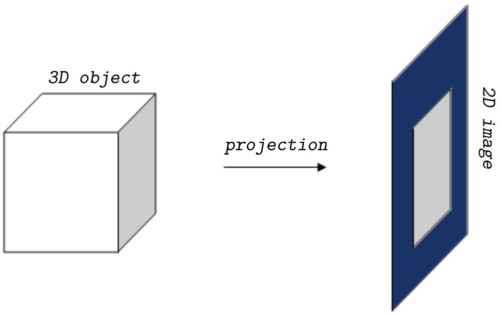


Figure 5.1: Projecting a 3D object onto the 2D space.

In general, the problem of overlapped or occluded information arises when one projects N-dimensional object(s) onto spaces with $(N-1)$ dimensions. The physical tangible space consists of only 3-dimensions, and no more. This means that the 4th dimension necessary to project objects onto a 3-dimensional space is missing, thus, overlapping and occlusions do not exist in the 3-dimensional world. Physically, this is true for all solid objects because they cannot occupy the same space.

Nevertheless, if one considers empty spaces encapsulated within a solid object it is possible to start thinking about overlapping in the 3-dimensional space. First, one needs to assume that these unoccupied spaces have some type of shape. For example, Figure 5.2 shows an empty space in the form of a tube encapsulated within a solid structure, in this case a cube.

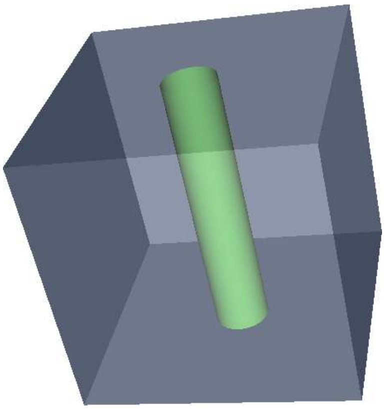


Figure 5.2: Empty space encapsulated in a 3D structure. The empty space is represented by a green tube and the encapsulating solid surrounding the empty space is represented by a cube.

Moreover, if two or more empty spaces are encapsulated within the same solid structure and they do not touch each other, one can conclude that they are independent structures. However, when these empty structures start crossing each other, one can presume that they start sharing space or, in other words, overlapping each other. This implies that the only way to have overlapping in the 3-dimensional space is when structures of empty space start sharing the same space. Figure 5.3 gives a visual example of 2 empty space structures, represented by tubes, overlapping each other in a 3-dimensional solid space represented by a cube. The section where the 2 tubes intersect each other is the space where the 2 structures overlap.

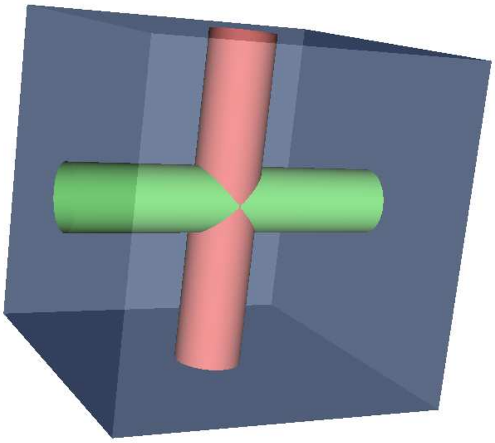


Figure 5.3: Example of overlapping structures in the 3-dimensional space. The solid section is represented by a cube. The 2 empty structures are represented by red and green tubes. Together they form a complex overlapping system.

In reality, one can recreate the above example by drilling similar holes in a solid cube. Certainly, these situations can also be observed in nature, for example: when an earthworm burrows into the soil, or by the growing roots of a plant. These cavities, left by worms or roots, are created at different times and locations, however, it is very probable that at some point several cavities will share overlapping space.

In order to create 3-dimensional structures of these overlapping spaces, the use of tomographic imaging methods is required. Furthermore, if one wants to analyze their shape, a methodology similar to that provided by the 2D ASMs is needed, which in this case has been modified for the 3-dimensional space. Additionally, depending on the size of the structure, sometimes it is better to exclude the very large and complex overlapping forms, which can detrimentally influence and mislead the results of any shape recognition methodology.

5.2 Related work

According to Herman (2009, p. v), tomographic imaging has been extensively used in the last 3 decades in many different scientific fields, specially in medicine and related areas, archeology, biology, material science, topography, etc. There are several modalities of imaging, for instance: Magnetic Resonance Imaging (MRI), X-ray Computed Tomography (CT), Positron Emission Tomography (PET). All of these methods can be used to create a virtual 3-dimensional model of the object being analyzed. However, only after advances in computer technology in the last two decades has it been possible to process the tremendous amount of data produced by such analyses (Defrise, 2001, p. 71).

Statistical shape models, such as ASMs, are a popular tool for the representation of 3D shape variations, and are widely used in the field of medicine, for example for modeling human organs and bones. Figure 5.4 shows a 3D model of a healthy human liver. This information can be used to build a comprehensive database with the shape of healthy human organs in order to differentiate them from sick ones, making it possible to detect diseases during the early stages of development.

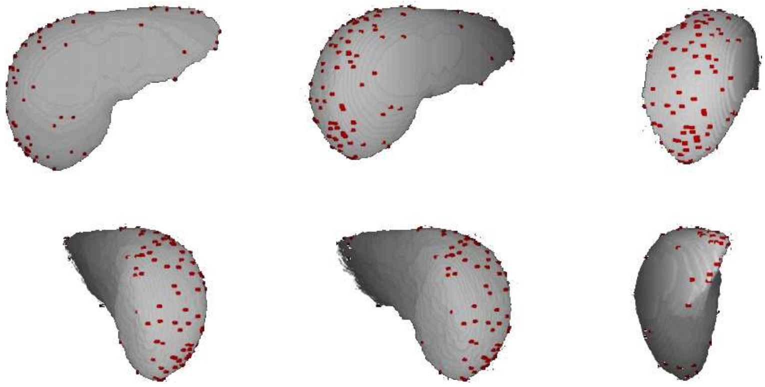


Figure 5.4: 3D Active Shape Model of a human liver. This figure shows different perspective views of the same mean shape, where the red dots represent the landmarks used to build the statistical model of a healthy liver. Image from Lim et al. (2006, p. 5).

In addition to medical applications, ASMs are also used in the field of 2D and 3D face recognition. For example, Caunce et al. (2010) used statistical models to improve the identification of faces.

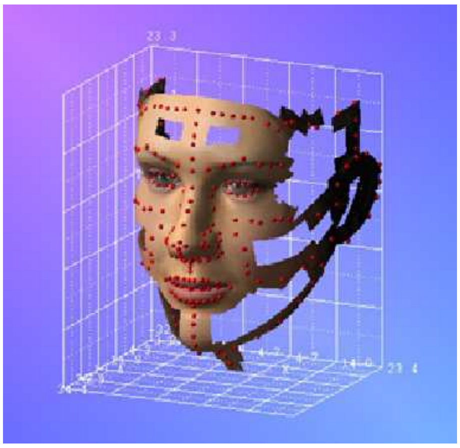


Figure 5.5: 3D Active Shape Model of a human face. The red dots represent the landmarks used to build the model. Image taken from Caunce et al. (2010).

The theory behind 2D and 3D ASMs is the same. Nevertheless, the landmarking procedures in the three dimensional space are more complicated, depending on the shape of the objects. Figure 5.5 shows a three dimensional model of a face with its corresponding landmarks.

When using an statistical model to identify structures that have a defined shape, such as faces, body organs, teeth, bones, etc., the locations to place landmarks are homologous in each structure. However, when one is working with objects that do not have an specific shape, the landmarking procedure can be quite challenging due to the correspondence problem. Approximating shapes that have a large amount of variability as geometric forms (i.e. lines, cubes, ellipsoids), greatly decreases the complexity of such structures and allows the use of straightforward landmarking methods.

3D ASMs in plant science

3D Active Shape Models have also found their way into the area of plant science. For example, Southam et al. (2009) generated models of plant organs (*Arabidopsis thaliana*) in order to understand the effects of genes in shape morphology during different growing stages. However, the landmarking approach used in that research was carried out completely manually.

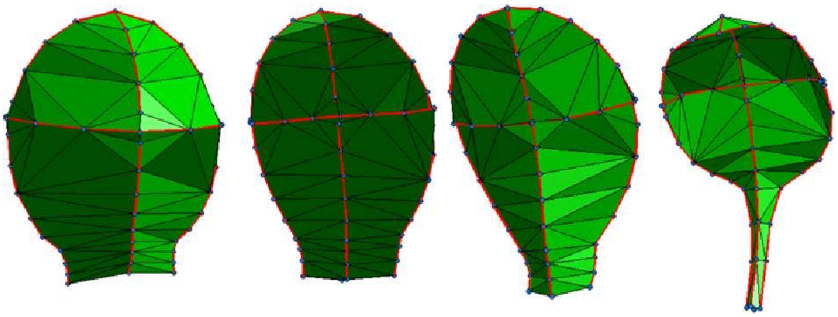


Figure 5.6: 3D Active Shape Models of Arabidopsis leaves. The blue dots represent the landmarks used to build the model. Image taken from Southam et al. (2009, p. 3).

Even though computed tomography and ASMs have been extensively used to analyze different objects, there are no approaches that use active shape modeling to investigate 3D overlapping problems.

5.3 Objectives

The objective of this part of the research is to transfer the 2D ASM methodology presented in previous chapters into the 3D space to find specific shape parameters that will allow the characterization of overlapping structures.

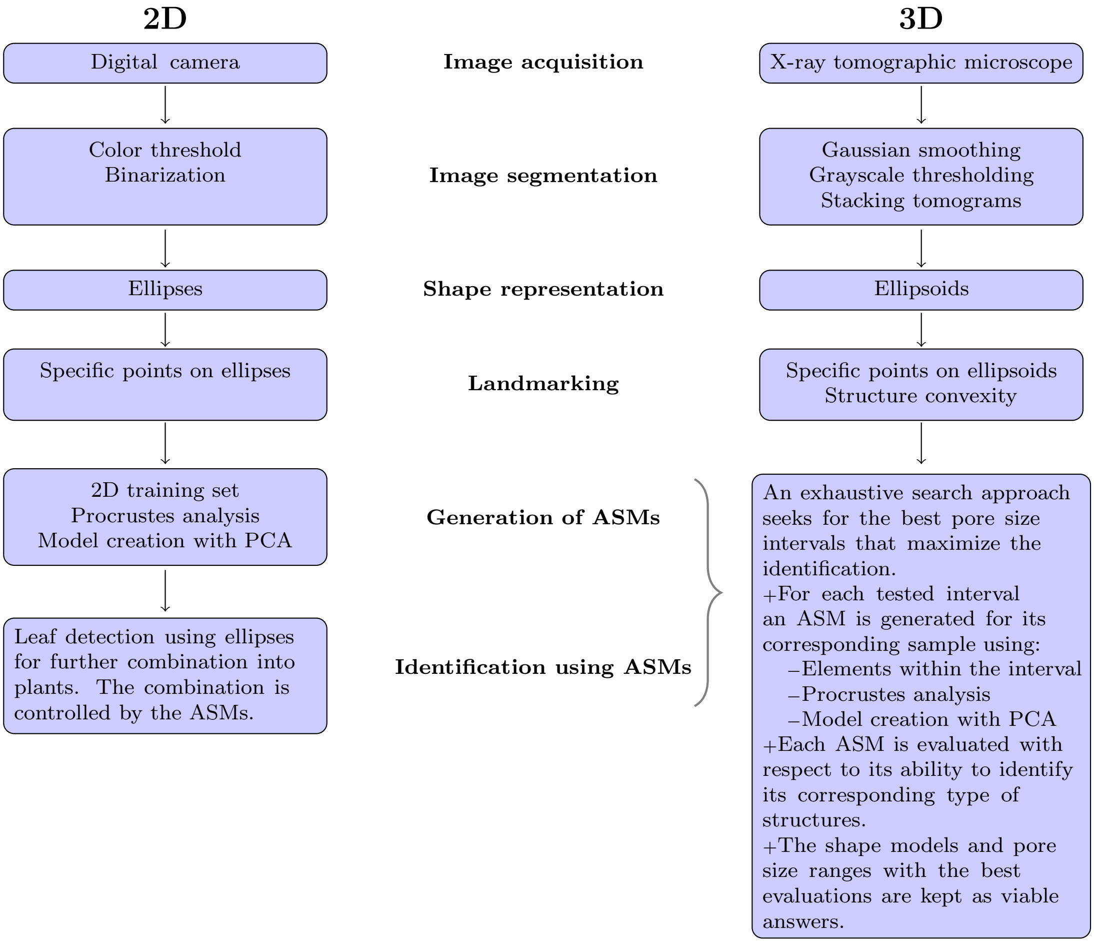
In plant science, there are several possibilities that one could investigate to transport the method, for example:

1. The analysis of inter-cellular spaces or inter-cellular chambers in different parts of the plant, i.e., leaf, stem, root.
2. The analysis of soil cavities created by root growth.
3. The analysis of the xylem pathways, which are in charge of the water and nutrient transportation within the plant.
4. The analysis of micro pore structures contained in soil and rocks.

In this investigation, the last case was selected because of the availability of high resolution images made by state of the art tomographic systems, which enable the reconstruction and analysis of such overlapping forms. These micro pore structures are essentially cavities left by the movement of organisms, solids, liquids and/or gases through a compact mass. This has the problem that a single cavity can belong to more than one pore structure, thus, the pores can partially occupy the same space. In other words the pores can overlap each other, creating very complex 3-dimensional amorphous shapes.

General idea on how to transport the methodology

The following set of steps gives a general description of the procedure followed in the implementation of the 3D ASMs to accomplish the identification of specific characteristics that micro overlapping systems (micro pores) might have. Moreover, each step will be placed side by side with the corresponding 2D procedure in order to give a more comprehensive idea of the process.



Leaf detection using ellipses for further combination into plants. The combination is controlled by the ASMs.

An exhaustive search approach seeks for the best pore size intervals that maximize the identification.
+For each tested interval an ASM is generated for its corresponding sample using:
  –Elements within the interval
  –Procrustes analysis
  –Model creation with PCA
+Each ASM is evaluated with respect to its ability to identify its corresponding type of structures.
+The shape models and pore size ranges with the best evaluations are kept as viable answers.

5.4 State of the art of micro pore analysis

The analysis of pore structures in soil and rocks is very important for plant sciences because they influence plant nutrition, water transport and O_2 uptake. In addition, is worth mentioning that some researchers have already put forward investigations concerning microtomography of pore structures. Just to mention some: Pierret et al. (2002) put forward a method to perform 3D reconstruction and quantification of macropores structures. Nakashima & Kamiya (2007) presented a technique to analyze the connectivity between pore structures. Sugiyama et al. (2010), Gallucci et al. (2007) and Burlion et al. (2006) analyzed the impact that pore structures have in the leaching process and mechanical properties of some materials. Finally, Masaphy et al. (2009) studied the effect of pores in the transport and element transfer in rocks.

5.5 Materials and methods

5.5.1 Rock and soil samples

The samples used in this work are part of a soil science project at the Leibniz University Hannover, and are used in this work to determine the shape characteristics of overlapping pore structures using 3D ASMs. In total 3 rock and organic-based soil samples were used in this investigation: the soil sample (ISR) was basically composed of sand and ground limestone; one of the rock samples was basalt (N8), while the other one was a granite mainly composed of feldspar (DAH).

It is worth mentioning that the data used to carry out this reseach was provided by S. Dutlz.

5.5.2 Image acquisition

Cylindrical sections with a diameter of 2.1 mm were extracted with the help of a corer. After extracting the samples, they were mounted on a specially designed metallic holder that facilitated their manipulation by the X-ray tomographic microscope. Figure 5.7 depicts a sample prepared for the tomographic analysis.

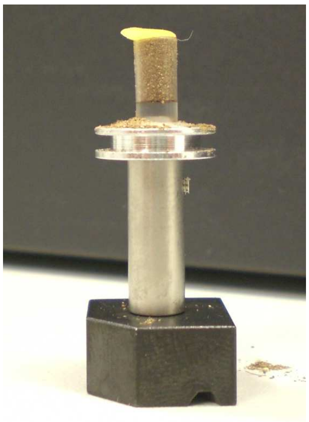


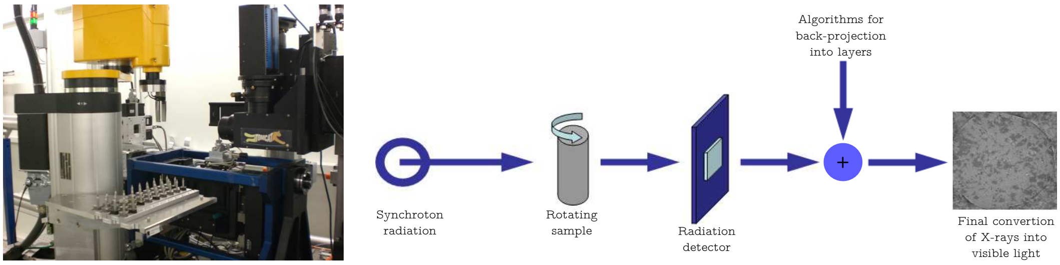
Figure 5.7: Sample prepared for tomographic analysis (Image by T. Rath 2010).

The data was acquired using a state of the art Synchrotron-based X-ray tomographic microscope at the TOMCAT (TOmographic Microscopy and Coherent rAdiology experimenTs) beamline at the Swiss Light Source of the Paul Scherrer Institute, Villigen, Switzerland (see Figure 5.8). This new type of technology has opened the doors into to the nanoscale world (Sakdinawat & Attwood, 2010). There are several study areas that have already performed research using this type of synchrotron microscopes, i.e., materials science, biological sciences, environmental science, archeology, paleontology, and, in this case, plant science.

Paul Scherrer Institute

Figure 5.8: SLS - Swiss Light Source at the Paul Scherrer Institute, Villigen, Switzerland. (Image from the PSIWeb (2011)).

The TOMCAT beamline has 3 principal components: the synchrotron radiation beamline, the sample manipulator and the radiation detector. The beamline bombards the rotating sample with X-rays; the radiation transmitted from the sample is then recorded by the detector (or X-ray camera), which is finally converted into visible light information i.e. 2D images, also known as tomograms (Stampanoni et al., 2006), see Figure 5.9. A tomogram is defined as a "two-dimensional image representing a slice or section through a three-dimensional object" (MedTerms, 2011).


Algorithms for back-projection into layers
+
Synchroton radiation
Rotating sample
Radiation detector
Final convertion of X-rays into visible light

(a)

(b)

Figure 5.9: TOMCAT beamline. (a) Image of the beamline. (b) Schematic representation of the image acquisition procedure. Images by T. Rath 2010.

The acquired images from the X-ray tomographic microscope are in a gray scale format, very much like any other X-ray picture. The lighter parts represent the more dense areas (the ones with more absorption), whereas the darker sections represent less dense areas (the ones with less absorption); this implies that the voids in the rock samples are the dark pixels in the tomograms (Marone et al., 2009; Stampanoni et al., 2006).

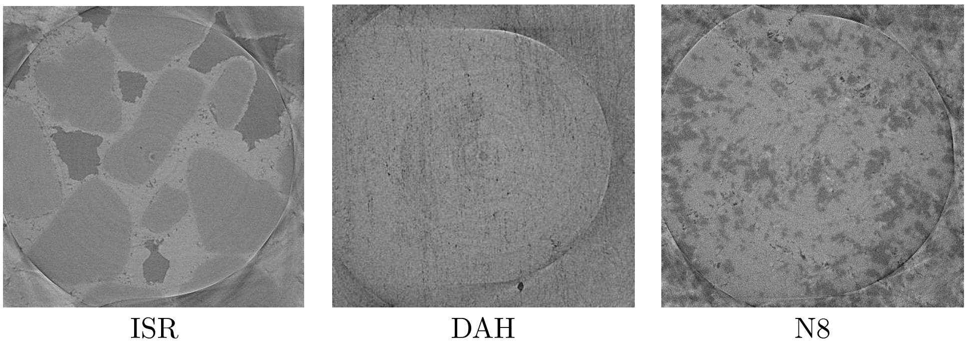


Figure 5.10: Example layers acquired with the X-ray tomographic microscope and used in the 3D reconstruction. Each image corresponds to different soil/rock samples. Images from the data provided by S. Dultz.

By looking closely at the images in Figure 5.10, one can realize that the tomograms have a circular ring, also called ring artifacts, caused by the tomograph back-projection. “Dead pixels in a CCD chip and damaged scintillator screens are responsible for sharp and marked vertical stripes in sinograms, which back-project to half circles (ring artifacts) in tomographic reconstructions” (Münch et al., 2009, p. 19). In order to eliminate this faulty information, just the circular sections inside the ring were considered when performing the 3D reconstruction.

In the samples used in this work, a set of 1024 gray scale tomograms were produced from each sample using X-ray tomography. Figure 5.11 shows an example of circular sections stacked on top of each other. These images have a square pixel size of $0.7{\times}0.7\ \mu\text{m}^2$. Additionally, the tomograms were separated by a distance of $0.7\ \mu$m. This means that if one stacks several tomograms on top of each other, the voxel[1] size would be $0.7{\times}0.7{\times}0.7\ \mu\text{m}^3$.

The techniques used in 3D reconstruction can be subdivided into two main groups: **volume reconstruction**, which is used in full object reconstruction, and **surface reconstruction**, which is used in terrain exploration. Tomographic reconstruction falls into the category of volume reconstruction, where objects are generated by stacking parallel tomograms. The information belonging to the object of interest is extracted from each tomogram and then recombined to form the 3D structure.

---

[1] “A voxel (volumetric pixel or, more correctly, Volumetric Picture Element) is a volume element, representing a value on a regular grid in three dimensional space.” (Wikipedia, 2010d)

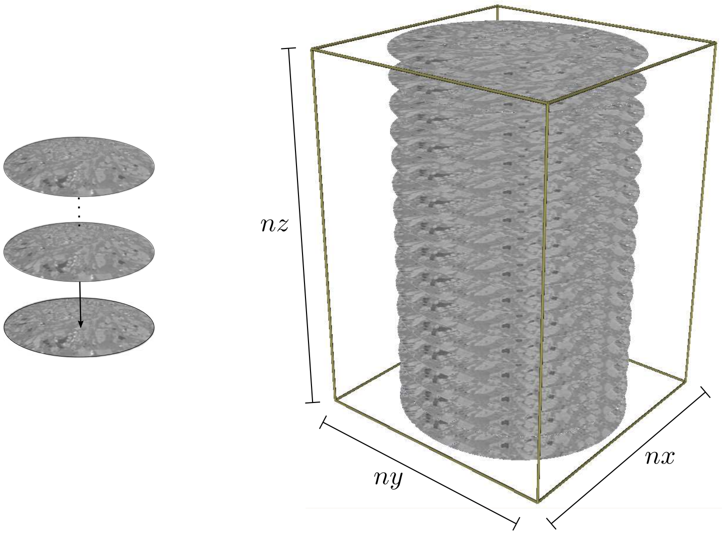


Figure 5.11: Example of a stack of tomographic images acquired at the TOMCAT beamline. The size of the structure in voxels is $nx \times ny \times nz$.

5.5.3 Image segmentation

5.5.3.1 Gaussian smoothing

Due to the noise introduced by the physical properties of the X-ray tomograph, the tomograms were subjected to noise reduction using a Gaussian filter. For each sample, several images were individually analyzed with different Gaussian kernel sizes. After visually comparing the outputs and their corresponding histogram, the kernel size was selected for each sample type; see Appendix A.10 for more information on how to implement this process.

Figure 5.12 shows a close-up of a section of one of the tomographic images before and after the process.

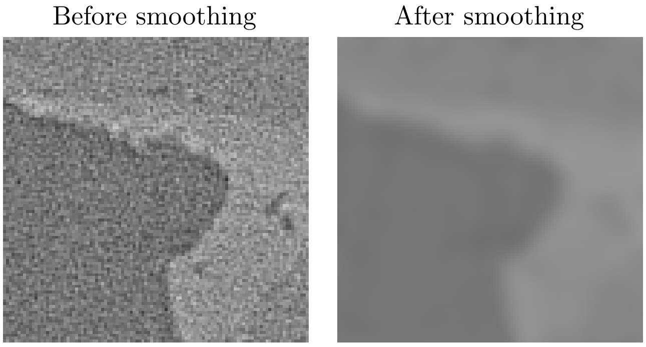


Figure 5.12: Close-up on the effect of Gaussian smoothing on a tomographic image that belongs to the soil sample ISR.

In addition to the previous example, the following images with their respective gray scale histogram demonstrate the effect of the Gaussian filter on one of the tomogram images from sample N8. It is worth noting that this smoothing process is applied to each layer separately.

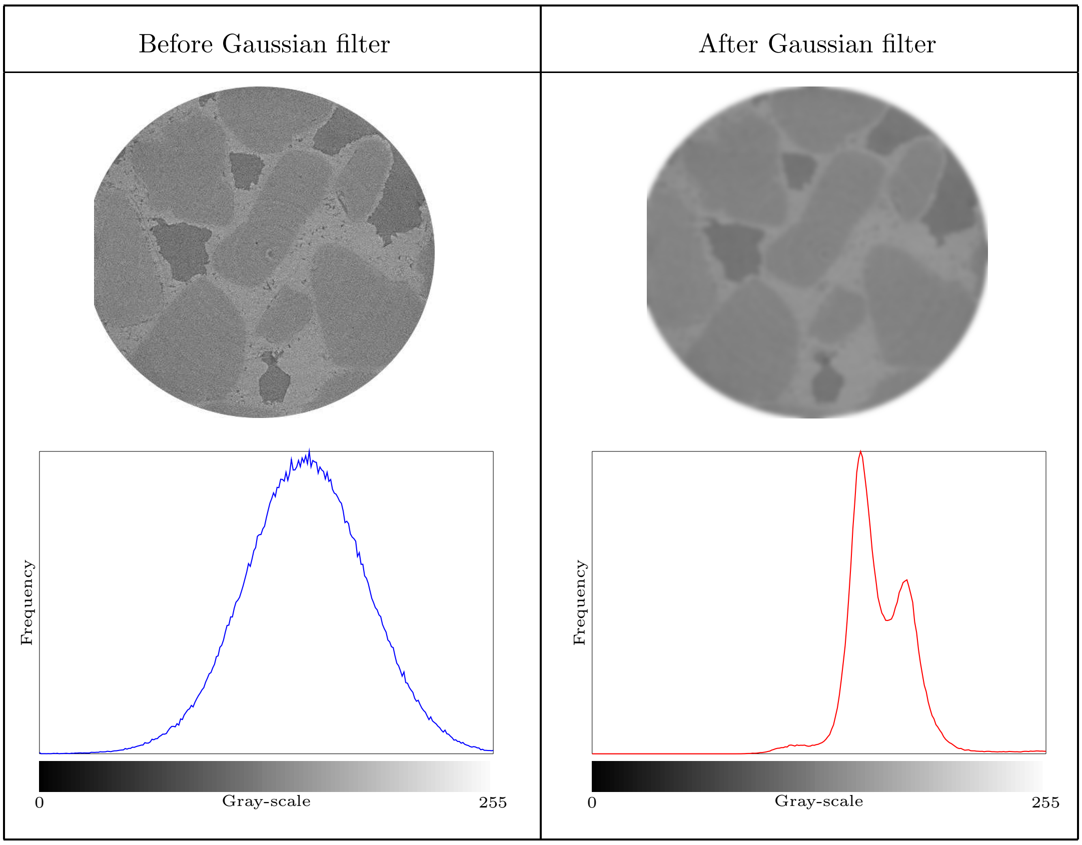


Figure 5.13: Effects of the Gaussian filter on on a tomographic image. This Figure also shows the respective gray scale histograms that helped with the identification of pores and minerals.

5.5.3.2 Grayscale threshold

After the Gaussian smoothing, gray scale thresholds were used to separate the pores from the minerals and other materials. In the above gray scale histogram plots (Figure 5.13), one can see how several peaks appear in the histogram after the noise reduction filter. These peaks (local maxima) and nadirs (local minima) represent the variation in color in the gray scale image, which in turn can be allocated to different types of minerals and the voids. This characteristic of the gray scale histograms can then be used as a basis to establish thresholds that separated the pores from the minerals. Due to the varying chemical composition of each of the materials, several tomographic layers from each sample were visually inspected and the thresholds manually adjusted to provide an optimal separation of the voids from the surrounding materials.

The following plots show an example of the resulting gray scale histograms, one for each sample type:

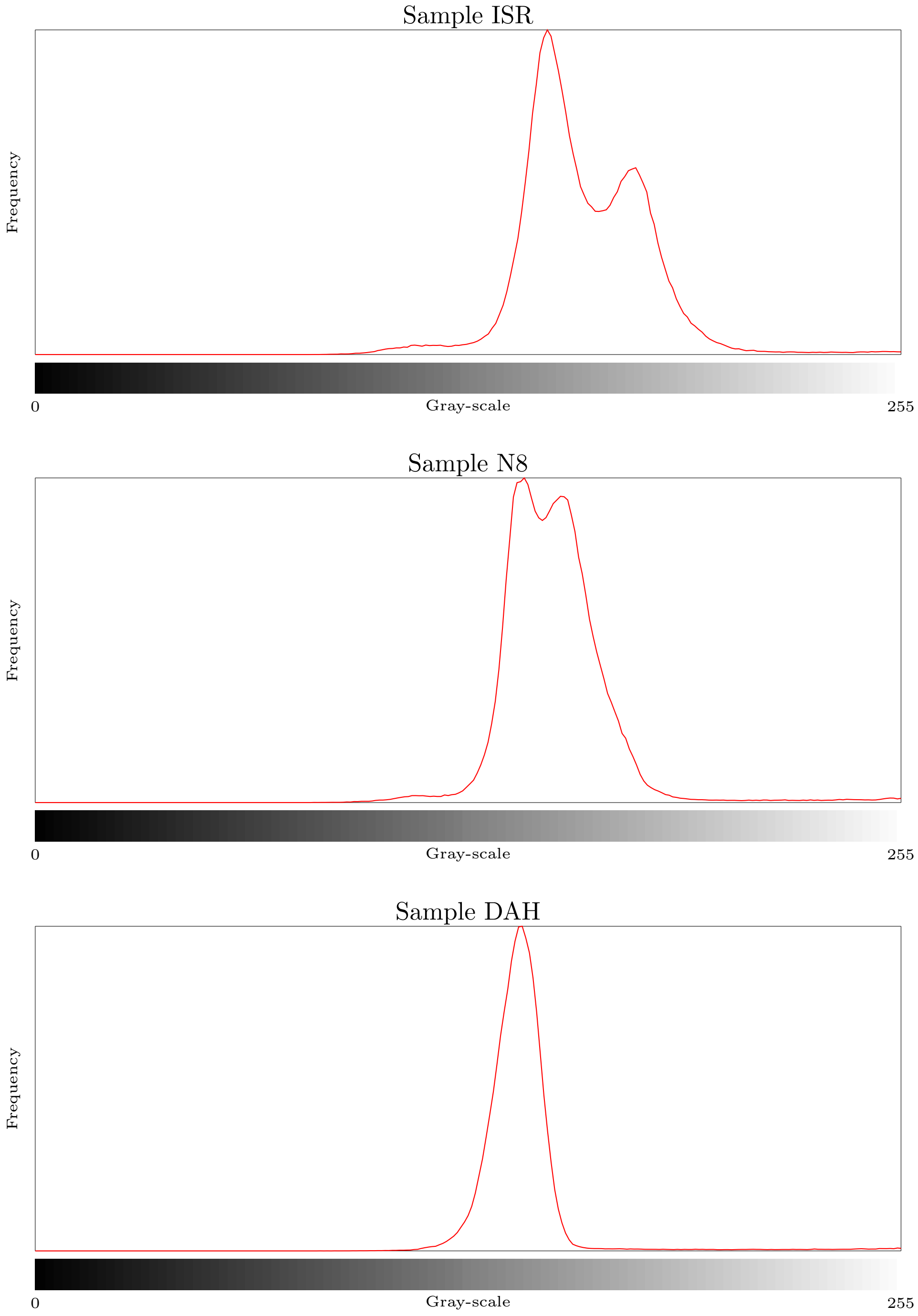
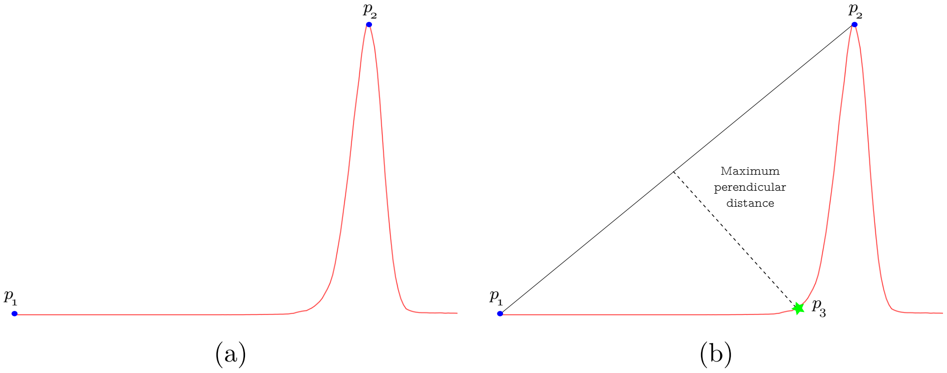


Figure 5.14: Histrogram plots that belong to 3 individual layers of the samples: ISR, N8 and DAH.

In the above plots (Figure 5.14) it is possible to see that sometimes the gray scale histogram does not provide clear information about the separation ranges, for example, the histogram of the DAH sample. In order to solve this situation the splitting technique for polygon approximation introduced in section 3.1.8 was used. This approach permitted the location of the breaking points on the curves, which in turn allowed a better visual tuning. Figure 5.15 shows an example of how to use the splitting technique.



(a) (b)

Figure 5.15: Histrogram analysis with the polygon approximation technique. (a) Originally, points p_1 and p_2 (a nadir and a peak respectively) do not provide accurate information when they are used as a gray scale range to locate the pores. (b) After locating the breaking point using the splitting technique, the new located point p_3 permits to establish new ranges, for example $[p_1, p_3]$, to locate the pores. The breaking point has maximum perpendicular distance to the straight line connecting p_1 and p_2.

Figure 5.16 depicts an example of the resulting pore information after applying the Gaussian filter and the gray scale thresholds to one of the tomographic layers from sample N8. This is a binary image that presents the information that this tomogram layer contains about the pore(s) that pass through it.

Figure 5.16: Tomogram layer after Gaussian filter and gray scale threshold. The dark areas represent parts of pore structures.

Once the Gaussian filter and gray scale threshold were defined for each sample, all the tomographic images in each individual sample were preprocessed in order to separate the regions of interest. Having extracted the desired data from each tomogram, one can proceed to stack the layers that contain this information, in this case the voids. For example, Figure 5.17 shows a perspective view of the resulting structure after stacking the 1024 tomograms from sample N8.

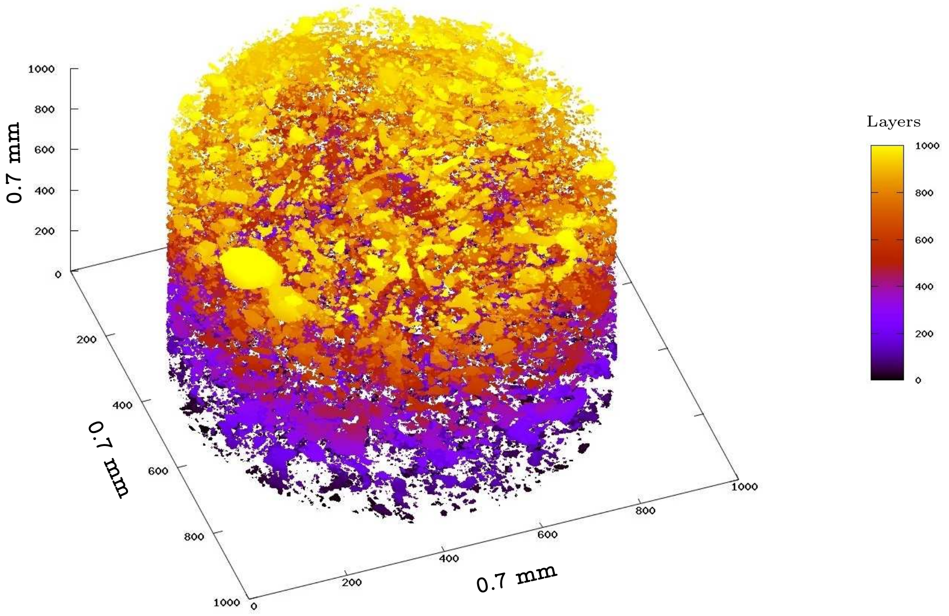


Figure 5.17: Perspective view of all pore structures after stacking 1024 preprocessed tomograms of the N8 sample.

5.5.3.3 3D Flood-fill

The 3D reconstruction of the voids was carried out with the help of a flood-fill algorithm. The flood-fill algorithm finds groups of interconnected pixels within an image. In its 3D version, it is capable of finding groups of interconnected voxels. These interconnected voxels represent the region of interest where the object (the pore structure) is contained. By definition, two interconnected voxels are those that have touching vertices. One can easily identify the neighbors of one voxel by placing it in the center of the $(3 \times 3 \times 3)$ cube, see Figure 5.18.

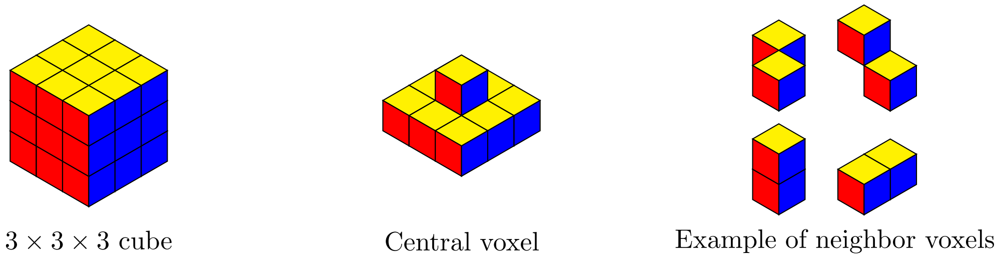


Figure 5.18: Cube to detect neighboring voxels $(3 \times 3 \times 3$ voxels). Additionally, the central voxel within this cube has a total of 26 neighbors.

The 3D flood-fill iteratively searches within the 26 possible neighbor voxels (see

Figure 5.18) every time a voxel with the desired information is found. Algorithm 5.1 indicates the steps that this method follows to find connected regions. In order to understand this procedure, one has to assume that the information extracted from all the tomograms was previously binarized and stacked in rectangular cuboids with a size of nx, ny, nz.

Algorithm 5.1 3D FLOOD-FILL

```
Input: A rectangular cuboid with a size of $nx$, $ny$, $nz$, where $nv$ the total number
    of voxels is $nv = nx * ny * nz$. Note that the information in each tomogram was
    previously binarized: 1 there is information, 0 there is no information.
    Execute the following steps for all $nv$ voxels in the rectangular cuboid.

    a: If a voxel with information (1) is found and it has not been located.

        1: Mark the voxel as located.
        2: Empty the voxel stack $S$ and the list of voxels $L$.
        3: Insert the voxel into the list $L$.
        4: Using the $3 \times 3 \times 3$ cube, look within its 26 neighbors for voxels with the
           same information (1).
        5: Mark these voxels as located.
        6: Insert the found voxels into the stack $S$ and into the list $L$.
        7: Execute the following steps until the stack $S$ is empty.

             i: Extract the top voxel of the stack $S$.
            ii: Using the $3 \times 3 \times 3$ cube, look within its 26 neighbors for voxels with
                the same information (1).
           iii: Mark these voxels as located.
            iv: Insert the found voxels into the stack $S$ and into the list $L$. Go to step
                (i).

        8: The list $L$ contains all the voxels corresponding to the current connected
           region and it is ready to be stored in a file or some other data structure.

    b: Select the next voxel in the rectangular cuboid and go to step (a) to find the
       next connected region.
Output: A set of $n$ connected regions.
```

By using the above described algorithm, several thousand pore regions were extracted from each tomogram set. In order to visualize an example of these 3D pore structures, Figure 5.19 is presented.

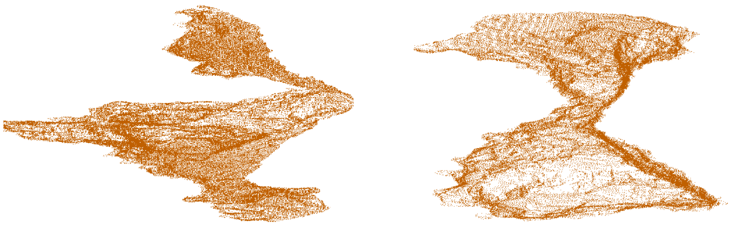


Figure 5.19: Example of extracted pore regions.

5.5.4 Shape representation

5.5.4.1 Introduction

This section presents different methods, which were applied in order to approximate the amorphous structures extracted from all the samples. The purpose for which these approximations were made is to reduce the complexity of the overlapping 3D structures by representing them using simpler geometric shapes. These approximations were carried out using 3 geometric object types: line, rectangular cuboid and ellipsoid.

Naturally, representing the shape of very complex forms with simpler geometrical shapes allows an easier comparison between the structures, and facilitates the use of simpler landmarking strategies in order to create statistical shape models.

For the purpose of creating the 3D ASMs, the aforementioned geometric objects were tested in order to learn which was the best representation for the majority of the structures. The decision to choose one geometric shape over the other was made taking into account the *error* that it generates when it is selected to represent an overlapping structure.

The *error* that one approximation produces is the overall sum of distances that each point has to the closest point of the geometric object, in this case a line, a rectangular cuboid or an ellipsoid.

5.5.4.2 Line representation

In order to represent a structure using a line segment, one needs to find the line that best fits all the points (voxels) contained in the 3D form. This task can be carried out, for instance, by using a least-squares regression or the Eigendecomposition method (Kahn, 1989).

After finding the 3D line that best represents the shape, every point that belongs to the analyzed pore structure is projected onto this line. Then, the *error* of this line section is calculated. Figure 5.20 depicts an example of an structure projected onto a line.

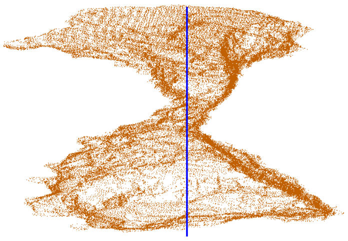


Figure 5.20: Pore structure approximated by a line.

5.5.4.3 Cuboid representation

When representing 3D structures with a rectangular cuboid, one has to find the minimum size of a 3D box that contains all the points within the form. Furthermore, in order to find the *error* that this representation produces, one has to calculate the distance between each point to its corresponding closest point on the box surface. This means that each voxel has to be projected onto each of the 6 faces of the box, and the projection with the minimum distance is the one that has the the closest point to the box. Figure 5.21 shows an example of an amorphous structure projected onto its containing box.

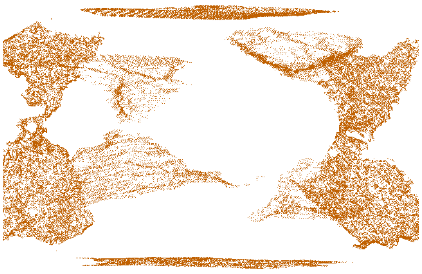


Figure 5.21: Pore structure approximated by a rectangular cuboid.

5.5.4.4 Ellipsoid representation

Just like in the previous cases, this approximation tries to represent all 3D forms with a geometric shape. However, in this case the geometric object is an ellipsoid. In order to carry out this task, one has to find the ellipsoid that best fits a given structure (see section 3.2.2.3). Then, as a means to calculate the *error*, one has to compute the closest point on the ellipsoid surface for each point. Figure 5.22 presents an example of other amorphous structure projected onto its ellipsoid approximation.

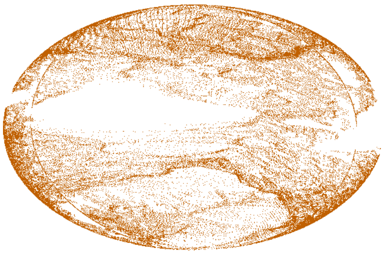


Figure 5.22: Pore structure approximated by an ellipsoid.

After analyzing the results of the 3D structure representation with geometric objects, it was decided to take the ellipsoid as the best fitting geometric shape. This was because over 99 % of the extracted structures were best represented by an ellipsoid, which was better than a line or a rectangular cuboid (see Table 5.1).

Overall score of each shape (%)		
Cuboid	Line	Ellipsoid
0	0.74	99.26

Table 5.1: Best fit approximation for each geometrical shape.

It is noteworthy that the line and cuboid approximations were better approximations when the structures were extremely small (less than 50 voxels). However, during this investigation the minimum pore size was set to 200 voxels, merely because it was desired to find recognition patterns within middle to large-sized shapes. One more reason for using a large minimum pore size is due to the image preprocessing on each tomogram, which affects larger 3D sturctures less than the smaller ones.

Having ellipsoids as the geometrical shape that represent the 3D structures greatly reduces the complexity of the shapes, so that it is easier to understand each structure. Additionally, it allows the use of the ellipsoid properties to generate a simple landmarking procedure, such as the one presented for the 2D case (see section 3.2.2). Figure 5.23 shows two examples of two amorphous structures approximated by ellipsoids, see sections 3.2.2.3 and 3.2.1.3 to find out more about ellipsoid and ellipse fitting.

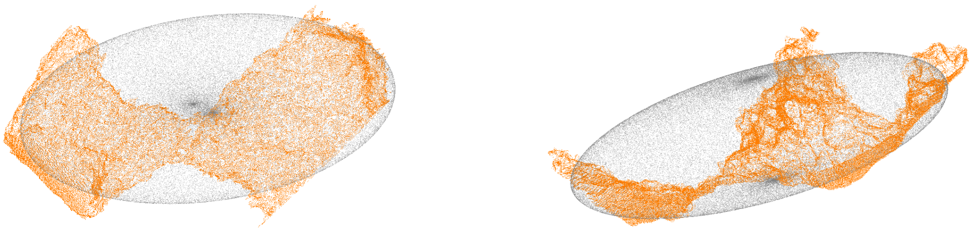


Figure 5.23: Examples of structures approximated by ellipsoids.

5.5.5 Landmarking

In order to create the 3D statistical model of the shape, or ASM, 15 landmarks were used: 1 that corresponds to the convexity of the structure and 14 landmarks that are extracted from each ellipsoid. The (x, y, z) coordinate landmarks that correspond to each approximated ellipsoid are: the semimajor axis $\pm a$, the semiminor axes $\pm b$, $\pm c$ and the 8 semilatus rectum points, see sections 3.2.1 and 3.2.2. Figure 5.24 depicts an example of a landmarked ellipsoid. Naturally, it was possible to select a smaller number of landmarks, for example: a, b, c and convexity. However, in this research it was decided to use 14 landmarks to best represent the form of each ellipsoid.

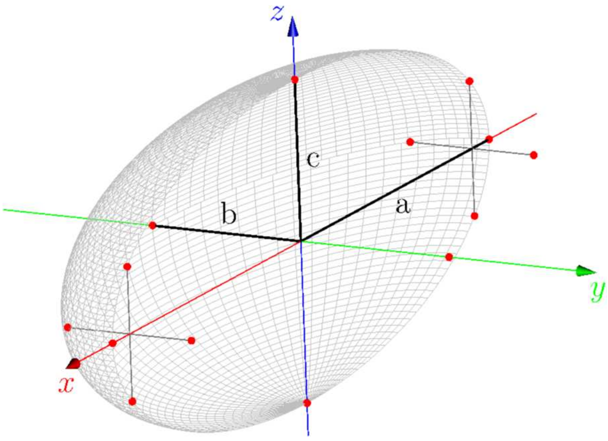
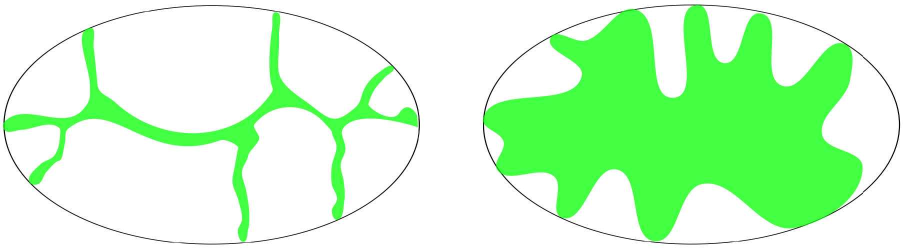


Figure 5.24: Landmarked ellipsoid showing the 14 extracted landmarks.

In addition to the landmarks that each ellipsoid will provide, the convexity of each structure is calculated for further use as a landmark. This was done because the convexity provides a hint about how the ellipsoid is filled. The calculation of the convexity is carried out by first computing the *convex hull*[2] of the given structure (Mehlhorn et al., 1996). Then, the volume of the structure is divided by the volume of the convex hull, giving the convexity value.

The range in which the value of the convexity falls is (0:1], where values close to zero mean that the structure has an elongated thin shape, whereas the values close to one mean that the structure is occupying the whole volume of its convex hull. Figure 5.25 shows an example of the convexity of two shapes. (a) Exhibits a convexity value close to zero because the shape does not occupy much of the space of the convex hull. (b) on the other hand depicts a shape with a convexity value close to 1 because the shape fills up most of the convex hull.



(a) (b)

Figure 5.25: Examplification of convexity values using an ellipse as the convex hull. (a) Shape with low convexity. (b) Shape with high convexity.

---

[2] A convex hull: “or convex envelope for a set of points X in a real vector space V is the minimal convex set containing X” (Farlex, 2010).

5.5.6 Exhaustive exploration approach

During this research, an exhaustive search was performed to find the best pore size intervals where the ASMs are able to identify the different pore samples. Of course, the space that is going to be explored is the collection of all pores extracted from all of the samples. The output of this search are specific pore size intervals where the shape model of each particular sample maximizes the identification between the structures of each sample.

5.5.7 Generation of ASMs and their evaluation

The generation of the Active Shape Models uses the methodology presented in section 3.3. The only difference is that the training sets are 3-dimensional (x, y, z). Furthermore, the Procrustes analysis, where the landmarks are aligned, was greatly simplified by reducing the complexity of the forms to an ellipsoid.

After generating a shape model, its ability to deform into other similar shapes is evaluated using the methodology shown in section 3.3.5. This means that in order to identify one particular shape all the deformation values b_i within the deformation vector $\mathbf{b}$ have to be bounded within the intervals $[-3\sqrt{\lambda_i}, 3\sqrt{\lambda_i}]$ created with the corresponding eigenvalues.

In section 5.5.6 it was mentioned that an exhaustive exploration was used, in which the evaluation of each ASM consists of the accumulation of all correct and incorrect identifications. A correct match happens when the ASM correctly identifies or rejects a particular structure. An incorrect match happens when the ASM erroneously identifies a particular structure.

5.5.8 Procedure to find the best size intervals

This section explains the implemented procedure for evaluating the performance of the ASMs that identify pore structures of a particular sample.

In order to avoid the problems that one might encounter with small-sized pores and large-size pores, a *global search interval* was set to the following values: [**200,60000**]. It is within this global interval that the search of subintervals that maximize the identification will take place.

The above mentioned global interval excludes pores that are too large, in which the overlapping is too complex and their inclusion might lead to unclear results using ASMs. Furthermore, in relation to the lowest interval value, it was taken into account that small size structures are very susceptible to noise and to the image preprocessing steps. Also, it is possible that these small structures are the product of noise or artifacts created by the tomographic machine. Therefore, only structures with more than a certain amount of voxels were included in the search for shape patterns, in this case 200.

After establishing the global search interval, the exploration approach seeks the best pore size intervals, see Algorithm 5.2.

Algorithm 5.2 EXHAUSTIVE SEARCH OF THE BEST PORE SIZE INTERVAL

```
Input: The 3 available pore samples (DAH, ISR and N8). A global search
interval, in this case: pore structures within [200,60000] voxels. This global
interval establishes the boundaries of the subintervals that will be tested.

1: Approximate all pore structures, within the global search interval, to the ellipsoid
   that best fits each one of them, and extract the first 14 landmarks. This is followed
   by the computation of the convex hull of each pore structure in order to calculate its
   convexity, which is the final landmark. From this point forward, each pore structure
   is considered to be a set of 15 landmarks.
2: For all possible subintervals $sub_i$ within the global interval, which are larger
   than 200 voxels in size (To be more specific there are $355228 \times 10^4$ subintervals
   to be analyzed).
     i: Select all pores within the size interval $sub_i$ and make 3 groups, one for
        each sample: $G_{DAH}$, $G_{ISR}$ and $G_{N8}$. If the number of elements of each group
        is greater than 10, continue the evaluation. Else, reject the interval.
    ii: Using the above created groups, create 6 subgroups that further divide
        the elements of each group as follows:
        $G_{DAH} \begin{cases} {}_A G_{DAH} & \text{(40\% of the elements)} \\ {}_B G_{DAH} & \text{(60\% of the elements)} \end{cases}$
        $G_{ISR} \begin{cases} {}_A G_{ISR} & \text{(40\% of the elements)} \\ {}_B G_{ISR} & \text{(60\% of the elements)} \end{cases}$
        $G_{N8} \begin{cases} {}_A G_{N8} & \text{(40\% of the elements)} \\ {}_B G_{N8} & \text{(60\% of the elements)} \end{cases}$
   iii: Generate 3 ASMs ($ASM_{ISR}$, $ASM_{DAH}$, $ASM_{N8}$) using the corresponding
        subgroups: ${}_A G_{DAH}$, ${}_A G_{ISR}$, ${}_A G_{N8}$.
    iv: Evaluate each ASM using all elements within the current interval range,
        see section 5.5.7. The elements that are used to create the ASM that is
        being evaluated are excluded from the evaluation step.

Output: The interval in which the ASMs maximize the identification of pores.
Naturally, one range and one ASM per sample.
```

As mentioned in Algorithm 5.2, every time and interval is analyzed, the elements of each pore sample are subdivided into two groups: A and B. The former is for creating the ASM and the latter is for evaluating its performance. The sketch depicted in Figure 5.26 shows how the subgroups A_i and B_i are created for a hypothetical pore size interval [100,1000] voxels. Additionally, one can see which of the created subsets are used to evaluate the ASMs.

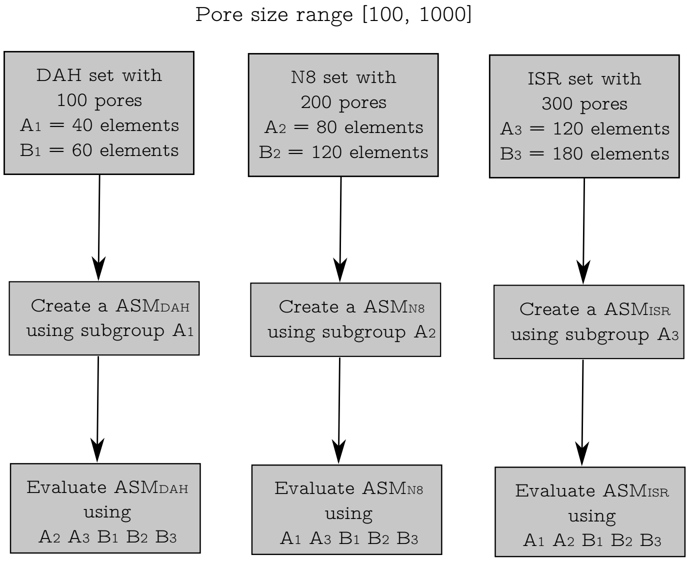

Pore size range [100, 1000]
DAH set with 100 pores A1 = 40 elements B1 = 60 elements
N8 set with 200 pores A2 = 80 elements B2 = 120 elements
ISR set with 300 pores A3 = 120 elements B3 = 180 elements
Create a ASMDAH using subgroup A1
Create a ASMN8 using subgroup A2
Create a ASMISR using subgroup A3
Evaluate ASMDAH using A2 A3 B1 B2 B3
Evaluate ASMN8 using A1 A3 B1 B2 B3
Evaluate ASMISR using A1 A2 B1 B2 B3

Figure 5.26: Graphic representation of the 3D ASM evaluation. The image depicts a pore size range in which the elements of all the samples are going to be used to evaluate their respective shape model.

Clearly, Algorithm 5.2 is a brute-force approach that looks in all pore size intervals until it finds the intervals that maximize the recognition of overlapping structures.

5.6 Results

Table 5.2 displays the total amount of pore structures extracted from each sample using the 3D flood-fill algorithm previously introduced.

Sample	Number of pore regions
DAH	123777
N8	32416
ISR	103759

Table 5.2: Total number of pore regions per sample.

The following table shows the size of the Gaussian kernel for each sample, as well as the gray scale thresholds used to extract the pores from the images. Additionally, this table shows the minimum and maximum size of pore regions in each sample extracted using the corresponding Gaussian kernel and gray scale threshold.

Sample	Kernel size (pixels)	Gray scale threshold	Minimum pore size (voxels)	Maximum pore size (voxels)
DAH	21	$[0-111]$	1	2968792
N8	15	$[0-110]$	1	500880
ISR	21	$[0-120]$	1	3245289

Table 5.3: Size of the Gaussian kernel and gray scale threshold for each sample.

The following image (Figure 5.27) depicts the pore size frequencies within the global search interval. It also shows the frequency plot of all available structures. It should be noted that in order to improve the visualization, the axes were set to a logarithmic scale.

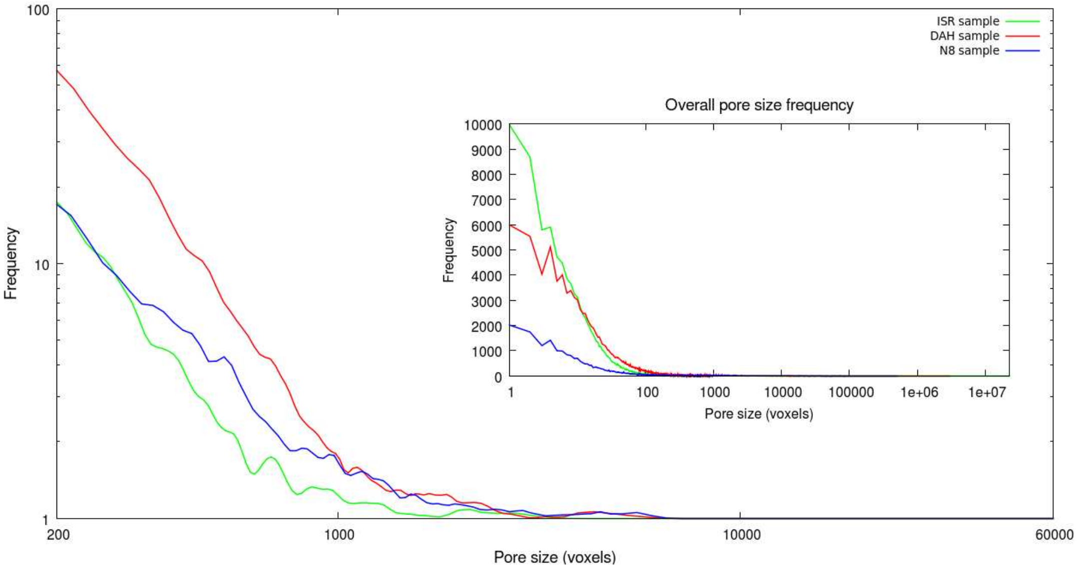


Figure 5.27: Pore size frequencies of the 3 available samples within the general search range [200-60000] voxels.

The following plots show the results of the extensive search procedure described in Algorithm 5.2 (Figures 5.28, 5.29, 5.30). It is possible to see all tested pore size intervals and their corresponding evaluation. Furthermore, the white areas in the plots indicate intervals where the sets did not have more than 10 elements, and therefore were not evaluated.

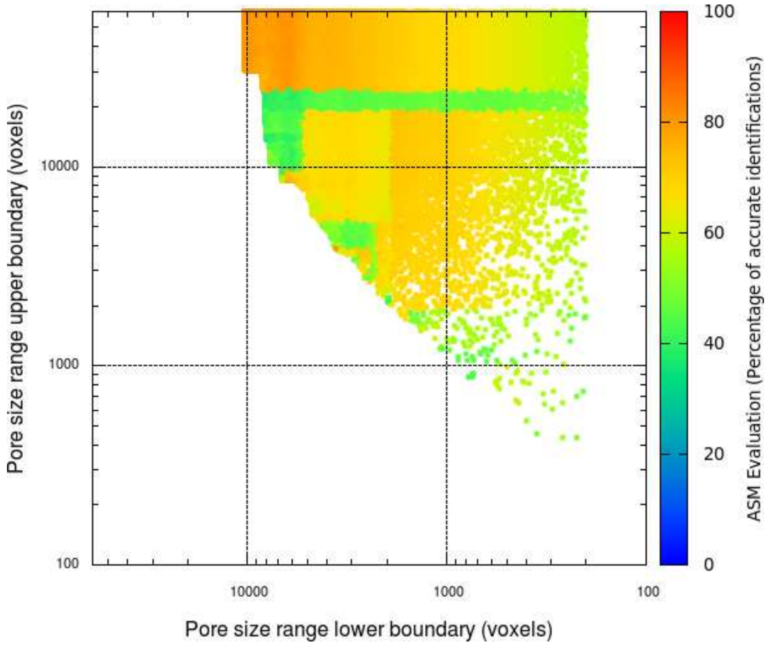


Figure 5.28: Exhaustive pore size interval analysis for the DAH sample.

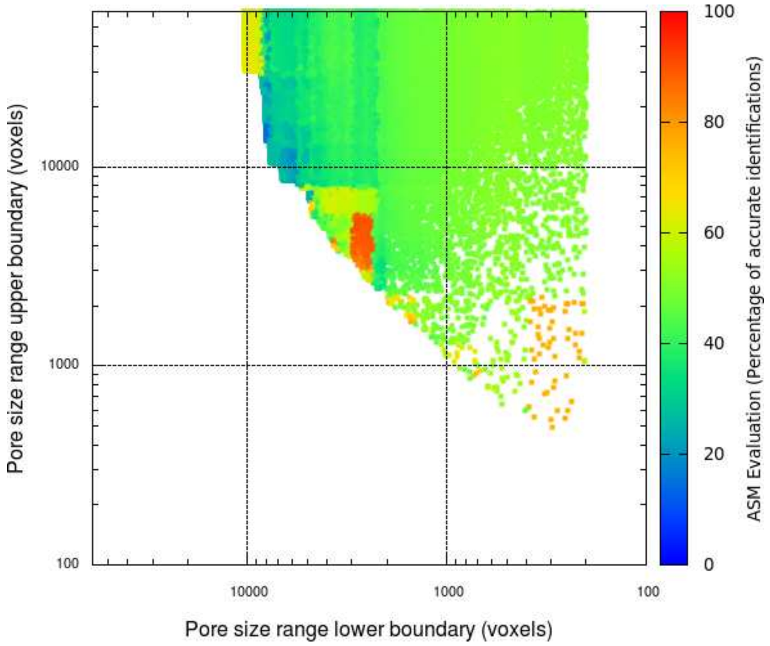


Figure 5.29: Exhaustive pore size interval analysis for the ISR sample.

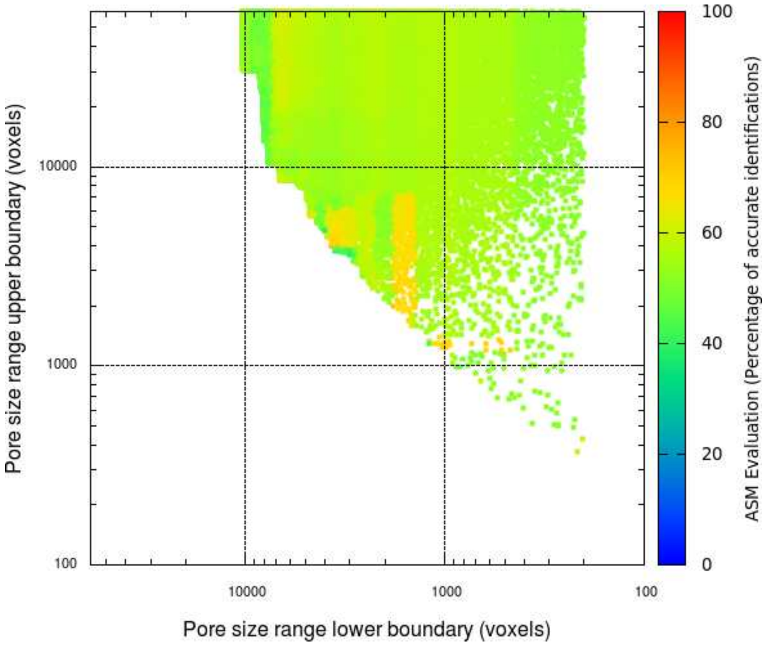


Figure 5.30: Exhaustive pore size interval analysis for the N8 sample.

It is also possible to visualize in all 3 samples that strong identification zones appear in certain interval areas (yellow, orange and red areas). For example, in Figure 5.29 there is a zone with a high percentage of accurate identifications within a relatively narrow range (depicted in red). Also, when the interval ranges become very large (green areas in the upper right hand corner of the plots), the evaluation of the shape models is not satisfactory because the combined complexity of all possible overlapping structures becomes too varied to handle and the ASMs cannot find shape patterns.

If the results of samples ISR and N8 are examined, one can deduce that when the overlapping is too extreme, meaning large pore sizes, it is not possible to create shape models that correctly identify patterns. However, this is not the case for the sample DAH, in which large pores within small ranges can be used for identification (red to orange area in the upper left hand corner of Figure 5.28).

The following table shows the best interval range for each sample: its lower boundary, its upper boundary and its identification effectiveness.

Model	Pore size range [lower boundary, upper boundary]	Identification accuracy (%)
ASM_{DAH}	[6295,37944]	81.27
ASM_{ISR}	[2851,4640]	90.50
ASM_{N8}	[3631 4027]	77.7778

Table 5.4: Best interval ranges to perform ASM identification

From the above table (Table 5.4), it is possible to see that the best identification performance is that of the ASM that belongs to the soil sample ISR, followed by the two rock samples: DAH and N8. The ASM with the largest pore size range is that of the sample DAH with a range that covers 31649 voxels in size. The other two samples have smaller ranges: 1789 for the sample ISR and 396 for the sample N8.

In all 3 samples it was possible to identify the overlapping structures with more than 75 % accuracy, showing that it possible to transport the 2D-ASM technique into the 3D space.

CHAPTER 6

Discussion

This chapter will review several aspects of the methods that are proposed in this work, such as their limitations, pros, cons, applications, etc. Additionally some ideas that might be worth considering for future research are also suggested. Finally, the conclusions of this work will be presented.

6.1 2D ASMs for detecting overlapping plants.

In the case of using 2D ASMs for the detection of overlapping plants, the entire recognition system is composed of 3 main elements: Color segmentation, leaf detection (ellipse detection) and ASM plant recognition. These 3 steps are a part of a chain, where every step influences the performance of the successive step.

Color segmentation ⟶ Leaf detection ⟶ ASM plant recognition

This implies that if one starts with errors in the color segmentation, these inaccuracies will be carried until the end of the calculations, producing poor results. The limitations of each of these elements are discussed below.

6.1.1 Color segmentation

There are many approaches, such as those put forward by Burgos-Artizzu et al. (2011), Gée et al. (2008), Tellaeche et al. (2008), Onyango & Marchant (2003), etc., that attempt to identify different plants by only using color segmentation, which is just one step in the procedure proposed here. These techniques can be executed quickly, identifying the presence of a plant (or plants), but lack the ability to give more information about the analyzed situation, such as the plant shape. To some extent, these approaches perform the identification in a bulky manner, leaving most of the information unanalyzed.

In the current work, the image acquisition was carried out in a chamber with optimal and constant light conditions. This situation facilitated the implementation of a straightforward segmentation process, in which the soil and plants were properly differentiated. Furthermore, one has to take into account that the segmentation used during this investigation only aims to partition the acquired images into soil and plant sections, which were immediately binarized after its detection. As a consequence, this led to loss of information, such as the texture of the plant.

Making use of a segmentation technique that includes all the available information might bring the advantage of a better subpartitioning within the already recognized plant region. This, in turn, might have a direct impact on the performance of the next step in the recognition chain, which is leaf detection. For instance, if one is able to separate leaf regions and stem regions, one would be able to target the leaf detection procedure only to the leaf regions. Also, knowing where the stem regions are provides information about the places where groups of leaves interconnect, i.e. the recognition of individual plants.

Mean shift clustering (Comaniciu & Meer, 2002) is a technique that one might consider to perform further color segmentation within the previously identified plant region. This method is able to find groups of colors without any previous knowledge of the data. However, this comes with a price in terms of execution time, due to the amount of computations required by the method.

Now, what happens if one wants to perform plant identification on an open field situation? Certainly, the color segmentation proposed in this work, which is adapted to fixed lighting conditions, must be modified in order to sufficiently deal with constantly changing illumination conditions. In this situation, open field systems, such as those proposed by Philipp (2004), Tian & Slaughter (1998), Bosch et al. (2007), Zheng et al. (2009), etc., are required.

6.1.2 Leaf detection

The proposed leaf detection method works under the assumption that the leaves have to a certain degree an elliptical shape. Naturally, this conjecture was made because it is true that the model plant (tobacco), as well as many other plants species, have leaves in the form of an ellipse. Naturally, this is not the case for several types of plants, such as those shown in Figure 6.1.

In spite of all the non-elliptical leaves, there are many plants that have leaves where it is still possible to make the above assumption. This approach can then

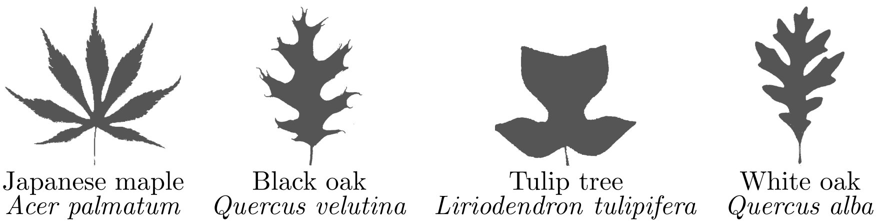


Figure 6.1: Examples of non-elliptical leaves. Images modified from the Pennsylvania Department of conservation and natural resources DCNR (2011).

be combined with other techniques which identify non-elliptical leaf shapes. Also, it is possible to think of techniques that fit several ellipses per leaf in order to approximate their shape.

One of the main advantages of treating leaves as ellipses is the reduction of the complexity of the original leaf shape. This allows the use of several techniques to attempt their detection; moreover, the use of procedures that detect this geometric shape enable its recognition under overlapping situations.

In regard to the methods of ellipse detection, there are several ways in which his task can be carried out. For instance, Yuen et al. (1989), Xu et al. (1990), Aguado & Nixon (1995) and Mclaughlin (1998), proposed similar techniques based on the Hough transform (Duda & Hart, 1972) and the random selection of 3 points along the outline of the shape to generate possible ellipses. Actually, these methods formed the basis of the ellipse detection method proposed in section 4.5.5; the only difference being the use of mean-shift clustering which finds the best fitting ellipses within the cloud of all the possible ones. Instead of the clustering approach, these other techniques use a voting scheme to find the ellipses, in which small ellipses have a very small chance to be detected because larger ellipses have more probability that their points are selected to generate possible ellipses. Mean shift clustering on the other hand will detect the small ellipses that are ignored by the voting scheme.

The disadvantage of both methods - mean shift clustering as well as the voting scheme - is that if one wants to increase the probability that the points of the small ellipses are selected, one has to increase the number of random trials of the method. However, this has the effect that the cloud of possible ellipses increases in size, and more time is therefore needed to analyze it. Another drawback of both approaches is that the selection of three random points produces a different output every time they are executed using the same data; this, of course, is not desirable.

In this thesis, a second approach for detecting ellipses was proposed, see section 4.5.6. This method is based on the detection of arc-sections within the outline of a given region. The identified arc-sections are then combined in order to fit ellipses that best superimpose the region. This detection scheme is faster and more accurate than the one presented in section 4.5.5. Additionally, this method does not have a random step that causes the results to vary from one execution to another, this means that the results will be the same if the approach is executed several times using the same data.

Another important advantage that this method offers is the simplicity in which it locates the arc-sections, which also reduces the computational time of the algorithm. Additionally, one can easily eliminate the small arc-sections by raising the threshold, if it is decided that they are negatively influencing the ellipse detection. Moreover, if the analyzed region contains line sections within its outline, then, one can use the splitting technique polygon approximation, presented in section 3.1.8, to determine which of the found arc-sections are, in fact, lines. Having correct arc-sections is essential before starting the arc combination step; one should remember that in this step the algorithm attempts to merge neighboring arc-sections with the intention of finding the arcs that belong to the same ellipse. The merging step is also quite straightforward: It is essentially a process that only tries to merge arcs that are interconnected by lines that are fully included in the analyzed region. This has the advantage that the algorithm only attempts to merge arcs with high probability of belonging to the same ellipse, thus speeding up the algorithm. Before merging two arcs, there is an ellipse validation step that quantifies the quality of the ellipse produced by the arcs that are being merged. This step prevents the merging of arcs that do not belong to the same ellipse, therefore, maintaining good quality detections.

There are other recognition approaches that use the detection and combination of arcs to find ellipses. For example, Nguyen et al. (2009), Qiao & Ong (2007), Hahn et al. (2008), Ji & Haralick (1999). All these methods, including the one proposed in this work, can be summarized in 3 steps: Arc detection, Arc merging, and Ellipse validation. Of course, each approach considers a different way to carry out each step.

With regard to problems that one might encounter when using the arc detection method, it is worth noting that the ellipse identification scheme was adjusted to best match the shape of the tobacco leaves. However, since these leaves are not totally elliptical, the ellipse detection algorithm can detect extra ellipses where there should be just one. Figure 6.2 shows two examples of the extra detections due to the divergence of the leaves from a perfect elliptical form.

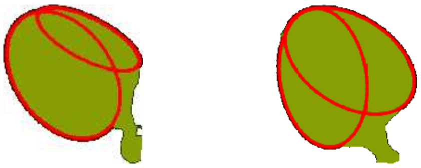


Figure 6.2: Probable extra detections of ellipses.

The fact that additional ellipses might be detected, has a direct impact on the following step of plant recognition. This can cause several problems, such as unnecessary combinations with the faulty ellipse, faulty plant detections, etc. One might think that unnecessary combinations using the faulty ellipses is to some extent a tolerable error, as long as the correct answer is ultimately found. However, in the case of a very time-sensitive plant identification system, these faulty ellipses will affect its performance, increasing the time to carry out the task at hand. The worst case scenario is that the faulty ellipses will lead to the detection of erroneous plants. Certainly, faulty plant detections are an undesirable type of error, because they are detrimental to the overall effectiveness of the plant recognition system.

It is also recommended to take into consideration the application where the ellipse detection method will be used, in order to properly adjust the parameters that control the identification. Essentially, there are two main parameters: Minimum arc size and the area of the ellipse that has to be contained within the analyzed region (see sections 4.5.6 and 4.6.2). These two parameters can be adjusted to allow the identification of objects that deviate substantially from an elliptical form, or the selection of objects which are perfect ellipses. The parameters used in this work are: Minimum arc size = 6 pixels, 97 % for the area of the ellipses contained in the RoI and 20 % for the new information parameter. It should be noted that the selection of these parameters was specially calibrated for the detection of tobacco plantlets, and further fine tunning should be considered for the detection of other type of plants.

6.1.3 ASM Plant recognition

After analyzing an image region in the search for leaves, the final step in the plant recognition system is to cluster the detected ellipses into plants. The proposed methodology for carrying out this task is to make decisions using the shape deformation analyses of different ASMs.

There are other researchers that have already put forward systems that use ASMs to identify plants or parts of plants, such as Søgaard & Heisel (2002), who proposed models for the classification of weed plantlets. However, they mention that the classification experiments were carried out without overlapping plants. In spite of this, the identification models were able to classify with an accuracy between 65 % to 90 %, depending on the weed species. Moeslund et al. (2005) used ASMs to match the shape of cactus leaves before estimating their 3D position. In this case, 86 % of the cactus leaves where identified, nevertheless, this is a semiautomatic system where an initial guess of the position and shape of the leaf has to be manually provided for the system to work. Persson & Åstrand (2008), also used ASMs to classify crops and weeds. In this instance, the identification of plants that are partially occluded is also taken into consideration. The suggested procedure is to create deformable models with shape information taken from the non-overlapped plants as well as overlapped plants, where the overlapped plants are treated as a single shape. To some extent, the main idea is to include as many overlapping situations as possible into the system, in order to enable it to identify partly occluded plants. This, of course, can be done up to a certain limit, though one has to be very careful not to over-train the shape model with overlapped shapes. If it is decided to include overlapping situations in the shape model, one has to be aware that the number of possible overlapping situations are infinite, that the shape model is being trained with erroneous shapes, and that there will be a point where the identification will fail because the variance within the training set is too large. Despite these facts, it is claimed that the classification models were able to match plants with more than 80 % accuracy.

The approach proposed in this work uses ASMs as information holders, which only contain information about true plant shapes. No extra information with overlapping plant shapes was supplied to the models. The ability of detecting partly occluded plants completely relies on the leaf detection step. In order to find the most probable plants, different combinations of ellipses are landmarked and tested with the ASM: the combinations that are not within the constraints of the model are disregarded, while the combinations that are within the shape constraints are kept for further evaluation. This evaluation consists of selecting the combination of ellipses (leaves) that best approximates the mean shape of the ASM, which in turn represents real plant shapes.

Combining ellipses can be computationally demanding and time consuming depending on the total number of detected ellipses. Actually, the problem is that the number of combinations grows exponentially. This is the reason why only a few neighboring ellipses are allowed to be combined when looking for a plant around a selected leaf. One has to take into account that every time a combination of ellipses is performed, the landmarking procedure is also executed, increasing the execution time. Table 6.1 illustrates the number of combinations that the algorithm has to perform with different numbers of neighboring ellipses.

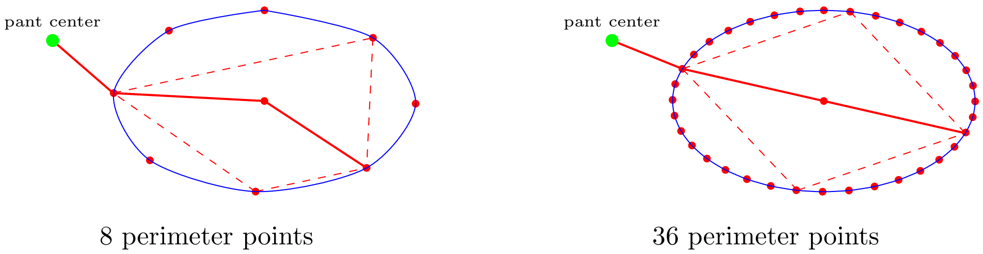
$\binom{n}{k}$	$k = 2$	$k = 3$	$k = 4$
$n = 4$	6	4	1
$n = 8$	28	56	70
$n = 12$	66	220	495
$n = 16$	120	560	1820

Table 6.1: Total number of leaf combinations. Where n is the number of neighboring ellipses and k is the number of leaves in the model.

Naturally, if the n number is set too low, the algorithm might not have the chance of finding the best plant around an area, however, if the n number is set too high, the algorithm will spend too much time combining ellipses. There is, therefore, a trade of between speed and accuracy.

Another factor that influences the algorithm is the number of perimeter points that each ellipse is allowed to have. This directly influences the speed and accuracy. It influences the speed because the more perimeter points each ellipse has, the more calculations the algorithm has to perform in order to find the best landmarks. It influences the accuracy because the more points the each ellipse has, the landmarks will be more representative of the shape of the leaf, and consequently the shape of the plant. Figure 6.3 depicts two examples of this problematic.

It is worth noting that during the experimental phase, the number of neighboring ellipses used in every combination step was 12, and the number of perimeter points used to represent each ellipse was 360 (one perimeter point per arcdegree). It was observed that the first value allows the algorithm to adequately explore the region around a detected leaf. However, the second value was extremely high, which negatively affected the speed of the plant identification algorithm.



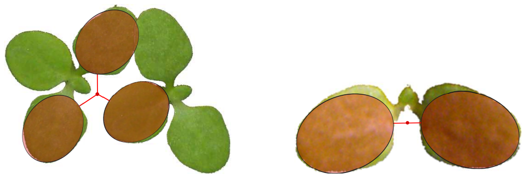
8 perimeter points

36 perimeter points

Figure 6.3: Problems that might occur with the selected number of perimeter points. Too few perimeter points might lead to erroneous landmarks, while too many will unnecessarily increase the number of calculations for every combination.

One more aspect which must be analyzed is the total number of landmarks used to represent a plant. In this case, the proposed technique uses only 5 landmarks for each plant leaf. This has the positive effect that the computational load is greatly reduced. One should remember that every time there is a combination of ellipses, the ASM has to analyze the viability of the shape. If there is a small number of landmarks, the ASM will rapidly perform its task.

Additionally, it is worth mentioning that the developed ASMs make minimum use of the information that the RoI provides, i.e. only to test if the plant center is within the RoI. Making certain that the plant center is within the RoI improves the analysis of complicated situations, where several plants are present in the image. If this simple test is not performed, the shape model might group together a set of ellipses as a viable combination of leaves, which in actuality do not belong to the same plant (see Figure 6.4 case 1). On the other hand, a drawback of checking whether the plant center is within the RoI becomes apparent when the shape of the plants, specially the ones with 2 leaves, have bent leaves that place the calculated center outside of the RoI (see Figure 6.4 case 2).



Case 1 Case 2

Figure 6.4: Example cases of erroneous detection of the plant and plant center. Case 1 shows an image where the center of the plant is outside of the RoI due to a incorrect combination of ellipses. Case 2 shows a case where the computed center of the plant is place out of the RoI due to the shape of the plant.

Of course, using the information that the analyzed region provides brings more advantages than disadvantages. However, in this research only one pixel is used to help the ASM identification system. One way to improve the recognition algorithm would be to connect the landmarks that form the skeleton of the plant. This might be easily carried out by joining the landmark points using the A* algorithm (Russell & Norvig, 2003, pp. 97-101) to find the shortest path between them. Certainly, the computed paths would have to fulfill certain characteristics that depend on the plant shape. Another way to improve the identification process would be to use the appearance (i.e. texture of the plant leaf) around the landmark points in order to check that every landmark is correctly placed. The texture information can be incorporated using Active Appearance Models (Cootes & Taylor, 2004, AMMs), a technique which is very close related to the ASMs.

Every time the algorithm is analyzing a certain area of the RoI, each of the developed shape models is executed with the aim of finding a plant with the appropriate shape around that area. If more than one ASM provides a possible match, then the algorithm analyzes and compares their deformation energy in order to decide which one provides is the best shape. The analysis is carried out by a function that takes into account the deformation energy and also the number of leaves. This function favors the plants with the most leaves because their models need more deformation energy to match a plant. In a way, the suggested function attempts to balance the competition between the ASMs.

It is possible to integrate further ASMs in order to recognize different types of plants. However, this implies that the amount of calculations required to analyze all the possibilities increase with every new ASM; if there is just one specific target plant with an specific shape, only one ASM can be used, thus speeding up the recognition process.

6.1.4 Possible applications

An identification system that is able to recognize shapes under overlapping situations can be applied in different areas for the automatization of tasks, especially those that entail repetitive and monotonous actions.

The following lists attempts to provide a general idea of the various application in which this system can be employed:

1. Sorting and grading agricultural produce. Systems like the ones proposed by Marchant et al. (1990) to sort potatoes, Deegan et al. (2010) to sort cucumbers and tomatoes, and Patel et al. (1998) to sort eggs, could benefit using this technique. Moreover, elliptically-shaped agricultural produce (i.e. oranges, limes, melons, chillies, aubergines, guavas) can be easily incorporated to the system.

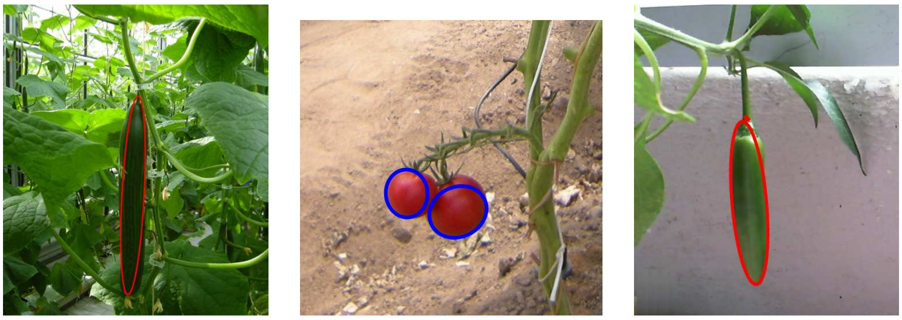
2. Selective plant breeding. In this case, automatic systems that have to identify plantlets with specific visual characteristics can be developed. Plants without these key characteristics can then be automatically culled from the plant stock.

3. Harvesting robots. In this situation, the robots require the ability to identify the crops under overlapping situations, which most of the time are caused by leaves or stems. For example: the cucumber harvesting robot developed by Henten et al. (2003), the mushroom picker developed by Reed et al. (2001), the strawberry harvesting robot developed by Hayashi et al. (2010), the cherry harvesting robot developed by Tanigaki et al. (2008), the tomato picker developed by Takahashi et al. (2001), and the grape harvesting robot (still in development) by Berenstein et al. (2010). Note that all the aforementioned crops have an elliptical shape, therefore, it would be straightforward to generate the corresponding ASMs.

4. Weed control. The aim of weed detection, automatic or not automatic, is to enable the possibility of eliminating any plant that is not desired in a crop field. The earlier the weeds are detected, the more effectively it is possible to eliminate them. Many methods have being already put forward, i.e., Schuster et al. (2007), Hemming (2000), Åstrandand & Baerveldt (2002), Jeon & Tian (2009). All of these approaches require a system that effectively identify weed plantlets for further elimination. There are three main techniques to fight weeds after their detection: mechanical removal, precision spraying and the creation of weed density maps that are used to know where to spray herbicides. Additionally, there is one moreover technique (still under development by Wöltjen et al. (2008)) that uses lasers to eliminate weeds.

After mentioning some of the possible applications, there is still one interesting point to address: Is it feasible to apply this recognition system in real life situations? In order to answer this question, one has to take into account two aspects: speed and accuracy. It is believed that the recognition system is fast enough for real applications. This conclusion is based on the observed performance at the experimental phase. If more speed in required, the algorithms could be easily implemented to take advantage of the parallel computing that the new Graphic Processing Units offer (Nickolls et al., 2008, GPUs). In this manner, the method would be accelerated to result in faster detections. In regard to the accuracy of the algorithm, it is also believed that it is accurate enough to work in actual situations. One should also consider that in a real application, the systems have more than one perspective view to analyze, thus, the probability of finding the desired objects increases.

Finally, in order to complement the idea of possible applications, the following images show the detection of cucumbers, tomatoes and serrano pepper using the ellipsoid detection method proposed in this work.



Cucumber Tomato Serrano pepper

Figure 6.5: Example of possible applications using ellipse detection.

Please note that in order to obtain the above detections, the parameters values of the ellipse detection algorithm were changed to best fit the shape of the object at hand. Also, the segmentation process was achieved using different of color filters (e.g. red for the tomatoes, etc) that were able to extract the regions that belong to the object.

6.2 3D ASMs for overlapping problems

Active Shape Models can be used not only in the 2D but also in higher dimensional spaces. The employed methodology, which was used to identify plantlets under overlapping conditions, was transfered to the 3-dimensional space with the goal of demonstrating that ASMs can also be used to identify characteristics of complex overlapping structures. One of the main differences is that instead of extracting data from a single plane (image), a 3-dimensional structure is reconstructed from several stacked planes (tomograms). Again, a shape simplification took place, allowing the use of landmarks to represent otherwise very complex and disparate objects.

6.2.1 Image preprocessing

In this investigation, as in the previous one, the preprocessing of the images plays a very important role in the methodology, because it greatly influences the final result. It was mentioned before that due to the physical properties of the synchrotron light source, the images were subjected to a preprocessing step using a 2D Gaussian filter to eliminate the noise. Moreover, based on visual inspection of several gray histogram analyses, after applying the Gaussian filter, the grey scale intervals that correspond to the minerals and voids were selected. Nevertheless, the are some aspects that would be worth taking into account in future research.

The way of dealing with the ring artifacts in each tomogram was quiet drastic because part of the information contained in each tomogram was disregarded. Certainly, keeping only the information that is within these rings eliminates their effect in the calculations, however, a filter like the one proposed by Münch et al. (2009) would eliminate the ring artifacts and enable the use of all the available information area.

After the exclusion of the faulty information caused by the ring artifacts, a 2D Gaussian filter could be used to further smooth the images. Then, following the steps of the proposed methodology, the gray histogram analyses would take place.

An interesting remark about the suggested technique is that the Gaussian smoothing is applied in the 2-dimensional space, thus, only one tomogram can be affected by this filter at a time; the use of a 3-dimensional filter would be a better option. This would mean that the filter would be able to affect several layers (i.e. several tomograms) at once taking into account neighboring voxels in all directions. This has a direct effect on the 3D reconstruction because small pore regions, which most likely would disappear with a 2D filter, might have the chance of not being removed if the effect of all its neighbors is accounted for. If small pore regions are considered in the reconstruction, it can occur that these regions are connecting sections between larger structures.

One more annotation to the proposed approach is that the threshold intervals are based on the visual inspection of a few of the gray scale histogram analyses. Again, this methodology only takes into account 2-dimensional data. Naturally, it is better to consider all the tomograms in the data set to perform the localization of the thresholds that identify the pores and other minerals. An idea worthy of exploration is to integrate a clustering system that uses all the gray scale values contained in all the tomograms in order to identify regions with specific color ranges. For instance, the mean shift clustering technique, proposed by Comaniciu & Meer (2002), is an viable option to perform the 3D color space analysis.

6.2.2 3D ASM identification

The main objective of this part of the research was to transport the methodology used in 2-dimensions into the 3-dimensional space. However, in this case the identification was carried out using complex overlapping structures. Naturally, the proposed technique also uses the reduction of the original shape into a geometrical object with the purpose of reducing the complexity of the problem and also to standardize the shapes of all structures. In this case, the geometrical objects that were used are ellipsoids, this has the advantage that a small number of landmarks can be used to represent the shape.

The selection of ellipsoids as the geometrical shape model was carried out by comparing how well different geometrical shapes (lines, cuboids and ellipsoids) represent different overlapping structures. In this case ellipsoids were selected because they had the smallest error when fitting the original form into a simplified geometrical object.

Without the assumption that overlapping structures were ellipses, it would be much more difficult to find shape patterns within the collected data. This is because the variability of the amorphous structures was too great for the successful application of the landmarking techniques. Moreover, the fact that only 15 landmarks were used to represent a pore made it possible to use an exhaustive search approach that was constantly generating shape models within different pore size intervals. This exhaustive search technique made it possible to identify a several intervals where it is possible to use ASMs to model specific shape characteristics.

When performing the search within all the pore size intervals, it was necessary to divide all the elements of every tested interval into groups: one to train the model, and one to test it. The search algorithm was carried out in this manner because there is only one tomographic data set per rock and soil sample. Of course, the bisection would not be necessary if two or more tomographic sets per sample were available.

With regard to the results of the detection, it is possible to see that there are specific intervals zones that allow accurate identification of overlapping structures (Figures 5.28, 5.29, 5.30). The best interval was found when evaluating the soil sample ISR, with over 90 % identification effectiveness. The other two samples (N8 and DAH) also have intervals in which the shape model was able to find particular patterns that allow their characterization, with an effectiveness of 81.27 % and 77.77 % respectively.

Just like in the 2D study, when the amount overlapping becomes too great, there is a detrimental effect in the ability of the algorithm to properly identify the structures. The results also show that when the overlapping structures become to large, they become too complex to be represented with a shape model, supporting the decision to exclude such large forms from the global pattern search.

With this work, it was shown that the proposed ASM methodology used in the 2-

dimensional space is transferable to the 3-dimensional space, including the shape simplification -ellipses in 2D and ellipsoids in 3D- which greatly simplifies the landmarking process.

The presented technique has been successfully implemented to investigate the role of fungal mycelium in the formation of carbonate concretions in growing media (Masaphy et al., 2009).

For future research, it is contemplated to include the preferred direction of each pore structure into the ASMs. It is believed that if this is taken into account, the typification of pore structures will be more accurate. In order to incorporate this feature, the eigendecomposition of each structure has to be computed, where the eigenvector that corresponds to the greatest eigenvalue represents the preferred orientation; see section 3.3.4 for more information about eigenvectors and eigenvalues.

6.2.3 Possible applications

The applications of this research are broad, however, in this particular case, it has more relevance for basic soil science studies. Finding and analyzing porous regions in mineral and rock samples can help to understand the chemical reactions caused by weathering and leaching that ultimately lead to the disintegration and release of elements bound in the rock. These elements might be incorporated back into the soils, giving added nutrients to the plant life or eroded into the rivers and oceans, something that is important for the global element cycles.

There are two main questions that encourage the analysis of pore structures and their geometry:

1. The first question that needs to be answered is if there are connected pores within the mineral or rock body.
2. If so, then the geometry and size of these pore can effect the interaction between the fluids and minerals. In the case of small pores, diffusion of fluids is very slow, hindering element turnover. Larger pores allow a higher flow of fluids to enter and leave the rock body, speeding up the reacting rate and weathering of the minerals.

Understanding these processes is important for several areas of study, i.e., plant nutrition, plant irrigation, soil leaching, where new models can be developed or the information can be used to expand on existing theories.

In the area of plant science, there are also possible applications. For example, one could model inter-cellular spaces or chambers with the aim of analyzing their role in gas and fluid exchange within the plant.

CHAPTER 7

Conclusions

The main objectives of this work were to transport the use of Active Shape Models into the world of plant science to solve complex overlapping situations. This was carried out with the purpose of developing methods that are able to identify shapes under difficult and fluctuating conditions, which is normally the case when working with plants or their produce.

This work presents two approaches that use the capabilities of ASMs to deform shapes: one that identifies plantlets under overlapping situations (2D), and a second method that identifies overlapping amorphous pore structures from soil and rock samples (3D).

In the 2D space, the identification system is able to correctly recognize plantlets under overlapping situations. It also demonstrated that approximating the shape of leaves as ellipses positively influenced the detection of plants that are growing in overlapping conditions: firstly the reduction of a complex shape into a basic geometric form simplifies the form recognition process; secondly using regular geometric forms allows the automatic reconstruction of missing data, such as in the case of overlapping leaves. Additionally, the presented model was evaluated using plants with different overlapping circumstances: from cases where there is no overlapping (0 % overlapping area), to cases where there were severely occluded plants (32 % overlapping area). Certainly, the performance of the algorithm decreases when the overlapping area between the plants increases, from almost 100 % efficiency to ca. 21 % efficiency. This, of course, is caused by the increasing amount of missing information and by the complications caused by the number of leaf combinations that the system has to evaluate. Regardless of the overlapping situations, the system is still able to recognize plants as long as enough information about the individual leaves was still visible.

A similar methodology was then transported into the 3D space, where the proposed identification system was also able to identify complex overlapping structures. The pore spaces in three different soil and rock samples were analyzed in order to test the feasibility of using ASMs to solve overlapping situations in the 3-dimensional space. These complex structures were first approximated into a simpler geometric shape, in this case an ellipsoid. This was necessary because the original shape variability was too great for the successful use of landmarking techniques. Using ellipsoids to represent the original shape of porous structures facilitated their recognition, even if they were of different sizes (i.e. contained a different number of voxels). The results indicate that certain pore size intervals exist for each sample where the system was able to find specific shape patterns that can then be used to identify distinct pore structure groups within each sample set.

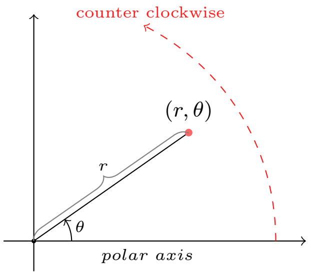
In all 3 samples it was possible to identify the overlapping structures with more than 75 % accuracy. For sample ISR, an interval range was found were over 90 % of the pores could be accurately identified, while sample DAH contained an interval range where the identification effectiveness was above 80 %, and sample N8 included an interval range where the effectiveness was above 77 %.

The techniques proposed in this work show that modeling the original form with geometrical shapes, in this case ellipses and ellipsoids, is an excellent way of reducing the complexity of identification problems. Of course, the combination of this scheme with ASMs provide a robust pattern recognition tool that can be applied to many areas where computer vision techniques are required. Such computer vision systems will play a key role in the exponentially increasing amount of automatization within agriculture and food production, where the required systems have to deal with the ever changing conditions in the natural environment.

APPENDIX A

Mathematical background

A.1 Polar Coordinates



The polar coordinate system uses a tuple of two variables (r, θ) to represent its coordinates. The term r in the tuple is the distance to a reference pole (this term is called radial coordinate or radial distance). The term θ in the tuple is the angle between the polar axis and the line segment connecting the pole and the point (this term is called angular coordinate). The pole is the reference coordinate that all other points use to measure its radial distance. The polar axis is the reference axis. Generally, the polar axis is taken to be the positive Cartesian x-axis, in other words, it is the horizontal line to the right of the pole. In order to have unique coordinate points, the values of the radial coordinates are limited to $r > 0$, and the values of the angular coordinates are within the interval $[0, 2\pi)$. By convention θ is measured in a counter clockwise direction.

A point (x, y) in Cartesian coordinates can be represented in polar coordinates (r, θ), and vice versa, using the following equations:

$$\begin{aligned} r &= \sqrt{x^2 + y^2} & \qquad x &= r\cos(\theta) \\ \theta &= \arctan\left(\tfrac{y}{x}\right) & \qquad y &= r\sin(\theta) \end{aligned}$$

A.2 Arctangent function

It is worth noting that the calculation of the angular values was carried out with a modified **atan2**(y, x) function[1], which ensures the interval $[0, 2\pi)$ of the angular coordinates.

---

[1] The atan2 is a function that is available for the C++ programming language. Its documentation can be found online at (Cplusplus.com, 2010; LinuxManPages, 2010; Wikipedia, 2010a).

Modified version of the atan2(y, x) function presented in Cplusplus.com (2010); LinuxManPages (2010); Wikipedia (2010a):

$$\theta = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0 \text{ and } y \geq 0 \\ \arctan\left(\frac{y}{x}\right) + 2\pi & \text{if } x > 0 \text{ and } y < 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \\ \frac{1}{2}\pi & \text{if } x = 0 \text{ and } y > 0 \\ \frac{3}{2}\pi & \text{if } x = 0 \text{ and } y < 0 \\ 0 & \text{if } x = 0 \text{ and } y = 0 \end{cases}$$

A.3 Metric schemes

There are some popular metric schemes to calculate the distance between two points p_1, p_2, where each point has an x-cartesian coordiante and and y-cartesian cordinate $p = (x, y)$.

Chessboard distance	$\max(\lvert x_2 - x_1\rvert, \lvert y_2 - y_1\rvert)$
Manhattan distance	$\lvert x_2 - x_1\rvert + \lvert y_2 - y_1\rvert$
Euclidean distance	$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

A.4 Centroid of a point set

The centroid is the average position over all the points included in the set.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

A.5 Rotations in $\mathbb{R}^3$

"A vector $\mathbf{u} \in \Re^3$ can be rotated counterclockwise through an angle θ around a coordinate axis by means of a multiplication $\mathbf{P} \cdot \mathbf{u}$ in which $\mathbf{P}$ is an appropriate orthogonal matrix" (Meyer, 2000, p. 328).

Rotation around the x-Axis $$\mathbf{P}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}$$

Rotation around the y-Axis $$\mathbf{P}_y = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}$$

Rotation around the z-Axis $$\mathbf{P}_z = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

A.6 Singular Value Decomposition (SVD)

Stewart (1993) mentions that “the contributions of five mathematicians Eugenio Beltrami (1835-1899), Camille Jordan (1838-1921), James Joseph Sylvester (1814-1897), Erhard Schmidt (1876-1959), and Hermann Weyl (1885-1955) who were responsible for establishing the existence of the singular value decomposition and developing its theory.”

“The SVD is intimately related to the familiar theory of diagonalizing a symmetric matrix. Recall that if $\mathbf{A}$ is a symmetric real $n \times n$ matrix, there is an orthogonal matrix $\mathbf{V}$ and a diagonal $\mathbf{D}$ such that $\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^{\mathsf{T}}$. Here the columns of $\mathbf{V}$ are eigenvectors for $\mathbf{A}$ and form an orthonormal basis for $\mathbb{R}^n$; diagonal entries of $\mathbf{D}$ the are the eigenvalues of $\mathbf{A}$. To emphasize the connection with the SVD, we will refer to $\mathbf{V}\mathbf{D}\mathbf{V}^{\mathsf{T}}$ as the *eigenvalue decomposition* for $\mathbf{A}$.” Kalman (1996, p. 2)

However, $\mathbf{A}$ does not need to be a square matrix; bellow there is a more general definition of the method:

“For each $\mathbf{A} \in \mathbb{R}^{m\times n}$ of rank r, there are orthogonal matrices $\mathbf{U}_{m\times m}$, $\mathbf{V}_{n\times n}$ and a diagonal matrix $\mathbf{D}_{r\times r} = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_r)$ such that

$$\mathbf{A} = \mathbf{U}\left(\mathbf{D}\right)_{m\times n}\mathbf{V}^{\mathsf{T}} \quad \text{with } \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > 0$$

The λ_i are called the nonzero singular values. The above factorization is called a singular value decomposition of $\mathbf{A}$, and the columns in $\mathbf{U}$ and $\mathbf{V}$ are called left-hand and right-hand singular vectors for $\mathbf{A}$, respectively.” (Meyer, 2000, p. 412)

“The first eigenvector of the matrix $\mathbf{A}$ points in the direction of maximum variation, and the corresponding eigenvalue measures the variation in this direction; i.e., it is the variance in this direction. The subsequent eigenvectors point in the directions of maximum variation orthogonal to the previous directions, and the eigenvalues again measure the variations.” (Muller et al., 2004, p. 524)

Most of the computational packages arrange the values of the eigenvalues and aigenvectors according to the variance they represent, such that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$$

where n is the number of eigenvalues and eigenvectors of matrix **A**. This arrangement facilitates dimensionality reduction if necessary.

"The directions of maximum variation are then given by the columns of **U** and the variances (eigenvalues λ_j of **A**) λ_j by the squares of the singular values, i.e., $\lambda_j = \sigma_j$. In probability theory the singular values are known as the *standard deviations*." (Muller et al., 2004, p. 524)

A.7 Complex numbers

Complex numbers describe the sum of a real number and an imaginary one

$$z = x + iy$$

where x is the real part of the complex number and y is the imaginary part.

"Every complex number can be represented as a point in the complex plane, and can therefore be expressed by specifying either the point's Cartesian coordinates (called rectangular or Cartesian form) or the point's polar coordinates (called polar form). The complex number z can be represented in rectangular form as" (Wikipedia, 2010b).

A.8 Euler's formula

The Euler formula states,

$$e^{i\theta} = \cos\theta + i\sin\theta,$$

where e is the euler's number, i is the imaginary unit and θ is an argument in radians.

A.9 Fourier Transform

For a continuous function $f(x)$, the Fourier Transform is defined as

$$F(u) = \int_{-\infty}^{\infty} e^{-i2\pi ux} f(x)dx,$$

the corresponding Inverse Fourier Transform

$$f(x) = \int_{-\infty}^{\infty} e^{i2\pi ux} F(u) du,$$

where $i = \sqrt{-1}$, x is a real variable and u is the frequency variable. Furthermore, if one uses the Euler's formula (see Appendix A.8) to substitute the term $e^{i2\pi ux}$, the trigonometric functions appear in the integrals. This is why it is said that the input function is transformed into a series of sines and cosines.

A.10 Gaussian smoothing of images

The following equation shows a Gaussian function in 2-dimensions:

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

"where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution" (from Wikipedia (2010c)). Figure A.1 shows an example of a 11×11 pixel Gaussian kernel. This kernel can be convoluted with every image pixel and its neighbors. The farther away a pixel is, the less likely it will effect the evaluated pixel, which is at the central point of the kernel. Moreover, the size of the kernel (controlled by σ) dictates how smooth the output image will be.

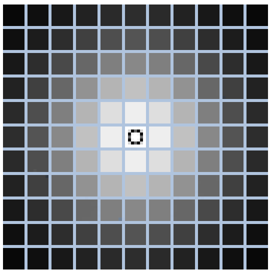


Figure A.1: Gaussian kernel example.

SYMBOLS AND ABBREVIATIONS

$\mathbf{A}$, $\mathbf{B}$, ..., $\mathbf{Z}$	bold-face, capital letters refer to matrices
A, B, C, D, E, F	constant terms of the ellipse equation
A, ..., G, H, I, J	constant terms of the ellipsoid equation
$\mathbf{a}$, $\mathbf{b}$, ... , $\mathbf{z}$	bold-face, small letters refer to vectors
a, b, c	ellipse and ellipsoid radii (semimajor and semiminor axes)
a_{id}	accurate identifications
arc	structure containing the points of an arc
arctan	inverse function of the trigonometric tangent function
AMM, AMMs	Active Appearance Model(s)
ASM, ASMs	Active Shape Model(s)
$\mathbf{C}$	covariance matrix
CT	X-ray Computed Tomography
cp	closest point to plant center
$\tilde{c}$	focal length
${}_nC_k$	combination of n ellipses using groups of size k
°C	degree Celsius
d	distance or dimension counter
DAH	rock sample
de	deformation energy
$\mathbf{E}$	ellipse
e	Euler's number
ec	ellipse center point
$\hat{e}$	ellipse eccentricity
$\tilde{f}$	focus point
$F(u)$	Fourier transform of $f(t)$
FD, FDs	Fourier descriptor(s)
f_e	fitness function or evaluation fucntion
$f(t)$	function in the time domain
fp	furthest point to plant center
GA	Genetic Algorithm
g	generation counter
$\mathtt{H}$	Image height
Hd	Hausdorff distance
hd	one-sided Hausdorff distance
ɦɱ	Hu moment
I	Image
ɨ	intersection point
i, j, k, n, m	general indices and counters, $i = \sqrt{-1}$ imaginary number
$K(u)$	kernel function
kde	kernel density estimator

L	line
Ł, ł	latus rectum, semilatus rectum
ɱ	moment
mad	mean absolute distance
$\check{m}$	middle point
max	maximum value
min	minimum value
$m_s(\mathbf{u})$	mean shift vector
MRI	Magnetic Resonance Imaging
$\mathcal{N}$	set containing k real arc neighbors
N8	rock sample
nv	total number of voxels
P Population	
p	Cartesian point: 2D (x, y) and 3D (x, y, z)
$\mathbf{p}$	RGB pixel
pc	plant center
p_B	pixel blue channel
p_G	pixel green channel
p_R	pixel red channel
p_G^*	pixel's computed green value
PCA	Principal Component Analysis
PET	Positron Emission Tomography
PSI	Paul Scherrer Institute
R	Region
$\mathbb{R}$	real numbers
r	radius or radial distance
RGB	Reg Green Blue color space
RoI	Region of Interest
$\mathcal{S}$	set of shapes
$\bar{\mathbf{s}}$	mean shape
$\mathbf{s}$	shape or set of points
sin, cos, tan	trigonometric functions
SLS	Swiss Light Source
sub	subinterval
SVD	Singular Value Decomposition
T	tangent line
$\intercal$	transpose of a matrix
thr	threshold
TOMCAT	TOmographic Microscopy and Coherent rAdiology experimenTs
V_K	desired percentage of variance
V_T	total variace
v	percentages of total variance
$\mathtt{W}$,	Image width
wa, wb	breadth of the leaf landmark
w_f	weighting factor
w_{id}	wrong identifications
ϖ_s	window size
x_c, y_c, z_c	center position coordinates

$\bar{x}$, $\bar{y}$, $\bar{z}$	mean values
x	x-axis; coordinate along the x-axis; an algebraic variable
y	y-axis; coordinate along the y-axis; an algebraic variable
z	z-axis; coordinate along the z-axis; an algebraic variable
θ , ω, β	angular values
α	wave-operator values
σ	standard deviation
μ	mean value
$\boldsymbol{\mu}$	vector of means
λ,	eigenvalue
$\boldsymbol{\lambda}$	eigenvalues
$\boldsymbol{\phi}$	eigenvector
$\boldsymbol{\Phi}$	eigenvectors
δ	element of a matrix
π	mathematical constant Pi
Δ	delta changes of a variable
Λ	Amplitude
ξ	roundness factor

LIST OF FIGURES

3.1 Object invariant transformations. 6

3.2 Shape block representation. 7

3.3 Example of geometric descriptors. 8

3.4 Distance transform. 9

3.5 Skeleton example. 10

3.6 (r, θ) Signature of a circle. 12

3.7 (r, θ) Signature of a leaf. 13

3.8 Chain code system: 4 and 8 direction code. 14

3.9 Example: Leaf chain code. 14

3.10 Polygon approximation: merging technique 15

3.11 Erroneous polygon approximation: merging technique 16

3.12 Polygon approximation: splitting technique 17

3.13 Erroneous polygon approximation: splitting technique 17

3.14 Example of shape of reconstruction using Fourier descriptors. . . . 20

3.15 Conic section: ellipse 21

3.16 Labelled diagram of an ellipse. 21

3.17 Latus rectum and semilatus rectum of an ellipse. 25

3.18 Fitting an ellipse to a given set of points. 26

3.19 The Ellipsoid 27

3.20 Landmarks placed on a leaf shape. 30

3.21 Labeling landmarks to establish correspondence. 31

3.22 ASM training set. 31

3.23 Example of a consistent landmarking scheme. 31

3.24 Rotaion step. 33

3.25 Example of aligned landkmarks and mean shape. 34

3.26 Sweetgum leaf model deformation. 38
3.27 Sweetgum leaf model wrong deformations. 39

4.1 Simple overlapping example. 41
4.2 Examples of overlapping situations in Biosystems 43
4.3 Model plant 46
4.4 Seeding the tobacco plantlets. 47
4.5 Images presented to the participants before they started solving the identification cases. 48
4.6 Example image showing the type of cases presented to the participants in the comparison experiment. 48
4.7 Image acquisition. 49
4.8 Effect of the color segmentation approach on two sample images. . 50
4.9 Border extraction. 51
4.10 Example of how to generate possible ellipses with 3 randomly selected border points 51
4.11 Tangent line approximation 52
4.12 Computing the center of an ellipse using 3 points. 53
4.13 Possible ellipses generated with the 3 random points method. . . . 54
4.14 Example of overlapping ellipses. 57
4.15 Basic arc-combination. 58
4.16 Behavior of the (r, θ) signature using an ellipse and a circle. . . . 59
4.17 Behavior of α values after applying Equation 4.5. 59
4.18 Ellipse arc-extraction using ascending descending analysis of the α_i values. 60
4.19 Circle arc-extraction using ascending descending analysis of the α_i values. 60
4.20 Arc extraction on a real plant contour. 62
4.21 Middle points of arc sequences. 62
4.22 Identifiying *real arc neighbors* drawing lines on the RoI. 63
4.23 Representing the leaves of a plantlet with ellipses. 66
4.24 Extracting landmarks step 1. 66
4.25 Extracting landmarks step 2. 67

4.26 Extracting landmarks step 3. 67
4.27 Extracting landmarks step 4. 67
4.28 Examples of fully landmarked plants. 68
4.29 Deformable template for tobacco plants with 2 leaves. 69
4.30 Deformable template for tobacco plants with 3 leaves. 69
4.31 Deformable template for tobacco plants with 4 leaves. 70
4.32 Plot of eigenvalues: 2 leaves model. 70
4.33 Plot of eigenvalues: 3 leaves model. 71
4.34 Plot of eigenvalues: 4 leaves model. 71
4.35 Example of real plant detection. 72
4.36 Locating neighboring ellipses. 73
4.37 Placing calculated plant center in a 3 × 3 square 74
4.38 Images used to compare the ellipse detection methods. 76
4.39 Output of the Arc ellipse detection using ellipses as input. 78
4.40 Output of the Arc ellipse detection using plants as input. 78
4.41 Cases used to evaluate the Arc ellipse detection. 79
4.42 Case a: analysis of a contour that contains 1309 pixels. 80
4.43 Case b: analysis of a contour that contains 1692 pixels. 81
4.44 Case c: analysis of a contour that contains 1556 pixels. 81
4.45 Case a: analysis of a contour that contains 1309 pixels. 82
4.46 Case b: analysis of a contour that contains 1692 pixels. 83
4.47 Case c: analysis of a contour that contains 1556 pixels. 83
4.48 Leaf identification on *Galium aparine* 84
4.49 Leaf identification on *Amaranthus retroflexus* 85
4.50 Leaf identification on *Chenopodium album* 85
4.51 Leaf identification on *Senecio vulgaris* 85
4.52 Leaf identification on *Beta vulgaris* 85
4.53 Leaf identification using *Apera spica-venti* 85
4.54 Overlapping area examples. 88
4.55 Results after evaluating the ASM recognition system. 90

4.56 Results after evaluating human recognition of plants. 92

4.57 Example of a series of images showing trays with real plants over time. 93

4.58 *Nicotiana tabacum* detection example 94

4.59 Real cases of *Nicotiana tabacum* plant recognition. 94

5.1 Projecting a 3D object onto the 2D space. 97

5.2 Empty space encapsulated in a 3D structure. 98

5.3 Example of overlapping structures in the 3-dimensional space. . . 99

5.4 3D Active Shape Model of a human liver. 100

5.5 3D Active Shape Model of a human face. 100

5.6 3D Active Shape Models of Arabidopsis leaves. 101

5.7 Sample prepared for tomographic analysis. 104

5.8 SLS - Swiss Light Source at the Paul Scherrer Institute, Villigen, Switzerland. 105

5.9 TOMCAT beamline. 105

5.10 Example layers used in the 3D reconstruction. 106

5.11 Example of a stack of tomograms. 107

5.12 Close-up on the effect of Gaussian smoothing on a sample image. 107

5.13 Effects of the Gaussian filter on a tomographic image. 108

5.14 Histrogram plots that belong to 3 individual layers of the samples: ISR, N8 and DAH. 109

5.15 Histrogram analysis with the polygon approximation technique. . 110

5.16 Tomogram layer after Gaussian filter and gray scale threshold. . . 110

5.17 Perspective view of all pore structures after stacking 1024 preprocessed tomograms. 111

5.18 Cube to detect neighboring voxels (3 × 3 × 3 voxels). 111

5.19 Example of extracted pore regions. 112

5.20 Pore structure approximated by a line. 113

5.21 Pore structure approximated by a rectangular cuboid. 114

5.22 Pore structure approximated by an ellipsoid. 114

5.23 Examples of structures approximated by ellipsoids. 115

5.24 Landmarked ellipsoid. 116
5.25 Examplification of convexity values. 116
5.26 Graphic representation of the 3D ASM evaluation. 119
5.27 Pore size frequencies of of the 3 available samples. 120
5.28 Exhaustive pore size interval analysis for the DAH sample. 121
5.29 Exhaustive pore size interval analysis for the ISR sample. 121
5.30 Exhaustive pore size interval analysis for the N8 sample. 122

6.1 Examples of non-elliptical leaves. 127
6.2 Probable extra detections of ellipses. 128
6.3 Problems that might occur with the selected number of perimeter points. 131
6.4 Example cases of erroneous detection of the plant and plant center. 132
6.5 Example of possible applications using ellipse detection. 134

A.1 Gaussian kernel example. 145

LIST OF TABLES

3.1 Numerical values of the 3 most important eigenvalues of the deformation predented in Figure 3.26. 39

4.1 Arrangement of plants within the tray area 47
4.2 Individual plant images used to train each deformable model. . . . 68
4.3 Results of comparing the two ellipse detection methods. 77
4.4 Description of the trays with real plants. 93

5.1 Best fit approximation for each geometrical shape. 115
5.2 Total number of pore regions per sample. 119
5.3 Size of the Gaussian kernel and gray scale threshold for each sample.120
5.4 Best interval ranges to perform ASM identification. 123

6.1 Total number of leaf combinations. 130

LIST OF ALGORITHMS

3.1 Polygon approximation merging technique 15
3.2 Polygon approximation splitting technique 16
3.3 Procrustes analysis 32
3.4 Principal Component Analysis 35
4.1 RGB - Green Extraction 50
4.2 Ellipse detection using Mean shift clustering 52
4.3 Arc extraction 61
4.4 Discover true neighboring arcs of an arc section 63
4.5 Arc combination 64
4.6 Discover all ellipses within a RoI 65
4.7 Plant landmarking procedure 68
4.8 Ellipse combination procedure 72
4.9 Matching real plants with ASM 75
5.1 3D Flood-fill 112
5.2 Exhaustive search of the best pore size interval 118

BIBLIOGRAPHY

Abidi, A. & R.C., G. (1986). Shape decomposition using elliptic fourier descriptors. *Proc.18th IEEE Southeast Symp. Sys. Theory*, (pp. 51 – 61).

Aguado, A. S. & Nixon, M. S. (1995). A new hough transform mapping for ellipse detection. *Image, Speech and Intelligent Systems*.

Akritas, A. G. & Malaschonok, G. I. (2004). Applications of singular-value decomposition (svd). *Mathematics and Computers in Simulation*, 67(1-2), 15 – 31. Applications of Computer Algebra in Science, Engineering, Simulation and Special Software.

Aleixos, N., Blasco, J., Navarrón, F., & Moltó, E. (2002). Multispectral inspection of citrus in real-time using machine vision and digital signal processors. *Computers and Electronics in Agriculture*, 33(2), 121 – 137.

Alien, P., Graham, J., Farnell, D., Marjanovic, E., Adams, J., Jacobs, R., Karayianni, K., Lindh, C., van der Stelf, P., Horner, K., & Devlin, H. (2007). Detecting osteoporosis from dental radiographs using active shape models. In *Biomedical Imaging: From Nano to Macro, 2007. ISBI 2007. 4th IEEE International Symposium on* (pp. 1256 –1259).

Angel, E. & Morrison, D. (1991). Speeding up bresenham's algorithm. *Computer Graphics and Applications, IEEE*, 11(6), 16 –17.

Ansari, N. & Li, K. (1993). Landmark-based shape recognition by a modified hopfield neural network. *Pattern Recognition*, 26(4), 531 – 542.

Antonio, Dell'Aquila (2006). Computerised seed imaging: a new tool to evaluate germination quality. *Communications in Biometry and Crop Science*, 1(1), 20–31.

Atallah, M. J. (1983). A linear time algorithm for the hausdorff distance between convex polygons. *Information Processing Letters*, 17(4), 207–209.

Bai, X. & Latecki, L. J. (2008). Path similarity skeleton graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30, 1282–1292.

Ballard, D. & Brown, C. (1982). *Computer Vision*. Prentice Hall.

Barrow, H. G., Tenenbaum, J. M., Bolles, R. C., & Wolf, H. C. (1977). Parametric correspondence and chamfer matching: two new techniques for image matching. In *Proceedings of the 5th international joint conference on Artificial intelligence - Volume 2* (pp. 659–663). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Bellmann, A., Suthau, T., Stoinski, S., Friedrich, A., Hellwich, O., & Gunga, H. (2005). 3D-modelling of dinosaurs. In *Proceedings 7th Conference on Optical 3-D Measurement Techniques. Vienna*, volume 1 (pp. 3–10).

Berenstein, R., Shahar, O. B., Shapiro, A., & Edan, Y. (2010). Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer. *Intell. Serv. Robot.*, 3, 233–243.

Bhat, S. & Savvides, M. (2008). Evaluating active shape models for eye-shape classification. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on* (pp. 5228 –5231).

Biederman, I. & Ju, G. (1988). Surface versus edge-based determinants of visual recognition. *Cognitive Psychology*, 20(1), 38 – 64.

Blasco, J., Aleixos, N., Cubero, S., Gómez-Sanchís, J., & Moltó, E. (2009a). Automatic sorting of satsuma (citrus unshiu) segments using computer vision and morphological features. *Computers and Electronics in Agriculture*, 66(1), 1 – 8.

Blasco, J., Aleixos, N., Gómez-Sanchís, J., & Moltó, E. (2009b). Recognition and classification of external skin damage in citrus fruits using multispectral data and morphological features. *Biosystems Engineering*, 103(2), 137 – 145.

Bookstein, F. (1991). *Morphometric Tools for Landmark Data*. Cambridge University Press.

Borlaug, N. (1996). Talk transcript from 34th convocation of the indian agricultural research institute, new delhi. 1996. Online; Last accessed on May 2011 at `http://www.archive.org/details/BorlaugIARI`.

Bosch, A., Muñoz, X., & Freixenet, J. (2007). Segmentation and description of natural outdoor scenes. *Image and Vision Computing*, 25(5), 727 – 740.

Bouet, M., Khenchaf, A., & Briand, H. (1999). Shape representation for image retrieval. In *Proceedings of the seventh ACM international conference on Multimedia (Part 2)*, MULTIMEDIA '99 (pp. 1–4). New York, NY, USA: ACM.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Bresenham, J. (1977). A linear algorithm for incremental digital display of circular arcs. *Commun. ACM*, 20, 100–106.

Brosnan, T. & Sun, D.-W. (2002). Inspection and grading of agricultural and food products by computer vision systems–a review. *Computers and Electronics in Agriculture*, 36(2-3), 193 – 213.

Brown, L. G. (1992). A survey of image registration techniques. *ACM Computing Surveys*, 24, 325–376.

Burgard, W., Fox, D., Jans, H., C., M., & Thrun, S. (1999). Sonar-Based Mapping With Mobile Robots Using EM.

Burgos-Artizzu, X. P., Ribeiro, A., Guijarro, M., & Pajares, G. (2011). Real-time image processing for crop/weed discrimination in maize fields. *Computers and Electronics in Agriculture*, 75(2), 337 – 346.

Burlion, N., Bernard, D., & Chen, D. (2006). X-ray microtomography: Application to microstructure analysis of a cementitious material during leaching process. *Cement and Concrete Research*, 36(2), 346 – 357.

Busin, L., Vandenbroucke, N., & Macaire, L. (2008). Color spaces and image segmentation. In P. W. Hawkes (Ed.), *Advances in Imaging and Electron Physics*, volume 151 (pp. 65 – 168). Elsevier.

Carroll, J. P. & Holden, N. M. (2005). A method to quantify weed distribution for relating to patch spraying systems. *American Society of Agricultural Engineers - ASAE*, 48(1), 27–35.

Caunce, A., Taylor, C., & Cootes, T. (2010). Improved 3d model search for facial feature location and pose estimation in 2d images. In *Proceedings of the British Machine Vision Conference* (pp. 81.1–81.10).: BMVA Press. doi:10.5244/C.24.81.

Cheng, H., Jiang, X., Sun, Y., & Wang, J. (2001). Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12), 2259 – 2281.

Cherng, A.-P. & Ouyang, F. (2003). A firmness index for fruits of ellipsoidal shape. *Biosystems Engineering*, 86(1), 35 – 44.

Chetverikov, D. (2003). A simple and efficient algorithm for detection of high curvature points in planar curves. In N. Petkov & M. A. Westenberg (Eds.), *Computer Analysis of Images and Patterns*, volume 2756 of *Lecture Notes in Computer Science* (pp. 746–753). Springer Berlin-Heidelberg.

Comaniciu, D. & Meer, P. (2002). Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5), 603–619.

Cootes, T. & Taylor, C. (2004). *Statistical Models of Appearance for Computer Vision.* Technical Report M13 9PT, Imaging Science and Biomedical Engineering, University of Manchester, U.K. Online; Last accessed on March 2009 at `http://www.isbe.man.ac.uk/~bim/Models/app_models.pdf`.

Cootes, T. F., Hill, A., Taylor, C. J., & Haslam, J. (1994). The use of active shape models for locating structures in medical images. *Image and Vision Computing*, 12(6), 355–366.

Cootes, T. F., Taylor, C. J., Cooper, D. H., & Graham, J. (1995). Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1), 38–59.

Courtney, J. & dePaor, A. (2004). Direct least-squares ellipse fitting. In *Proceedings of the 7th IASTED International Conference on Computer Graphics and Imaging.*

Coxeter, H. S. M. (1989). *Introduction to Geometry (Wiley Classics Library).* Wiley, 2nd edition.

Cplusplus.com (2010). C++ atan2 function. Online; Last accessed on September 2010 at `http://www.cplusplus.com/reference/clibrary/cmath/atan2/`.

Culham, J., He, S., Dukelow, S., & Verstraten, F. A. J. (2001). Visual motion and the human brain: what has neuroimaging told us? *Acta Psychologica*, 107(1-3), 69 – 94.

Daeho, L. & Seung-Gwan, L. (2010). Polygonal approximation of digital curves to preserve original shapes. *[ETRI] ETRI Journal.*

Daliri, M. R. & Torre, V. (2008). Robust symbolic representation for shape recognition and retrieval. *Pattern Recognition*, 41(5), 1782 – 1798.

Davies, R., Twining, C., & Taylor, C. (2008). *Statistical Models of Shape: Optimisation and Evaluation.* Springer Publishing Company, Incorporated.

DCNR (2011). Pennsylvania department of conservation and natural resources. forestry, common trees of pennsylvania. Online; Last accessed on May 2011 at `http://www.dcnr.state.pa.us/forestry/commontr/index.aspx`.

De-An, Z., Jidong, L., Wei, J., Ying, Z., & Yu, C. (2011). Design and control of an apple harvesting robot. *Biosystems Engineering*, In Press, Corrected Proof, –.

de Berg, M., van Kreveld, M., Overmars, M., & Schwarzkopf, O. (2000). *Computational Geometry: Algorithms and Applications.* Springer-Verlag, second edition.

Deegan, K. C., Koivisto, L., Näkkilä, J., Hyvönen, L., & Tuorila, H. (2010). Application of a sorting procedure to greenhouse-grown cucumbers and tomatoes. *LWT - Food Science and Technology*, 43(3), 393 – 400.

Defrise, M. (2001). A short reader's guide to 3d tomographic reconstruction. *Computerized Medical Imaging and Graphics*, 25(2), 113 – 116.

Diamant, E. (2008). Unveiling the mystery of visual information processing in human brain. *CoRR*, abs/0807.0337.

Du, J.-X., Wang, X.-F., & Zhang, G.-J. (2007). Leaf shape based plant species recognition. *Applied Mathematics and Computation*, 185(2), 883 – 893. Special Issue on Intelligent Computing Theory and Methodology.

Duan, L., Yang, W., Bi, K., Chen, S., Luo, Q., & Liu, Q. (2011). Fast discrimination and counting of filled/unfilled rice spikelets based on bi-modal imaging. *Computers and Electronics in Agriculture*, 75(1), 196 – 203.

Duda, R. O. & Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15, 11–15.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification (2nd Edition).* Wiley-Interscience.

Edwards, G. J., Taylor, C. J., & Cootes, T. F. (1998). Interpreting face images using active appearance models. In *Proceedings of the 3rd. International Conference on Face and Gesture Recognition*, FG '98 (pp. 300–305). Washington, DC, USA: IEEE Computer Society.

Fabbri, R., Costa, L. D. F., Torelli, J. C., & Bruno, O. M. (2008). 2d euclidean distance transform algorithms: A comparative survey. *ACM Comput. Surv.*, 40, 2:1–2:44.

Farlex, I. (2010). The Free Dictionary - Dictionary, Encyclopedia and Thesaurus. Online; Last accessed on August 2010 at `http://www.thefreedictionary.com`.

Fitzgibbon, A., Pilu, M., & Fisher, R. (1998). Direct least square fitting of ellipses.

Fitzgibbon, A., Pilu, M., & Fisher, R. B. (2002). Direct least square fitting of ellipses. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5), 476–480.

Flusser, J. (2006). Moment Invariants in Image Analysis. *PROCEEDINGS OF WORLD ACADEMY OF SCIENCE, ENGINEERING AND TECHNOLOGY*, 11, 196–201.

Flusser, J., Zitova, B., & Suk, T. (2009). *Moments and Moment Invariants in Pattern Recognition*. Wiley.

Frank Ayres, E. M. (2008). *Schaum's Calculus*. McGraw-Hill, 5 edition.

Freeman, H. (1961). Techniques for the Digital Computer Analysis of Chain-Encoded Arbitrary Plane Curves. In *Proc. Nat'l Electronics Conf.*, volume 17 (pp. 421–432).

Fu, K. & Mui, J. (1981). A survey on image segmentation. *Pattern Recognition*, 13(1), 3 – 16.

Gallucci, E., Scrivener, K., Groso, A., Stampanoni, M., & Margaritondo, G. (2007). 3d experimental investigation of the microstructure of cement pastes using synchrotron x-ray microtomography. *Cement and Concrete Research*, 37(3), 360 – 368.

Garcia, J. & Gomez, J. (2008). Calibration and evaluation of a machine vision algorithm to detect agricultural tractor movement. *Bulgarian Journal of Agricultural Science*, 14(4), 436–444.

Gavrila, D. (2007). A bayesian, exemplar-based approach to hierarchical shape matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(8), 1408 –1421.

Glasbey, C. A. & Horgan, G. W. (1995). *Image analysis for the biological sciences*. Statistics in practice. New York, NY, USA: John Wiley & Sons, Inc.

Golland, P., Eric, W., & Grimson, L. (2000). Fixed topology skeletons. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1 (pp. 10 –17).

Golub, G. H. & van Loan, C. F. (1996). *Matrix Computations (Johns Hopkins Studies in Mathematical Sciences)(3rd Edition)*. The Johns Hopkins University Press, 3rd edition.

Gonzales-Barron, U. & Butler, F. (2006). A comparison of seven thresholding techniques with the k-means clustering algorithm for measurement of bread-crumb features by digital image analysis. *Journal of Food Engineering*, 74(2), 268–278.

Gonzalez, R. & Woods, R. (2001). *Digital Image Processing*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2nd edition.

González, E., Adán, A., Feliú, V., & Sánchez, L. (2008). Active object recognition based on fourier descriptors clustering. *Pattern Recognition Letters*, 29(8), 1060 – 1071. Pattern Recognition in Interdisciplinary Perception and Intelligence - PRintPerclntel.

Gower, J. (1975). Generalized procrustes analysis. *Psychometrika*, 40, 33–51. 10.1007/BF02291478.

Granlund, G. H. (1972). Fourier preprocessing for hand print character recognition. *Computers, IEEE Transactions on*, C-21(2), 195 – 201.

Gregory, J., Barr, R., Yoshida, K., Reid, D., Silman, A., Hosie, G., Alesci, S., Macfarlane, G., & Aspden, R. (2008). 410 a new extended active shape model to predict total hip replecement in patients with osteoarthritis (oa). *Osteoarthritis and Cartilage*, 16(Supplement 4), S178 – S179. 2008 World Congress on Osteoarthritisheld.

Guojun, G., Chaoqun, M., & Jianhong, W. (2007). *Data Clustering: Theory, Algorithms, and Applications (ASA-SIAM Series on Statistics and Applied Probability)*. SIAM.

Gyaourova, A., Kamath, C., & Cheung, S.-C. (2003). *Block matching for object tracking*. Technical Report UCRL-TR-200271, Center for Applied and Scientific Computing, Lawrence Livermore National Laboratory.

Gée, C., Bossu, J., Jones, G., & Truchetet, F. (2008). Crop/weed discrimination in perspective agronomic images. *Computers and Electronics in Agriculture*, 60(1), 49 – 59.

Hahn, K., Jung, S., Han, Y., & Hahn, H. (2008). A new algorithm for ellipse detection by curve segments. *Pattern Recognition Letters*, 29(13), 1836 – 1841.

Hamilton, H. P. (1830). *Analytical system of Conic sections*. Cambridge University London, third edition edition.

Havlicek, Joseph, J. (1986). Megatrends affecting agriculture: Implications for agricultural economics. *American Journal of Agricultural Economics*, 68(5), pp. 1053–1064.

Hayashi, S., Shigematsu, K., Yamamoto, S., Kobayashi, K., Kohno, Y., Kamata, J., & Kurita, M. (2010). Evaluation of a strawberry-harvesting robot in a field test. *Biosystems Engineering*, 105(2), 160 – 171.

Helmut, A., Bernd, B., & Johannes, B. (1995). Approximate matching of polygonal shapes. *Annals of Mathematics and Artificial Intelligence*, 13(3-4), 251–265.

Hemming, J. (2000). *Computer vision for identifying weeds in crops*. PhD thesis.

Henten, E. J. V., Tuijl, B. A. J. V., Hemming, J., Kornet, J. G., Bontsema, J., & Os, E. A. V. (2003). Field test of an autonomous cucumber picking robot. *Biosystems Engineering*, 86(3), 305 – 313.

Herman, G. T. (2009). *Fundamentals of Computerized Tomography: Image Reconstruction from Projections*. Springer Publishing Company, Incorporated, 2nd edition.

Hilbert, D. (1994). *Theory of algebraic invariants*. New York, NY, USA: Cambridge University Press.

Holzer, S., Hinterstoisser, S., Ilic, S., & Navab, N. (2009). Distance transform templates for object detection and pose estimation. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, 8(2), 179–187.

Huang, K., Wu, Z., Fung, G., & Chan, F. (2005). Color image denoising with wavelet thresholding based on human visual system model. *Signal Processing: Image Communication*, 20(2), 115 – 127.

Huang, Y.-J. & Lee, F.-F. (2010). An automatic machine vision-guided grasping system for phalaenopsis tissue culture plantlets. *Computers and Electronics in Agriculture*, 70(1), 42 – 51.

Huysmans, T., Moens, P., & Audekercke, R. V. (2005). An active shape model for the reconstruction of scoliotic deformities from back shape data. *Clinical Biomechanics*, 20(8), 813 – 821.

Hähnel, D., Burgard, W., & Thrun, S. (2003). Learning compact 3D models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1), 15 – 27. Best Papers of the Eurobot '01 Workshop.

Iñesta, J. M., Buendí, M., & Ángeles Sarti, M. (1998). Reliable polygonal approximations of imaged real objects through dominant point detection. *Pattern Recognition*, 31(6), 685 – 697.

Jefferies, M. E., Cree, M., Mayo, M., & Baker, J. T. (2005). Using 2d and 3d landmarks to solve the correspondence problem in cognitive robot mapping. In C. Freksa, M. Knauff, B. Krieg-Brückner, B. Nebel, & T. Barkowsky (Eds.), *Spatial Cognition IV. Reasoning, Action, and Interaction*, volume 3343 of *Lecture Notes in Computer Science* (pp. 434–454). Springer Berlin / Heidelberg.

Jeon, H. Y. & Tian, L. F. (2009). Direct application end effector for a precise weed control robot. *Biosystems Engineering*, 104(4), 458 – 464.

Ji, Q. & Haralick, R. (1999). A statistically efficient method for ellipse detection. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 2 (pp. 730 –734 vol.2).

Juang, L.-H. & Wu, M.-N. (2010). Mri brain lesion image detection based on color-converted k-means clustering segmentation. *Measurement*, 43(7), 941 – 949.

Kaewapichai, W., Kaewtrakulpong, P., Prateepasen, A., & Khongkraphan, K. (2007). Fitting a pineapple model for automatic maturity grading. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 1 (pp. I –257 –I –260).

Kahn, P. C. (1989). Simple methods for computing the least squares line in three dimensions. *Computers and Chemistry*, 13(3), 191 – 195.

Kalman, D. (1996). A singularly valuable decomposition: The svd of a matrix. *College Math Journal*, 27, 2–23.

Kass, M., Witkin, A., & Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal Of Computer Vision*, 1(4), 321–331.

Kawollek, M. (2006). *Modellbasierte Robotersteuerung zur gartenbaulichen Pflanzenproduktion auf der Grundlage bildanalytischer Verfahren am Beispiel von Gerbera jamesonii*. PhD thesis, Leibniz Universität Hannover. Institut für Technik in Gartenbau und Landwirtschaft., Hannover, Germany.

Kellman, P. J. & Spelke, E. S. (1983). Perception of partly occluded objects in infancy. *Cognitive Psychology*, 15(4), 483 – 524.

Kennedy, J. (2009). *A Fast Bresenham Type Algorithm For Drawing Ellipses*. Technical report, Mathematics Department, Santa Monica College, 1900 Pico Blvd. Santa Monica, CA. 90405, USA.

Keyes, L. & Winstanley, A. (1999). Fourier descriptors as a general classification tool for topographic shapes. In *Proceedings Irish Machine Vision and Image Processing Conference (IMVIP)* (pp. 193–203). Dublin City University, Dublin, Ireland.

Kirk, K., Andersen, H. J., Thomsen, A. G., Jørgensen, J. R., & Jørgensen, R. N. (2009). Estimation of leaf area index in cereal crops using red-green images. *Biosystems Engineering*, 104(3), 308 – 317.

Klette, G. (2002). *Skeletons in Digital Image Processing*. Technical Report CITR-TR-112, CITR, The University of Auckland, New Zealand.

Koch, K., McLean, J., Segev, R., Freed, M. A., II, M. J. B., Balasubramanian, V., & Sterling, P. (2006). How much the eye tells the brain. *Current Biology*, 16(14), 1428 – 1434.

Kolesnikov, A. & Fränti, P. (2007). Polygonal approximation of closed discrete curves. *Pattern Recognition*, 40(4), 1282 – 1293.

Kondo, N., Monta, M., & Fujiura, T. (1996). Fruit harvesting robots in japan. *Advances in Space Research*, 18(1-2), 181 – 184. Physical, Chemical, Biochemical and Biological Techniques and Processes.

Krim, H. & Yezzi, A. (2006). *Statistics and Analysis of Shapes*. Birkhäuser.

Krim, H. & Yezzi, Jr., A. J. (2005). *Statistics and Analysis of Shapes (Modeling and Simulation in Science, Engineering and Technology)*. Birkhauser.

Lakshmi, J. & Punithavalli, M. (2009). A survey on skeletons in digital image processing. In *Digital Image Processing, 2009 International Conference on* (pp. 260 –269).

Lestrel, P. (1997). *Fourier Descriptors and Their Applications in Biology*. Cambridge University Press, USA.

Li, S. Z. (1999). Recognizing multiple overlapping objects in image: An optimal formulation. *IEEE Trans. Image Processing, 9(2):273*, 277, 2000.

Lim, S.-J., Udupa, J. K., Souza, A., Jeong, Y.-Y., Ho, Y.-S., & Torigian, D. A. (2006). A new general method of 3d model generation for active shape image segmentation. 6144(1), 61444B.

LinuxManPages (2010). atan2 function. Online; Last accessed on September 2010 at `http://linux.die.net/man/3/atan2`.

Loncaric, S. (1998). A survey of shape analysis techniques. *Pattern Recognition*, 31(8), 983 – 1001.

Lootens, P., Waes, J. V., & Carlier, L. (2007). Description of the morphology of roots of chicorium intybus l. partim by means of image analysis: Comparison of elliptic fourier descriptors and classical parameters. *Computers and Electronics in Agriculture*, 58(2), 164 – 173.

Lowe, D. G. (1991). Fitting parameterized three-dimensional models to images. *IEEE Transactions on pattern analysis and machine intelligence*, 13, 441–450.

Marchant, J. & Sistler, F. (1993). Computer vision. *Computers and Electronics in Agriculture*, 9(1), vii – viii. Computer Vision.

Marchant, J. A., Onyango, C. M., & Street, M. J. (1990). Computer vision for potato inspection without singulation. *Computers and Electronics in Agriculture*, 4(3), 235 – 244.

Marone, F., Hintermüller, C., McDonald, S., Abela, R., Mikuljan, G., Isenegger, A., & Stampanoni, M. (2009). X-ray Tomographic Microscopy at TOMCAT. volume 186 (pp. 012042+).

Masaphy, S., Zabari, L., Pastrana, J., & Dultz, S. (2009). Role of fungal mycelium in the formation of carbonate concretions in growing media—an investigation by sem and synchrotron-based x-ray tomographic microscopy. *Geomicrobiology Journal*, 26(7), 442–450.

Mclaughlin, R. A. (1998). Randomized hough transform: improved ellipse detection with comparison. *Pattern Recogn. Lett.*, 19(3-4), 299–305.

Mebatsion, H., Boudon, F., Godin, C., Pradal, C., Génard, M., Goz-Bac, C., & Bertin, N. (2011). A novel profile based model for virtual representation of quasi-symmetric plant organs. *Computers and Electronics in Agriculture*, 75(1), 113 – 124.

Mebatsion, H. & Paliwal, J. (2011). A fourier analysis based algorithm to separate touching kernels in digital images. *Biosystems Engineering*, 108(1), 66 – 74.

MedTerms (2011). The medical dictionary. Online; Last accessed on May 2011 at `http://www.medterms.com`.

Mehlhorn, K., Näher, S., Seel, M., Seidel, R., Schilz, T., Schirra, S., & Uhrig, C. (1996). Checking geometric programs or verification of geometric structures. In *In Proc. 12th Annu. ACM Sympos. Comput. Geom* (pp. 159–165).

Meyer, C. D., Ed. (2000). *Matrix analysis and applied linear algebra*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Milborrow, S. & Nicolls, F. (2008). Locating facial features with an extended active shape model. In D. Forsyth, P. Torr, & A. Zisserman (Eds.), *Computer Vision – ECCV 2008*, volume 5305 of *Lecture Notes in Computer Science* (pp. 504–513). Springer Berlin / Heidelberg.

Millner, A. D. & Goodale, M. A. (1996). *The Visual Brain in Action*. Oxford University Press.

Moeslund, T. B., Aagaard, M., & Lerche, D. (2005). 3D Pose Estimation of Cactus Leaves using an Active Shape Model. *Proceedings of the Seventh IEEE Workshops on Application of Computer Vision 2005*, 1(1), 468 – 473.

Moler, C. (2004). *Numerical Computing with MATLAB*. SIAM.

Muller, N., Magaia, L., & Herbst, B. M. (2004). Singular Value Decomposition, Eigenfaces, and 3D Reconstructions. *SIAM Review*, 46(3), 518–545.

Münch, B., Trtik, P., Marone, F., & Stampanoni, M. (2009). Stripe and ring artifact removal with combined wavelet—fourier filtering. *Opt. Express*, 17(10), 8567–8591.

Nakashima, Y. & Kamiya, S. (2007). Mathematica programs for the analysis of three-dimensional pore connectivity and anisotropic tortuosity of porous rocks using x-ray computed tomography image data. *Journal of Nuclear Science and Technology*, 44(9), 1233–1247.

Nam, Y., Hwang, E., & Kim, D. (2008). A similarity-based leaf image retrieval scheme: Joining shape and venation features. *Computer Vision and Image Understanding*, 110(2), 245 – 259.

Neto, J. C., Meyer, G. E., Jones, D. D., & Samal, A. K. (2006). Plant species identification using elliptic fourier leaf shape analysis. *Computers and Electronics in Agriculture*, 50(2), 121 – 134.

Nguyen, T. M., Ahuja, S., & Wu, Q. (2009). A real-time ellipse detection based on edge grouping. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on* (pp. 3280 –3286).

Nickolls, J., Buck, I., Garland, M., & Skadron, K. (2008). Scalable parallel programming with cuda. *Queue*, 6, 40–53.

Norell, K. (2011). Automatic counting of annual rings on pinus sylvestris end faces in sawmill industry. *Computers and Electronics in Agriculture*, 75(2), 231 – 237.

Onyango, C. M. & Marchant, J. A. (2003). Segmentation of row crop plants from weeds using colour and morphology. *Computers and Electronics in Agriculture*, 39(3), 141 – 155.

Onyango, C. M., Marchant, J. A., & Ruff, B. P. (1995). Model based location of pigs in scenes. *Computers and Electronics in Agriculture*, 12(4), 261 – 273.

Oppelt, A. & Grandke, T. (1993). Magnetic resonance imaging. *Superconductor Science and Technology*, 6(6), 381.

Panigrahi, S., Misra, M. K., & Willson, S. (1998). Evaluations of fractal geometry and invariant moments for shape classification of corn germplasm. *Computers and Electronics in Agriculture*, 20(1), 1 – 20.

Pastrana, J. (2005). Gait Optimization for Humanoid Robots. Master's thesis, Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany.

Patel, V. C., McClendon, R. W., & Goodrum, J. W. (1998). Development and evaluation of an expert system for egg sorting. *Computers and Electronics in Agriculture*, 20(2), 97 – 116.

Patla, A. E. (1997). Understanding the roles of vision in the control of human locomotion. *Gait and Posture*, 5(1), 54 – 69.

Pedro, F. & Daniel, H. (2004). *Distance transforms of sampled functions.* Technical report, Cornell Computing and Information Science.

Persson, M. & Åstrand, B. (2008). Classification of crops and weeds extracted by active shape models. *Biosystems Engineering*, 100(4), 484 – 497.

Philipp, I. (2004). *Development and use of automatic weed detection in sugar beet based on image processing.* PhD thesis.

Pierret, A., Capowiez, Y., Belzunces, L., & Moran, C. (2002). 3d reconstruction and quantification of macropores using x-ray computed tomography and image analysis. *Geoderma*, 106(3-4), 247 – 271.

Pitas, I. (2000). *Digital Image Processing Algorithms and Applications.* Wiley-Interscience, First Edition edition.

Pla, F. & Juste, F. (1995). A thinning-based algorithm to characterize fruit stems from profile images. *Computers and Electronics in Agriculture*, 13(4), 301 – 314.

Poppe, R. (2007). Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(1-2), 4–18. Special Issue on Vision for Human-Computer Interaction.

Precht, M. & Kraft, R. (1993). *Bio-Statistik 2.* Muenchen, Germany: Oldenbourg, R.

Premaratne, P., Safaei, F., & Nguyen, Q. (2006). Moment invariant based control system using hand gestures. In D.-S. Huang, K. Li, & G. Irwin (Eds.), *Intelligent Computing in Signal Processing and Pattern Recognition*, volume 345 of *Lecture Notes in Control and Information Sciences* (pp. 322–333). Springer Berlin / Heidelberg.

Prokop, J. & Reeves, A. P. (1992). A survey of moment-based techniques for unoccluded object representation and recognition. In *CVGIP: Graphical Models and Image Processing* (pp. 438–460).

PSIWeb (2011). Paul scherrer institute. Online; Last accessed on May 2011 at `http://www.psi.ch`.

Pérez, A. J., López, F., Benlloch, J. V., & Christensen, S. (2000). Colour and shape analysis techniques for weed detection in cereal fields. *Computers and Electronics in Agriculture*, 25(3), 197 – 212.

Qiao, Y. & Ong, S. (2007). Arc-based evaluation and detection of ellipses. *Pattern Recognition*, 40(7), 1990 – 2003.

Rakun, J., Stajnko, D., & Zazula, D. (2011). Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry. *Computers and Electronics in Agriculture*, In Press, Corrected Proof, –.

Rath, T. (1997). *Methoden zur computerbildanalytischen Pflanzenidentifikation am Beispiel dendrologischer Bestimmungen, Gartenbautechnische Informationen*. Technical Report Heft 42, Leibniz Universität Hannover. Institut für Technik in Gartenbau und Landwirtschaft. Hannover, Germany. Habilitationsarbeit.

Ray, B. K. & Ray, K. S. (1995). A new split-and-merge technique for polygonal approximation of chain coded curves. *Pattern Recognition Letters*, 16(2), 161 – 169.

Reed, J. N., Miles, S. J., Butler, J., Baldwin, M., & Noble, R. (2001). AE–Automation and Emerging Technologies: Automatic Mushroom Harvester Development. *Journal of Agricultural Engineering Research*, 78(1), 15 – 23.

Rocha, A., Hauagge, D. C., Wainer, J., & Goldenstein, S. (2010). Automatic fruit and vegetable classification from images. *Computers and Electronics in Agriculture*, 70(1), 96 – 104.

Russell, S. & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition.

Saengdeejing, A., Charoenlap, N., & Qu, Z. (2002). 2D shape recognition using recursive determination of landmark and fuzzy ART network learning. In *Control, Automation, Robotics and Vision, 2002. ICARCV 2002. 7th International Conference on*, volume 3 (pp. 1620 – 1631 vol.3).

Sakdinawat, A. & Attwood, D. (2010). Nanoscale x-ray imaging. *Nature Photonics*, 4, 840–848.

Sarfraz, M., ul Hassan, M., & Iqbal, M. (2009). Object recognition using fourier descriptors and genetic algorithm. In *Soft Computing and Pattern Recognition, 2009. SOCPAR '09. International Conference of* (pp. 318 – 323).

Schroeder, W., Ken, M., & Lorensen, B. (2006). *Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th Edition*. Kitware.

Schuster, I., Nordmeyer, H., & Rath, T. (2007). Comparison of vision-based and manual weed mapping in sugar beet. *Biosystems Engineering*, 98(1), 17 – 25.

Schönemann, P. (1966). A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31, 1–10. 10.1007/BF02289451.

Seufer, I. & Späth, H. (1999). Least squares fitting of conic sections with type identification by nurbs of degree two. *Mathematical Communications*, 4(2), 207–218.

Shlens, J. (2005). *A Tutorial on Principal Component Analysis*. Technical report, Systems Neurobiology Laboratory, Salk Insitute for Biological Studies.

Smyth, P. P., Taylor, C. J., & Adams, J. E. (1997). "automatic measurement of vertebral shape using active shape models". *Image and Vision Computing*, 15(8), 575 – 581. British Machine Vision Conference.

Søgaard & Heisel (2002). Weed classification by active shape models. *In Automation and Emerging Technologies - AgEng Budapest, Hungary.*

Søgaard, H. T. (2005). Weed classification by active shape models. *Biosystems Engineering*, 91(3), 271 – 281.

Southam, P., Strasser, J., Lee, K., Avondo, J., & Bangham, A. (2009). *UFEEL: Using Haptics and Stereo to Place Landmarks in Three-Dimensional Volumetric Images*, (pp. 2653–2656). IEEE Computer Society.

Spiegel, M., D., R., & Moyer, R. (2010). *Schaum's Outline of College Algebra, Third Edition*. Schaum's Outline Series. McGraw-Hill.

Spreer, W. & Müller, J. (2011). Estimating the mass of mango fruit (Mangifera indica, cv. Chok Anan) from its geometric dimensions by optical measurement. *Computers and Electronics in Agriculture*, 75(1), 125 – 131.

Srivastava, A., Joshi, S., Mio, W., & Liu, X. (2005). Statistical shape analysis: clustering, learning, and testing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(4), 590 –602.

Stampanoni, M., Groso, A., Isenegger, A., Mikuljan, G., Chen, Q., Bertrand, A., Henein, S., Betemps, R., Frommherz, U., Böhler, P., Meister, D., Lange, M., & Abela, R. (2006). Trends in synchrotron-based tomographic imaging: the SLS experience. *Proceedings of SPIE*, 6318(1), 63180M+.

Stewart, G. W. (1993). On the early history of the singular value decomposition. *SIAM Rev.*, 35, 551–566.

Strang, G. (2005). *Linear Algebra and Its Applications*. Brooks Cole, 4th edition edition.

Sugiyama, T., Promentilla, M., Hitomi, T., & Takeda, N. (2010). Application of synchrotron microtomography for pore structure characterization of deteriorated cementitious materials due to leaching. *Cement and Concrete Research*, 40(8), 1265 – 1270.

Suzuki, S. & Abe, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1), 32 – 46.

Sylla, C. (2002). Experimental investigation of human and machine-vision arrangements in inspection tasks. *Control Engineering Practice*, 10(3), 347 – 361.

Søgaard, H. & Lund, I. (2007). Application accuracy of a machine vision-controlled robotic micro-dosing system. *Biosystems Engineering*, 96(3), 315 – 322.

Takahashi, Y., Ogawa, J., & Saeki, K. (2001). Automatic tomato picking robot system with human interface using image processing. *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE*, 1, 433–438.

Tanigaki, K., Fujiura, T., Akase, A., & Imagawa, J. (2008). Cherry-harvesting robot. *Computers and Electronics in Agriculture*, 63(1), 65 – 72. Special issue on biorobotics.

Taylor, C. J., Cooper, D. H., & Graham, J. (1992). Training models of shape from sets of examples. In *In Proc. British Machine Vision Conference* (pp. 9–18).: Springer-Verlag.

Tellaeche, A., BurgosArtizzu, X., Pajares, G., Ribeiro, A., & Fernández-Quintanilla, C. (2008). A new vision-based approach to differential spraying in precision agriculture. *Computers and Electronics in Agriculture*, 60(2), 144 – 155.

Tian, L. F. & Slaughter, D. C. (1998). Environmentally adaptive segmentation algorithm for outdoor image segmentation. *Computers and Electronics in Agriculture*, 21(3), 153 – 168.

Tillett, R. D., Onyango, C. M., & Marchant, J. A. (1997). Using model-based image processing to track animal movements. *Computers and Electronics in Agriculture*, 17(2), 249 – 261. Livestock Monitoring.

Tistarelli, M., Bicego, M., & Grosso, E. (2009). Dynamic face recognition: From human to machine vision. *Image and Vision Computing*, 27(3), 222–232. Special Issue on Multimodal Biometrics - Multimodal Biometrics Special Issue.

Toth, R., Bloch, B. N., Genega, E. M., Rofsky, N. M., Lenkinski, R. E., Rosen, M. A., Kalyanpur, A., Pungavkar, S., & Madabhushi, A. (2011). Accurate prostate volume estimation using multifeature active shape models on t2-weighted mri. *Academic Radiology*, 18(6), 745 – 754.

Treiber, M. A. (2010). *An Introduction to Object Recognition: Selected Algorithms for a Wide Variety of Applications.* Springer Publishing Company, Incorporated, 1st edition.

Trenner, J. H. & Allen, J. (1898). *An elementary course in analytic geometry.* The Cornell Mathematical Series. American Book Company.

Triebel, R., Burgard, W., & Dellaert, F. (2005). Using hierarchical EM to extract planes from 3d range scans. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on* (pp. 4437 – 4442).

Tsechpenakis, G., Rapantzikos, K., Tsapatsoulis, N., & Kollias, S. (2004). Object tracking in clutter and partial occlusion through rule-driven utilization of model-based snakes.

Ullman, Shimon (1996). *High-Level Vision: Object Recognition and Visual Cognition.* The MIT Press, illustrated edition edition.

Unay, D., Gosselin, B., Kleynen, O., Leemans, V., Destain, M.-F., & Debeir, O. (2011). Automatic grading of bi-colored apples by multispectral machine vision. *Computers and Electronics in Agriculture*, 75(1), 204 – 212.

van Aken, J. (1984). An efficient ellipse-drawing algorithm. *IEEE Comput. Graph. Appl.*, 4, 24–35.

Velduis, J. H. & Brodland, G. W. (1999). A deformable block-matching algorithm for tracking epithelial cells. *Image and Vision Computing*, 17(12), 905 – 911.

von Witzke, H., Noleppa, S., & Schwarz, G. (2009). *Global agricultural market trends revisited: The roles of energy prices and biofuel production.* Technical Report Nr. 89/2009, Humbolt - Universität zu Berlin Wirtschafts- und Sozialwissenschaften an der Landwirtschaftlich - Gärtnerischen Fakultät, Philippstr. 13, D-10099 Berlin, Germany.

VXL (2010). The vision-something-libraries. a collection of c++ libraries designed for computer vision. Online at `http://vxl.sourceforge.net/` last accessed on August 2010.

Wakefield, M. & King, J. (1994). A trend analysis of computing in agricultural extension. *Computers and Electronics in Agriculture*, 11(2–3), 239–248.

Weisstein, E. W. (2010a). Ellipse. MathWorld - A Wolfram Web Resource. Online; Last accessed on June 2011 at `http://mathworld.wolfram.com/Ellipse.html`.

Weisstein, E. W. (2010b). Ellipsoid. MathWorld - A Wolfram Web Resource. Online; Last accessed on June 2011 at `http://mathworld.wolfram.com/Ellipsoid.html`.

Wells, P. N. T. (2006). Ultrasound imaging. *Physics in Medicine and Biology*, 51(13), R83.

Wikipedia (2010a). atan2 function. Online; Last accessed on September 2010 at `http://en.wikipedia.org/wiki/Atan2`.

Wikipedia (2010b). Complex numbers. Online; Last accessed on September 2010 at `http://en.wikipedia.org/wiki/Polar_coordinate_system`.

Wikipedia (2010c). Gaussian blur. Online; Last accessed on September 2011 at `http://en.wikipedia.org/wiki/Gaussian_blur`.

Wikipedia (2010d). Voxel. Online; Last accessed on September 2010 at `http://en.wikipedia.org/wiki/Voxel`.

Woebbecke, D. M., Meyer, G. E., von Bargen, K., & Mortensen, D. A. (1995). Color indices for weed identification under various soil, residue, and lighting conditions. *Transactions of the ASAE*, 38(1), 259–269.

Wöltjen, C., Haferkamp, H., Rath, T., & Herzog, D. (2008). Plant growth depression by selective irradiation of the meristem with co2 and diode lasers. *Biosystems Engineering*, 101(3), 316 – 324.

Xu, L., Oja, E., & Kultanen, P. (1990). A new curve detection method: randomized hough transform (rht). *Pattern Recogn. Lett.*, 11, 331–338.

Yoo, Y. (2001). Tutorial on fourier theory. Online; Last accessed on January 2011. `http://www.cs.otago.ac.nz/cosc453/student_tutorials/fourier_analysis.pdf` .

Yuen, H. K., Illingworth, J., & Kittler, J. (1989). Detecting partially occluded ellipses using the hough transform. *Image Vision Computing.*, 7(1), 31–37.

Zhang, D. & Lu, G. (2002). A comparative study of fourier descriptors for shape representation and retrieval. In *Proc. of 5th Asian Conference on Computer Vision (ACCV)* (pp. 646–651).: Springer.

Zhang, D. & Lu, G. (2004). Review of shape representation and description techniques. *Pattern Recognition*, 37(1), 1 – 19.

Zheng, L., Zhang, J., & Wang, Q. (2009). Mean-shift-based color segmentation of images containing green vegetation. *Computers and Electronics in Agriculture*, 65(1), 93 – 98.

Zion, B., Alchanatis, V., Ostrovsky, V., Barki, A., & Karplus, I. (2007). Real-time underwater sorting of edible fish species. *Computers and Electronics in Agriculture*, 56(1), 34 – 45.

Zion, B., Shklyar, A., & Karplus, I. (1999). Sorting fish by computer vision. *Computers and Electronics in Agriculture*, 23(3), 175 – 187.

Åstrandand, B. & Baerveldt, A.-J. (2002). An agricultural mobile robot with vision-based perception for mechanical weed control. *Auton. Robots*, 13, 21–35.

Julio César Pastrana Pérez
Born on July 17th, 1978 in Cunduacan, Tabasco, Mexico
Nationality - Mexican

Education

Nov 2006 - Feb 20102	**Doctoral studies**. Institute of Biological Production Systems. Biosystems and Horticultural Engineering Section. Leibniz Universität Hannover. Hannover, Germany.
Oct 2002 - Oct 2005	**Master of Applied Computer Science**. Albert-Ludwigs-Universität Freiburg. Freiburg im Breisgau, Germany.
Sep 1995 - May 2000	**Bachelor of Science Electronic Systems Engineer**. Instituto Tecnológico y de Estudios Superiores de Monterrey. Monterrey, Mexico.
Jan 1998 - Jun 1998	International Exchange Student. Indiana University. Bloomington, Indiana, USA.

Work Experience

Biosystems and Horticultural Engineering Section. Leibniz Universität Hannover (Germany)
Nov 2006 - Current. Research associate

Colegio de Postgraduados Campus Campeche (Mexico) Jan 2005 - Oct 2006. Junior researcher

NimbRo-Learning Humanoid Robots Laboratory. Albert-Ludwigs-Universität Freiburg (Germany)
Oct 2004 - Oct 2005. Master's degree research

Autonomous Intelligence Systems Laboratory. Albert-Ludwigs-Universität Freiburg (Germany)
Aug 2003 - Mar 2004. Internship and Practical course

La Salle University (Spain) Sep 2001 - Jul 2002. Software developer

Matthias Rath B.V. (Netherlands) Jun 2000 - Jun 2001. Software developer

H.E.B. Supermarkets (Mexico) Jan 2000 - May 2001. Internship

International Programs ITESM (Mexico) Aug 1999 - Dec 1999. Social Service

Electronic Library ITESM (Mexico) Jun 1998 - Apr 1999. Technical Support

Languages

Spanish - Native speaker
English - Fluent speaker
German - Fair speaker

Awards

- StartUp-Impuls 2011. Second place in the science category.
  Topic: Weed control using laser (C. Marx & J. Pastrana)

- DAAD Scholarship holder (2006-2010 Code: A/06/13407)

Publications

Role of Fungal Mycelium in the Formation of Carbonate Concretions in Growing Media - An Investigation by SEM and Synchrotron-Based X-Ray Tomographic Microscopy. S. Masaphy, L. Zabari, J. Pastrana, S. Dultz. Geomicrobiology Journal, vol. 29, no. 7, pp. 442-450. 2009.

3-dimensional vision analysis of highly resolved tomography images of fine pore systems of soil and growing media samples. J. Pastrana, T. Rath, S. Dultz. International conference on agricultural engineering, AgEng, Hersonissos, Crete, Greece, 23-25 June 2008.

Vision system identification of overlapping plants. J. Pastrana, T. Rath. Greensys, International Symposium for High Technology for Greenhouse System, Quebec, Canda. 14.-19.6.2009.

3D Bildverarbeitung von rntgenmikrotomografischen Bodenanalysen im Mikrometerbereich. J. Pastrana, T. Rath, S. Dultz. Bornimer Agrartechnische Berichte, Heft 62, 24-30. Leibniz-Institut fr Agrartechnik Potsdam-Bornim, 2008. ISSN 0947-7314.

Vision based plant recognition under overlapping situations. J. Pastrana, T. Rath. Bornimer Agrartechnische Berichte, Heft 62, 16-23. Leibniz-Institut fr Agrartechnik Potsdam-Bornim, 2008. ISSN 0947-7314.

Vision based plant recognition under overlapping situations in horticulture (Poster). J. Pastrana, T. Rath. First Symposium on Horticulture in Europe. University of Vienna, 17-20.02. 2008, SHE, S. 164, ISSN 1996-9449, ISHS.

Gait Optimization for Humanoid Robots (Master thesis). Julio Pastrana, University of Freiburg, 2005.

Active Localization of People with a Mobile Robot Based on Learned Motion Behaviors. M. Bennewitz, J. Pastrana, and W. Burgard. In: Proceedings of the Third Workshop on Selforganization of Adaptive Behavior (SOAVE), 2004.