APPLICATION OF DIGITAL IMAGE PROCESSING IN AUTOMATED ANALYSIS OF INSECT LEAF MINES

Yee Man Theodora Cho

Doctor of Philosophy

Electronic Engineering

University of York

April 2020

Abstract

Automated species identification has become a popular alternative to manual classification in the past few decades, as a result of advancement in digital image processing techniques and machine learning algorithms. This project aims to devise a new approach for the detection of leaf mines and fungal spots from digital images, and to investigate the possibility of monitoring the growth of leaf mines.

Leaf-mining insects primarily belong to the orders of moths (Lepidoptera), flies (Diptera) and beetles (Coleoptera); or the suborders of sawflies (Symphyta) and wasps (Apocrita). Every spring and summer the larvae of leaf-mining insects feed on leaf tissues until maturity and vacate the mines as adults. As most species of leaf miners attack garden plants or crops, they are generally regarded as pests, despite rarely causing severe long-term detrimental effect on their host plants. Increase in human activities has led to the spread of these invasive species globally in recent years, and the demand for an effective classification system to monitor their distribution is rising consistently.

Samples from three species of leaf-mining insects were included in this project: horse chestnut leaf miner (*Cameraria ohridella*), apple leaf miner (*Lyonetia clerkella*), and holly leaf miner (*Phytomyza ilicis*). Leaves with tar spots (*Rhytisma acerinum*) were also introduced as variations. The proposed method uses image processing techniques such as thresholding, conversion between colour spaces, edge detection, image segmentation, and morphological operations. This project also explores the use of machine learning algorithms as analytical monitoring and predictive tools, using the growth of *C. ohridella* leaf mines as an example.

Contents

Abstract 2

Declaration 6

Acknowledgement 7

List of figures 8

List of tables 11

List of abbreviations 12

1 Introduction 14
- 1.1 Computer-aided taxonomy 15
- 1.2 Motivation for the project 16
- 1.3 Aims and objectives 18
- 1.4 Project overview 19
- 1.5 Implications and potential significance 21
- 1.6 Thesis structure 21

2 Background 22
- 2.1 Automated species identification 22
  - 2.1.1 Related research 25
  - 2.1.2 Leaf Watch 25
- 2.2 Introduction to leaf-mining insects and tar spots 28
  - 2.2.1 Horse chestnut leaf miner 29
  - 2.2.2 Apple leaf miner 35
  - 2.2.3 Holly leaf miner 37
  - 2.2.4 Tar spots 39
- 2.3 Summary 40

3 Digital image processing 42
- 3.1 Colour models 42
  - 3.1.1 RGB and CMY colour models 48
  - 3.1.2 HSI and HSV colour models 48

3.1.1 Colour transformations 53
3.2 Image segmentation 54
3.2.1 Intensity thresholding 54
3.2.2 Edge detection 58
3.3 Morphological image processing 63
3.3.1 Basic morphological algorithms 63
3.3.2 Morphological boundary extraction 65
3.4 Summary 68

4 Automated classification 70
4.1 An overview of classification 70
4.2 Clustering analysis 72
4.2.1 Applications of clustering 73
4.2.2 Clustering algorithms 74
4.2.3 Image segmentation by clustering 76
4.3 Neural networks 78
4.3.1 Feed-forward networks 80
4.3.2 Training algorithms 81
4.3.3 Back propagation 82
4.3.4 Image processing using neural networks 85
4.4 Evolutionary algorithms 86
4.4.1 Genetic algorithms 87
4.4.2 Evolving neural networks 88
4.4.3 Image segmentation using genetic algorithms 89
4.5 Probability for machine learning 91
4.5.1 Bayes' decision theory 91
4.5.2 Principal component analysis 92
4.6 Summary 93

5 Implementation 94
5.1 Data collection 94
5.1.1 Image acquisition using digital cameras 96
5.1.2 Scanning mechanism 97
5.1.3 Image compression and file formats 99
5.2 Image analysis 102
5.2.1 Image preprocessing 104
5.2.2 Image segmentation based on colour spaces 105
5.2.3 Image segmentation based on thresholding 120
5.2.4 Edge-based segmentation 123
5.3 Feature extraction 127
5.4 Summary 129

6 Data analysis 131
6.1 Segmentation results 131
6.2 Daily growth of *C. ohridella* leaf mines 135
6.3 Limitations 146
6.3.1 Challenges in data collection 146
6.3.2 Image acquisition 147

6.3.3 Errors in segmentation 148
6.4 Growth rate analysis 149
6.4.1 Machine learning in MATLAB 150
6.4.2 Curve fitting 151
6.4.3 Performance analysis 155
6.5 Summary 158

7 Conclusion 159
7.1 Future work 161

Appendices
A MATLAB code 162
B Additional data of *C. ohridella* leaf mines 166
C Regression plots and error histograms 173
D Taxa of leaf miners, tar spot and host plants 178

References 181

Declaration

This thesis is a presentation of original work and composed by myself as the sole author. This work, whether in whole or in part, has not been presented or published at this university or other institutions. This research project was carried out under the supervision of Dr David Chesmore, Dr Gianluca Tempesti, and Dr John Szymanski in the Department of Electronic Engineering at the University of York. All sources are acknowledged in the reference section, except where explicitly indicated throughout the thesis.

Acknowledgement

I would like to thank my supervisors, Dr David Chesmore, Dr Gianluca Tempesti, and Dr John Szymanski, for their help and guidance throughout the process of this project. I would especially like to thank Dr Chesmore for proposing this project, and for his encouragement and advice. Thank you to the staff in the Department of Electronics for their help.

It is a great pleasure to thank everyone who helped me with the project, as well as fellow research students and other people whom I have met at the university and worked with for the past few years. Finally, I am most grateful to my parents, who have been supportive and inspiring throughout my life, with their love and patience. Special thanks to my family and all my friends, for encouraging and supporting me.

List of figures

Figure 1.1	Horse chestnut plant infested with *C. ohridella*	19
Figure 1.2	Flow diagram of the proposed system	20
Figure 2.1	Scoring damage to horse chestnut leaves	26
Figure 2.2	Screen shots of the Leaf Watch application	27
Figure 2.3	Leaf watch results	27
Figure 2.4	*C. ohridella* in different life stages	30
Figure 2.5	Infested *Aesculus hippocastanum* leaves	30
Figure 2.6	Growth of larvae	31
Figure 2.7	Incremental distribution of *C. ohridella*, 1984 – 2007	31
Figure 2.8	*C. ohridella* mines on sycamore maple	34
Figure 2.9	Horse chestnut leaf blotches	34
Figure 2.10	*Lyonetia clerkella*	36
Figure 2.11	Distribution map for *L. clerkella* in the United Kingdom	37
Figure 2.12	Leaf mines of *P. ilicis* on *Ilex aquifolium*	38
Figure 2.13	*Phytomyza ilicis* in different life stages	39
Figure 2.14	*Rhytisma acerinum* on *Acer pseudoplatanus*	40
Figure 3.1	Fundamental steps of digital image processing	43
Figure 3.2	Binary, greyscale, and colour images of a leaf mine	44
Figure 3.3	Full-colour, greyscale, and pseudocolour images	45
Figure 3.4	Value at each pixel in binary, greyscale, and RGB images	47
Figure 3.5	Primary and secondary colours	47
Figure 3.6	Various representations of the HSI colour model	50
Figure 3.7	Hue and saturation in the HSI/HSV model	51
Figure 3.8	Example of RGB and HSI component images	52
Figure 3.9	Geometric interpretation of intensity slicing	55
Figure 3.10	Idealised bimodal histogram of intensity levels	57
Figure 3.11	Idealised multimodal histogram of intensity levels	57
Figure 3.12	An ideal vertical edge and its horizontal intensity profile	59
Figure 3.13	A 3 × 3 region of an image	60

Figure 3.14	Masks of the Robert, Prewitt, and Sobel edge operators	60
Figure 3.15	The modified masks for detecting diagonal edges	61
Figure 3.16	A 5 × 5 mask approximation of the LoG function	62
Figure 3.17	Erosion	64
Figure 3.18	Dilation	64
Figure 3.19	Boundary detection using erosion	66
Figure 3.20	Hole filling	66
Figure 3.21	Structuring elements of morphological opening operations	67
Figure 3.22	Edges of leaf mines on the RGB and binary images	69
Figure 4.1	Flow diagram representing main steps in classifier design	71
Figure 4.2	Tree of classification types	72
Figure 4.3	Clustering methodology	74
Figure 4.4	Example of a dendrogram	75
Figure 4.5	Schematic diagram of a McCulloch-Pitts neuron	79
Figure 4.6	Feed-forward networks	79
Figure 4.7	A two-layer feed-forward network	80
Figure 4.8	Back propagation in a three-layer network	83
Figure 4.9	Image recognition	83
Figure 4.10	Typical CNN structure	85
Figure 4.11	Block diagram of the adaptable image segmentation process	90
Figure 4.12	Conceptual design of a multi-level computer vision process	90
Figure 5.1	A decimal one penny coin scanned at 300 dpi.	99
Figure 5.2	Image compression	100
Figure 5.3	Baseline JPEG encoder and decoder	101
Figure 5.4	Block diagram of the JPEG 2000 encoder and decoder	101
Figure 5.5	2-D Cartesian coordinates in MATLAB	103
Figure 5.6	RGB colour rendition of the original colour scene	106
Figure 5.7	RGB and HSI components of a full-colour image	107
Figure 5.8	Original images used in figures 5.11 and 5.12	109
Figure 5.9	*L. clerkella* – segmentation using the HSI colour space	110
Figure 5.10	*R. acerinum* – segmentation using the HSI colour space	114
Figure 5.11	Products of saturation binary masks and hue images	118
Figure 5.12	Masking the green pixels	119
Figure 5.13	Multilevel thresholding	121
Figure 5.14	Multilevel thresholding of a *C. ohridella* leaf mines	122
Figure 5.15	Removing a small object from a binary image	124
Figure 5.16	A scanned image of a *C. ohridella* leaf mine	124
Figure 5.17	A flat diamond structure element object	125
Figure 5.18	The process of identifying spots caused by *R. acerinum*	126
Figure 5.19	Calculating area and perimeter in MATALB	128
Figure 5.20	4-connected connectivity and 8-connected connectivity	128
Figure 5.21	Flow diagram of the image processing section	130

Figure 6.1	Segmentation results of *C. ohridella* leaf mines	133
Figure 6.2	Segmentation results of *R. acerinum* tar spots	133
Figure 6.3	Segmentation results of *L. clerkella* leaf mines	134
Figure 6.4	Segmentation result of a *P. ilicis* leaf mine	134
Figure 6.5	Leaf mine example #1	136
Figure 6.6	Leaf mine example #2	137
Figure 6.7	Leaf mine example #3	139
Figure 6.8	Leaf mine example #4	140
Figure 6.9	Leaf mine example #5	141
Figure 6.10	Leaf mine example #6	142
Figure 6.11	Leaf mine example #7	143
Figure 6.12	Leaf mine example #8	143
Figure 6.13	Leaf mine example #9	143
Figure 6.14	Leaf mine example #10	144
Figure 6.15	Leaf mine example #11	144
Figure 6.16	Leaf mine example #12	144
Figure 6.17	Leaf mine example #13	145
Figure 6.18	Leaf mine example #14	145
Figure 6.19	Sample neural network in MATLAB	151
Figure 6.20	Workflow for training regression models	151
Figure 6.21	Fitting data with generalised linear regression	153
Figure 6.22	Fitting data with generalised linear regression	154
Figure 6.23	The growth of 14 horse chestnut leaf mines	154
Figure 6.24	Two-layer feed-forward neural network	155
Figure 6.25	Regression plots	156
Figure 6.26	Error histogram	156
Figure 6.27	Regression plots after retraining	157
Figure 6.28	Error histogram after retraining	157
Figure B1	The growth of 8 horse chestnut leaf mines	169
Figure C1	The two-layer feed-forward network	173
Figure C2	Additional regression plots after retraining	175
Figure C3	Additional error histogram after retraining	175
Figure C4	Additional regression plots after retraining	176
Figure C5	Additional error histogram after retraining	176
Figure C6	Input-output fitting	177
Figure C7	Input-output fitting (after retraining)	177

List of tables

Table 2.1	Interpretation of the scoring system of Leaf Watch	26
Table 2.2	Rank table of soybean leaf spot disease	28
Table 3.1	Colours in RGB and HSI representations	51
Table 6.1	Area of *C. ohridella* leaf mine #1	138
Table 6.2	Area of *C. ohridella* leaf mine #2	138
Table 6.3	Area of *C. ohridella* leaf mine #3	140
Table 6.4	Area of *C. ohridella* leaf mine #4	140
Table 6.5	Area of *C. ohridella* leaf mine #5	141
Table 6.6	Area of *C. ohridella* leaf mine #6	142
Table B1	Area of *C. ohridella* leaf mine #7	166
Table B2	Area of *C. ohridella* leaf mine #8	166
Table B3	Area of *C. ohridella* leaf mine #9	167
Table B4	Area of *C. ohridella* leaf mine #10	167
Table B5	Area of *C. ohridella* leaf mine #11	167
Table B6	Area of *C. ohridella* leaf mine #12	168
Table B7	Area of *C. ohridella* leaf mine #13	168
Table B8	Area of *C. ohridella* leaf mine #14	168
Table B9	Area of *C. ohridella* leaf mines #13 and #14	168
Table B10	Area of *C. ohridella* leaf mine #15	169
Table B11	Area of *C. ohridella* leaf mine #16	170
Table B12	Area of *C. ohridella* leaf mine #17	170
Table B13	Area of *C. ohridella* leaf mine #18	170
Table B14	Area of *C. ohridella* leaf mine #19	171
Table B15	Area of *C. ohridella* leaf mine #20	171
Table B16	Area of *C. ohridella* leaf mine #21	171
Table B17	Area of *C. ohridella* leaf mine #22	172

List of abbreviations

Leaf-mining insects and plant diseases

C. ohridella	*Cameraria ohridella* (horse chestnut leaf miner)
G. aesculi	*Guignardia aesculi* (horse chestnut leaf blotch)
L. clerkella	*Lyonetia clerkella* (apple leaf miner)
P. ilicis	*Phytomyza ilicis* (holly leaf miner)
R. acerinum	*Rhytisma acerinum* (tar spots)

Host plants of leaf mines

A. hippocastanum	*Aesculus* (horse chestnut or buckeye)
A. turbinata	*Aesculus turbinata* (Japanese horse chestnut)
A. octandra	*Aesculus octandra* (American species buckeye)
A. glabra	*Aesculus glabra* (Ohio buckeye)
A. sylvatica	*Aesculus sylvatica* (painted buckeye)
A. pavia	*Aesculus pavia* (red buckeye)
A. indica	*Aesculus indica* (Indian buckeye)
A. chinensis	*Aesculus chinensis* (Chinese buckeye)
A. californica	*Aesculus californica* (California buckeye)
A. flava	*Aesculus flava* (yellow buckeye)
A. platanoides	*Acer platanoides* (Norway maple)
A. saccharinum	*Acer saccharinum* (silver maple)
A. pseudoplatanus	*Acer pseudoplatanus* (sycamore maple)

Automated classification or machine learning-related

AI	Artificial intelligence
NNs	Neural networks
ANNs	Artificial neural networks
CNNs	Convolutional neural networks
LTSMs	Long-short term memory networks
MLP	Multilayer perceptron
BP	Backpropagation

SVMs	Support vector machines
MLR	Multiple linear regression
EAs	Evolutionary algorithms
GAs	Genetic algorithms
SHAN	Sequential, agglomerative, hierarchical, nonoverlapping
PCA	Principal component analysis
PC(s)	Principal component(s)
MSE	Mean squared error

Colour models

RGB	Red, green, and blue
CMY	Cyan, magenta, and yellow
CMYK	Cyan, magenta, yellow, and black
HSI or HSV	Hue, saturation, and intensity (or value)

Miscellaneous

CAT	Computer-aided taxonomy
MATLAB	MATrix LABoratory
DAISY	Digital Automated Identification System
CCD	Charger coupled device
CMOS	Complementary metal-oxide semiconductor
BSI-CMOS	Back-illuminated CMOS
ISO	International Organization for Standardization
DPI or dpi	Dots per inch
PPI or ppi	Pixels per inch
MP	Megapixels
MB	Megabytes
JPEG or JPG	Joint Photographic Experts Group
DCT	Discrete cosine transform
FDCT	Forward discrete cosine transform
IDCT	Inverse discrete cosine transform
PNG	Portable network graphic
TIFF	Tagged image file format
CAD	Computer-aided diagnostic
DNA	Deoxyribonucleic acid
DBE	Detect-before-extract
AG	Average grey-level
AF	Average feature
MRI	Magnetic resonance imaging
CT	Computer tomography
EM	Electromagnetic
ADC	Analogue-to-digital converter
GUI	Graphical user interface

Chapter 1

Introduction

Over the past few decades, with the development of new technologies and techniques, automated species identification using digital image processing has emerged as a powerful tool. Fundamentally, automated classification can be considered as a method to organise a data set, and interpreting them in a way so that information can be extracted and understood [1].

Compared to the more traditional way of comparing data manually, automated species identification can be seen as the natural progression, with the additional benefit of its cost-effectiveness, and the possibilities to improve on accuracy and precision. Computerised methods require the expertise and efforts of engineers and scientists from a wide range of fields; and their collaborations throughout in recent years have resulted in some encouraging and promising results. Similar techniques are also applied in the analysis of magnetic resonance imaging (MRI) images and computed tomography (CT) scans for medical purposes, and for processing remote sensor data of geological elements. As a result, the popularity and demand for automated classifiers has been consistently on the rise.

In the fields of entomology and zoology, routine species identification traditionally was mainly conducted under laboratory settings, with experts comparing specimens. While the conventional way is reliable, its cost-effectiveness has come into question as it requires specialised knowledge and training [2]; and with more species of animals and plants being discovered in the past few decades, improvement on efficiency and accuracy was understandably

desired. While the beginning of digital image processing can be dated back to the 1960s [3, pp. 3 – 7], research on automated species identification first dates back to the 1980s, as an alternative approach to manual species classification. Since then, with the rapid development of devices for capturing and displaying digital images, techniques for digital image processing and automated classification have been improving accordingly. Despite misgivings of some over its long-term efficacy, the popularity of computer-sided taxonomy (CAT) surges and such systems are much sought after.

1.1 Computer-aided taxonomy

Taxonomy, in its widest sense, is the practice of discovering and classifying individual species of plants and animals. Pioneers such as Adanson and Linné devised innovative systems to classify biological organisms, with the former applying clustering techniques [4] and the latter popularising a new way to name species, which made identification more efficient [5]. Centuries later Sokal and Sneath introduced polythetic classification, in which characteristics of taxonomic units were studied and taken into account during the identifying process [6].

In its early days, species identification was performed manually, which often included making observation and taking physical measurements. Morphological features such as shape, colour, sizes etc. have always played important roles in species identification. Manual classification requires the expertise of taxonomists or curators, and could become time-consuming when a large database is needed to provide ecological information [7]; the advance in technology, alongside the need for rapid field identification, gave rise to the first automated computer-aided taxonomy (CAT) identification systems. One of pioneering applications was DAISY (Digital Automated Identification System), a universal identification systems developed by O'Neill [8] in the mid-1990s, which is still in-use nowadays for monitoring agricultural crops and biodiversity [9].

Compared to the traditional approach, CAT systems have the advantage of its accessibility and usability outside a laboratory environment. While promising results have assured the ability of performing species identification using automated or computer-based systems, this approach has yet to emerge as the

typical way of routine identification, since procedures such as system training and image processing have not been widely adopted in biodiversity research.

Gaston and O'Neill suggested that automated identification was considered by some as (1) difficult – complicated algorithms and processes are often incorporated; (2) labour intensive – traditionally the identification process required the expertise of taxonomists, however computer-based systems would have to be planned and developed by engineers; (3) threatening – it would take time for the taxonomic industry to embrace this modernised and diverse approach; (4) different – the traditional approach had been performed for decades and its viability was proven; and 5) expensive – a mix of hardware and software had to be invested into [10].

It should be mentioned that another alternative to the traditional method is a molecular approach based on DNA full barcoding – by extracting, amplifying and sequencing the DNA (deoxyribonucleic acid) of organisms. DNA-based methods are considered important development in taxonomy and systematics, with molecular data being a vital part in the analysis of biodiversity [11]. DeSalle contended that even though there are problems with DNA barcoding applications, the use of DNA sequence information can contribute to refining species discovery [13]. However, Hebert *et* al. acknowledged the prevalence of misidentification due to the variation in molecular data across, especially beyond Order-level [12]. Will and Rubinoff argued that DNA barcoding should not replace morphology for species-level identification and classification, or this could potentially impede the understanding of biodiversity [11].

1.2 Motivation for the project

Taxonomic impediment refers to the gaps of knowledge in the taxonomic system, the shortage of skilled and trained taxonomists or curators, and how the ability to facilitate effective biodiversity management could be affected as a result [14]. The three main areas of taxonomy in biodiversity conservation, as decreed in the Darwin Declaration, are: identification, assessment and monitoring – including biological surveys and improvement in rapid assessment methods; conservation – the identification of taxa in need of conservation action; and sustainable use,

with the objective to achieve sustainable agriculture, horticulture and forestry, as well as biological and diseases control, and managing invasive species [14].

Leaf miners are a group of invasive insects which feed on leaves of their host plants and are predominantly regarded as pests. The damage caused by leaf-mining insects vary considerably regarding the species of leaf miners and host plants, as well as the environmental conditions and stages of development. Leaf miners generally damage the aesthetical value of the plants, and can lead to lower crop production.

C. ohridella, for example, is a species of leaf-mining insects that has come to prominence in the past few decades. Its main host plant *Acer hippocastanum* is a common ornament tree of choice in urban areas, and when combined with the increase in human activities, this means the population of *C. ohridella* was allowed to spread rapidly across the Europe. *C. ohridella* has been observed to be feeding on secondary host plants in areas where horse chestnut plants, their primary host plant, are especially heavily infested [15] [16] [17]. This essentially poses a new threat to other species of chestnut and maple trees, as the expansion in areas infested by *C. ohridella* has increased significantly in the past few decades, both within the United Kingdom and across Europe.

This thesis proposed an approach for measuring and monitoring the growth of these invasive species, based on the detection of leaf mines from digital images. This new approach can provide useful information for entomologists and biologists who study the insects and their host plants, thus fulfilling some of the goals illustrated in the Darwin Declaration. Effective pest control measures can be applied, benefitting the agricultural and gardening communities alike.

Although the popularity of computer-aided taxonomy continues to rise, most existing automated identification systems for plants focus on the detection of bacteria, viruses or fungi in agricultural crops. Species identification system designed specifically for detecting or classifying leaf mines remained limited. Image-based automated classification systems have been developed for the following plant species: sugar beat [18], different species of crops native to Tamil Nadu [19], pomegranate [20], potato [21], orchard [22], maize [23], cucumber [24], chilli plants [25], wheat and grape [26], soybean [27], oil palm [28] etc.

The alternative approach of using DNA barcoding, meanwhile, involves recovering DNA from empty leaf mines and has only started to emerge in the past decade. This generally requires the laborious process of scraping out recently vacated leaf mines to collect larval skins and frass, which comes with the difficulties of determining precise plant – leaf miner interactions based on residue within leaf mines [29]. The success rate of this molecular approach in amplification varies, although molecular techniques for species identification are consistently advancing. Yet by contrast to the traditional method of comparing samples and computer-aided identification systems, the molecular approach tends to be more complex and costly [30].

1.3 Aims and objects

The aim of this project is to investigate methodologies and develop tools that could aid the development of a semi-automated system for detecting leaf mines on digital images of leaves. This incorporates various digital image processing techniques commonly used in automated classifiers and computer-aided taxonomy systems, which have been proved to be suitable for the purposes, but rarely used in the monitoring of leaf mines. Three species of leaf miners are considered, including the horse chestnut leaf miner (*Cameraria ohridella*), apple leaf miners (*Lyonetia clerkella*), and holly leaf miners (*Phytomyza ilicis*), as well as the fungi more commonly known as "tar spots" (*Rhytisma acerinum*). This project also proposes a method to monitor the growth of *C. ohridella* leaf mines, and investigate into the possibility of predicting the trend of growth of the larvae.

The horse chestnut leaf miner *Cameraria ohridella*, as seen in figure 1.1, was initially chosen as a primary focus as samples are widely available compared to procuring images of fields of agricultural crops with various plant diseases. In order to ensure the proposed method is not overfitted to suit a single species, variation is then introduced by including specimens of apple leaf miners (*Lyonetia clerkella*), and holly leaf miners (*Phytomyza ilicis*). Each of these variations has different characteristics compared to *C. ohridella* – *L. clerkella* leaves gallery mines on surfaces of leaves, instead of blotch mines; and *P. ilicis* is a fly, unlike the other two, which are moths. This system is also modified to accommodate the fungus more commonly known as tar spots (*Rhytisma acerinum*).

Figure 1.1: Horse chestnut plant infested with C. ohridella

1.4 Project overview

Similar to many computer vision systems, the proposed leaf mine detection tool combines digital image processing, pattern recognition and artificial intelligence. It uses a range of image processing techniques commonly seen in other automated classifiers and computer-aided taxonomy systems such as intensity thresholding, conversion between colour spaces, morphological operations, edge detection, and other types of image segmentation. The segmentation results aid the process of feature extraction from image data, and the interpretation of these features shows that leaf mines or tar spots can be located on a digital image which shows the surface of a leaf.

The use of neural networks, in conjunction with evolutionary algorithms, will be explored as a tool to aid the classification of leaf mine infestation. A reference

database serves as a training set for testing algorithms, before they are incorporated into the automated classification system. This provides an efficient and semi-automated way to measure infested areas and severity. The flow diagram in figure 1.2 describes how the proposed system detects damaged areas caused by leaf mines.

In order to adapt to accommodate the further two species of leaf mines, as well as data collected with other methods, it is noted that certain modification has to be adapted. This is due to the shape, colour and sizes of the mines or spots, as well as the dimension of the digital images that are acquired with different methods. The system can potentially work similarly on mines of *C. ohridella* on various species of host plants if samples become available; assuming the same methods can be used to estimate a threshold when separating the leaf mines from the rest of the leaf. Based on observation, it can be speculated that the approach could be adjusted to suit other plant diseases.

This project also proposes a method to monitor the growth rate of blotch leaf mines using feature extracted from images of leaf mines. The area of leaf mines is measured using the proposed system over a number of days, which can be shown to correspond to various growth stages of the larvae. *C. ohridella* mines are chosen for this purpose due to the amount of data available. The training of the dataset using neural network facilitates the investigation into the possibility of predicting the trend of growth of the larvae.

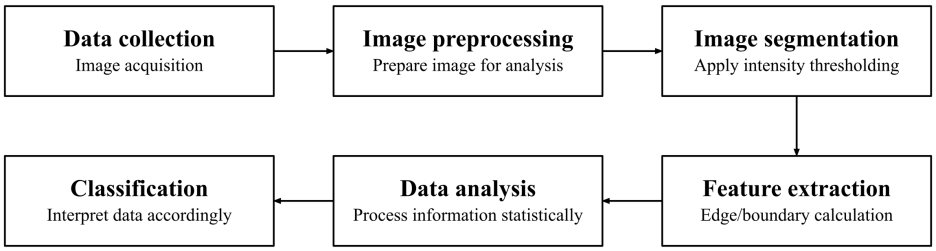


Figure 1.2: Flow diagram of the proposed system

1.5 Implications and potential significance

The accurate identification of leaf-mining insects is an ardour process – apart from during oviposition, adult leaf miners do not generally associate with their host plants [30]. Their larvae, meanwhile, are camouflaged to a certain degree during most of their life cycle. The leaf tissues provide a certain amount of protection against physical environment and predators, which complicates the management of these invasive pests. Nevertheless, the correction identification of leaf-mining species is important for the monitoring of the invasive species.

By basing the approach on images of leaf mines, features of the mines can be extracted and used as the parameters to aid the classification of leaf miners. A challenge for efficient identification of leaf mines is the relationship between leaf miner and host plants. The size and shape of leaf mines vary, depending on their species, development stage, the condition of the host plants, and the external environment. For instance, the growth of a larva can be impeded by the edges or veins of the leaf; similarly, its growth can also be constricted by existing leaf mines or areas that are otherwise damaged.

The proposed method facilitates the measurement of morphological properties of leaf mines, which are commonly used in many related research. The tools for detecting leaf mines from images are fundamental to any image-based classification system for the identification of leaf mines or leaf-mining insects, thus assisting the research into the behavioural development of leaf miners.

1.6 Thesis structure

In this report, chapters 2 provides the background information of the project, on the three species of leaf-mining insects, previous studies of automated species identification and other related research. Chapters 3 and 4 discuss the techniques of digital image processing and routine identification using machine learning respectively. Chapter 5 illustrates how the techniques have been incorporated and implemented in this project, and chapter 6 would provide insight to the data analysing methods. The report ends in chapter 7 with a conclusion and suggestions on side projects or future work.

Chapter 2

Background

Species identification using computer vision systems is becoming more significant in the fields of taxonomy and entomology. This chapter provides an overview of some existing computer-aided taxonomy systems designed to detect plant diseases using digital images. The majority of examples that will be discussed in this chapter incorporate image processing techniques, such as image segmentation and various morphological processes; feature extraction based on texture, colour etc.; before using machine learning algorithms such as neural networks as classifiers, and occasionally incorporating techniques from the fields of pattern recognition and artificial intelligence.

This chapter also elaborates on the biological background of horse chestnut leaf miner (*Cameraria ohridella*), apple leaf miners (*Lyonetia clerkella*), holly leaf miners (*Phytomyza ilicis*), and the fungus more commonly known as tar spots (*Rhytisma acerinum*).

2.1 Automated species identification

Computer-aided taxonomy systems are being introduced as an alternative to the traditional species identification method of comparing samples manually, and as a response to the demand in routine identification for new species. CAT systems are seen as more efficient than manual classification based on morphological features or behavioural attributes, and as a less costly or complex approach compared DNA barcoding,

As mentioned in section 1.2, most plant-related computer-aided taxonomy systems or automated classifiers to date were developed for plant diseases; nonetheless, some of the techniques and methodologies are applicable for the detection of leaf mines from digital images. As a result, these CAT systems are significantly relevant to this current research project. Such systems usually incorporating image processing techniques, such as image segmentation and various morphological processes; feature extraction based on texture, colour etc.; and neural networks, which is often used as an automated classifier.

For instance, Dhaygude and Kumbhar [18], Arivazhagan *et al*. [16], Iandge *et al*. [23] and Shen *et al*. [27] devised systems that convert images of diseased leaves from RGB (red, green and blue) colour model to HSI (hue, saturation and intensity), thereby masking or removing green pixels, which indicate areas as background, before segmenting the image into its binary equivalent and using neural networks as the classifier. In their investigation in leaf diseases caused by fungi, bacteria and viruses, Dhaygude and Kumbhar used sugar beets leaves [18]; Arivazhagan *et al*. incorporated 500 leaves from 30 different plant species native to Tamil Nadu, including banana, beans, jackfruit, lemon, mango, potato, tomato and sapota [16]. Iandge *et al*. proposed and experimentally evaluated a method of detecting leaf diseases automatically by means of neural network, using maize plants, which are prone to stem borer (*Chilo partellus*) infestation and brown stripe downy mildew disease [23]. Shen *et al*. devised a mechanism for grading damage of grey spots on leaves of soya bean plants by calculating the quotient of disease spots; it used Otsu's method and the Sobel operator for image segmentation and edge detection respectively [27].

Some of these experiments focused on extracting certain features from digital images, and using them to help identify the diseases, such as Dhaygude and Kumbhar [18], Patil and Bodhe [31]. Hainruddin *et al*. used morphological processes and extracted features using an algorithm based on image texture and colour, corresponding to four types of macronutrient and micronutrient nutrient deficiencies in oil palm (*Elaeis guineensis*), based on the fertilisers used [28]. Husin *et al*. used 107 samples of chilli plants, of which 21 were healthy and 86 were affected by insects, bacteria, fungi or viruses [32]. This system incorporates image processing techniques including Fourier filtering, edge detection and

morphological operations to enhance images and extract features from them; as well as a user interface constructed with LABVIEW for capturing images.

Neural networks have been one of the most popular classifiers for computer-aided taxonomy systems, and these computational models are widely acknowledged as successful classifiers in many existing applications. Used by Arivazhagan *et al.* [16], Kulkarni and Patil [20], Iandge *et al.* [23], Wang *et al.* [26] [33] among others, neural networks are used in the classification stage. Iandge *et al.* achieved a high percentage of success in identifying a range of diseases of maize plants; and inferred that the diseased leaves show significant differences noticeable in neural outputs when compared to healthy equivalents [23]. Wang *et al.* achieved accuracy above 90% [26] [33]. Phadikar and Sil devised a software prototype system for detecting rice plant diseases leaf black (*Magnaporthe grisea*) and brown spot (*Cochliobolus Miyabeanus*) using SOM, an unsupervised learning technique [34]. Using SOM and digital images of rice plants, Phadikar and Sil achieved an accuracy of 70 – 92%.

Occasionally used in conjunction with neural networks are other pattern recognition algorithms, such as the application of *K*-means clustering to separate pixels into groups before passing the segmented images through neural networks. Experiments of Al-Hiary *et al.* and Bashish *et al.* used this segmentation technique on leaf samples from the Al-Ghor area in Jordan, to detect and classify plant diseases such as early scorch, cottony mould, ashen mould, late scorch and tiny whiteness [35] [36].

Although the computer-aided taxonomy systems described here were generally regarded as successful, some reported misclassification due to the inconsistency of symptoms in plant diseases and the number of features taken into consideration [19] [20] [26] [34]. Suggestions to improve classification accuracy include: increasing size of training samples [19], classifying pixels separately [37], and the development of hybrid algorithms of generic algorithms and neural networks [23] [35].

In the following chapter, basic digital image processing techniques are discussed, as well as their application in other fields, which can be applied computer-aided taxonomy systems to improve segmentation results.

2.1.1 Related research

Considering automated species identification has been applied to numerous research areas, such as entomology, biodiversity, ecology, agriculture, there are approaches other than image-based systems. Acoustics, for instance, was chosen as the medium for identifying beetle larvae in imported goods by Schofield and Chesmore [38]. Acoustics have also been used in species identification in other animals such as birds [39] [40] [41], lemur [42] and grasshoppers [43].

The application of image-based CAT systems is not limited to detecting plant diseases in agricultural and horticultural systems. The aforementioned DAISY project established the possibility of automating the routine identification process, and many insect-based automated systems have since been developed. These includes the classifiers for moths developed by Mayo and Watson [44], Bastista *et al.* [18], Wen *et al.* [22], Yu *et al.* [46], and Flandin [47]. Kang *et al.* proposed an identification process for seven species of butterflies [48], and Larios *et al.* devised a method to classify four stoneflies taxa [9].

In relation to CAT systems for plant diseases, there are existing mobile applications for plant identification using digital image processing and artificial intelligence, including: PlantSnap [50], PictureThis [51], My Garden Answers [52], iPlant [53], SmartPlant [54], Plantifier [55], Plant Identification [56], Smart Identifier: Plant+Insect [57], Pl@ntNet [58], Garden Flower Identification [59], Plant Identification & Info [60], and WildPlantID [61]. Such systems in general involve photos taken with smartphones, and often with a varying degree of user input. These applications often have the functionality to classify common house or garden plants, with additional care information; yet few focus on identifying pests or plant pathogens. Their target audience is mostly plant owners who wish to learn how to care for their plants at home, and entomologists or taxonomists who wish to study them, however.

2.1.2 Leaf Watch

Although not an automated classification systems, Leaf Watch is one of the very few projects that concerns horse chestnut leaf miners, and incorporates digital

image processing techniques with elements of classification [62]. It is a mobile application which allows users to upload photographs of horse chestnut leafs invested with leaf mines. The photographs, with the additional information of geographic information, form a database which helps with the classification of severity, in the form of a scoring system, as seen in figure 2.1 and table 2.1 [63] [64] [66]. Some screenshots of the mobile application are shown in figure 2.2.

This project was a collaboration between the University of Hull and University of Bristol, led by Evans and Pocock [63] [65] [66] [67], with the group of participants who had very limited knowledge on either the leaf miners or the horse chestnut plant. Participants were provided damaged leaves of horse chestnut *Aesculus hippocastanum* and asked to identify, as well as score, the damage according to the diagram in figure 4.10 [68]. Figure 2.3 shows the results gathered by the Leaf Watch application in 2012. The numbers inside the pins represents the number of entries submitted to the application in the corresponding area. A grey pin denotes unknown amount of damage observed, a white pin indicates no damage (damage score = 0); and darker pins represent ascending damage scores from 1 to 4 [66].

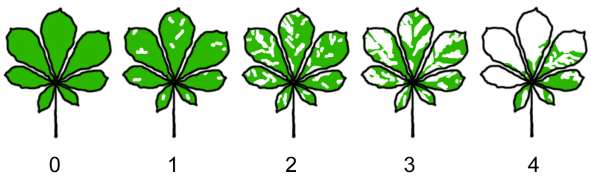
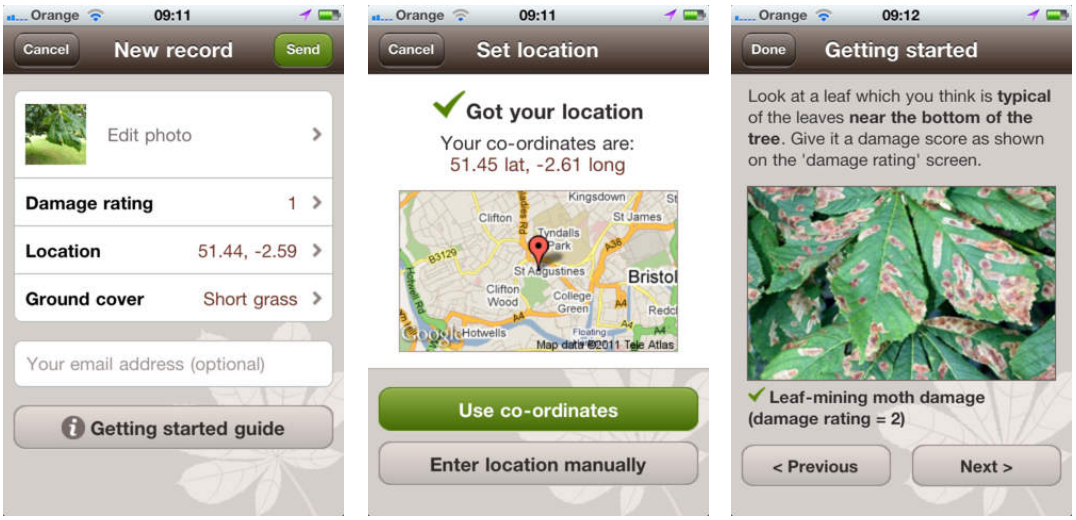


Figure 2.1: Scoring damage to horse chestnut leaves [63]

Score	Description of leaves
0	The leaf is completely green, no visible evidence of moth attack.
1	There are a couple of white or brown mines, showing little damage.
2	Less than half of the leaf is covered with white or brown leaf mines.
3	Around half of the leaf is covered with white or brown leaf mines.
4	The white or brown mines definitely cover more than half the leaf. (In some cases the mines may cover the entire leaf.)

Table 2.1: Interpretation of the scoring system of Leaf Watch [66]



(a) *(b)* *(c)*

Figure 2.2: Screen shots of the Leaf Watch application [64]

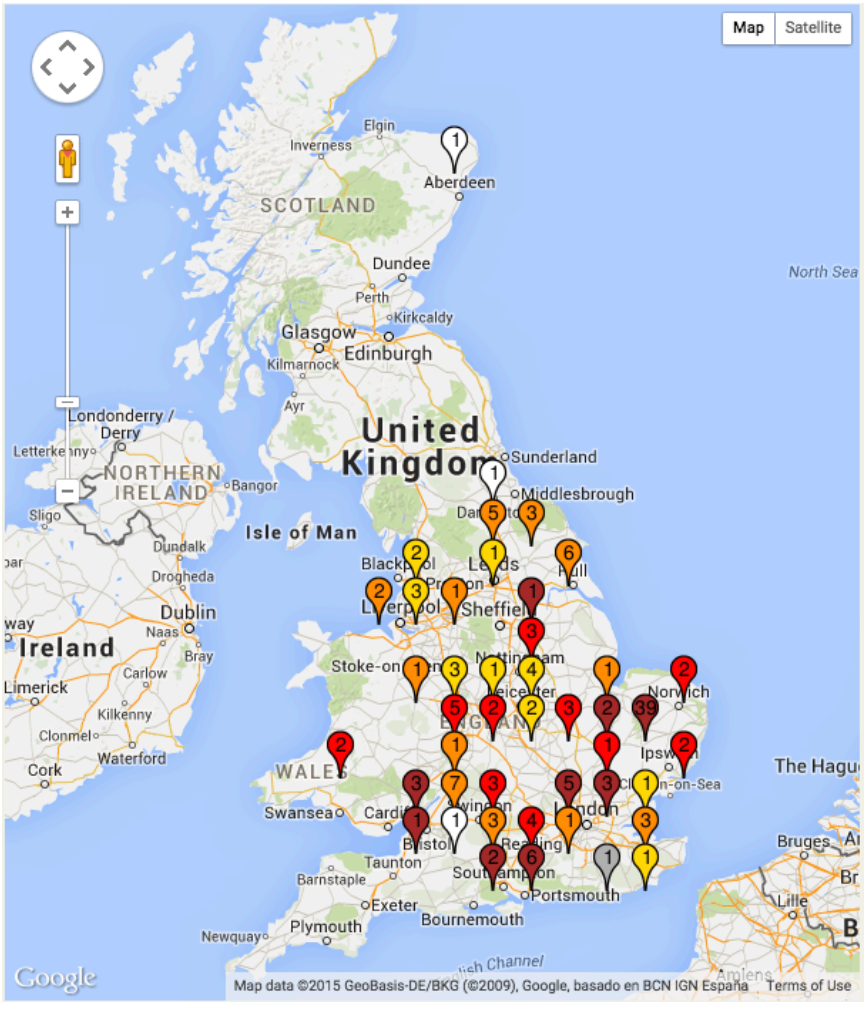


Figure 2.3: Leaf Watch results (2012) [66]

Like many observation type research, the damage scores were opinion-based and prone to bias as a result [22]. An alternative method to quantise the amount of damage is by calculating the percentage of area infested with leaf mines using image segmentation techniques, which will be elaborated in chapter 3. Table 2.2 shows a grading system for grey leaf spot disease on soybean plantation which adopted a similar method [27].

Regardless of the user-influenced classification method, Leaf Watch is a good example of how digital image processing techniques can be used on *C. ohridella.*

Disease level	Ratio of lesion to leaf area
T0	$t = 0$
T1	$0 < t \leq 0.01$
T2	$0.01 < t \leq 0.05$
T3	$0.05 < t \leq 0.2$
T4	$0.2 < t \leq 0.5$
T5	$0.5 < t \leq 1$

Table 2.2: Rank table of soybean leaf spot disease [27]

2.2 Introduction to leaf-mining insects and tar spots

Larvae of insects belonged to the orders of moths (Lepidoptera), flies (Diptera) and beetles (Coleoptera), or the suborders of sawflies (Symphyta) and wasps (Apocrita) often begin their lives as eggs on the surface of leaves. As the larvae hatch, they start to feed on leaf tissues of their host plants. Subsequently, the larvae are protected from predators and are less susceptible to insecticides.

Leaf miners cause damages to ornamental and horticultural plants alike, from lowering their aesthetical value to reducing crop production. As such, leaf-mining species are regarded as pests that cause significant economic impact [2] [69]. Therefore, the proposed method for monitoring their growth would be beneficial for the taxonomists who focus on defining and identifying the leaf miners, the

entomologists who are studying the behaviour of leaf-mining species and their relationship with the host plants, and the farmers who wish to control the infestation among their crops.

2.2.1 Horse chestnut leaf miner

Cameraria ohridella (Lepidoptera: Gracillariidae), more commonly known as the horse chestnut leaf miner, as the name would suggest mainly feeds on the common horse chestnut *Aesculus hippocastanum*. Figure 2.4 (a) shows an adult *C. ohridella*. One of its earliest sighting is believed to be near Lake Ohrid, Macedonia in 1984 [70]; since then, the population of *C. ohridella* has sprung across Central and Western Europe. Although its origin remains debatable [71], its expansion over the past three decades has been described by Valade *et al.* [72] as "explosive, progressive and highly invasive", while the potential of infestation within or outside Europe was acknowledged by Lees *et al.* [71].

The leaf mines start to appear in late Spring every year, in correspondence to the three main flight peaks of adult *C. ohridella*: between mid-April and early June, from late June to early August, and from mid-August to mid-September), but adult moths can appear till early October [73] [74]. The rate of parasitism does not increase with successive generations within the same year [73]. Later generations may have restricted area to feed on because of larvae from earlier generations on the same leaf, leading to a change in feeding pattern in terms of shape, area, height and width, etc. Generally favouring branches of low or medium height under shady areas and avoiding the canopy, female adult moths each can lay anywhere from 20 to around 80 eggs on the upper surfaces of leaves [75] [76] [77]. Figures 2.5 are examples of *C. ohridella* leaf mines on horse chestnut leaves.

Depending on environmental factors, eggs hatch within 4 to 21 days [73] [76], with pupa stage lasting for 12 to 20 days during summer generations [76]. Figure 2.4 (b) shows an egg on the surface of a horse chestnut leaf. As the larva feeds on saps through four instars (and an optional fifth), a mine is gradually formed on the upper surface of leaves. During the cylindrical feeding instar, the circular reddish-brown mine expands by about 4 to 7 square centimetres along the main

veins of the leaf [75]. Additionally, there are two non-feeding spinning stages, and a cocoon may not be necessary in earlier generations [76] [77] [78]. After 20 to 45 days, the larva emerges as an adult moth, which lives for a few days. The complete life cycle of *C. ohridella* is between 6 to 11 weeks [76], repeating one to four times per year. Figure 2.6 shows the leaf mines in various stages.

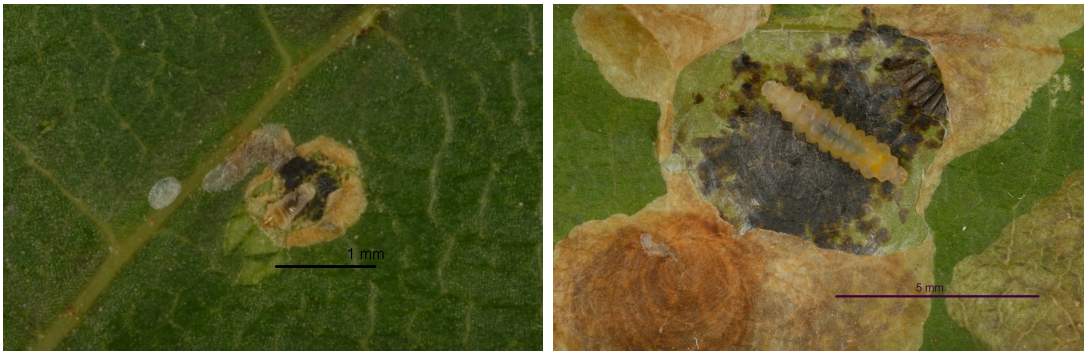
(a) *(b)*

Figure 2.4: C. ohridella in different life stages

(a) Adult moth at resting position [79]
(b) Ag egg, on the upper surface of a horse chestnut leaf [80]

(a) *(b)*

Figure 2.5: Infested Aesculus hippocastanum leaves



(a) *(b)*

Figure 2.6: Growth of larvae

(a) First larval instar, sap feeding with empty eggshell and the beginning of the leaf mine [81]

(b) Cameraria ohridella second spinning larval instar [82]

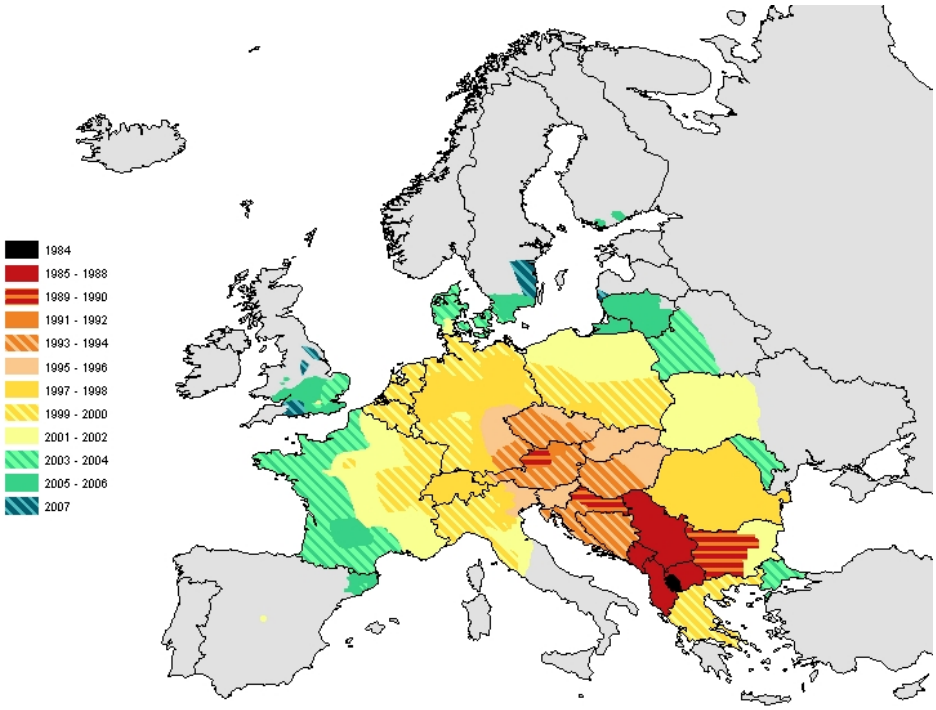


Figure 2.7: Incremental distribution of Cameraria ohridella, 1984 – 2007 [83]

Valade *et al.* [72] suggested the likelihood of moths originating from natural stands of *A. hippocastanum*; with population of *C. ohridella* having higher genetic diversity sampled from natural habitats, compared to planted trees. Given its main host plant, *A. hippocastanum*, has been a popular choice of ornamental tree for over a century, and that human-related activities are generally believed to encourage long-distance migration [72], mines of *C. ohridella* has been observed in over 30 European countries, as shown in the distribution map in figure 2.7.

The damages caused by *C. ohridella* are believed to be of mainly aesthetical value, and long term heavy infestation are not considered detrimental to the plants' general health [84] [85] [86] [87]. Damage that occurs later in the season tends to impact the performance of plants [88]. Percival *et al.* suggested *C. ohridella* might pose long-term harm on the reproductive capacity of infested trees, due to a 16 – 98% loss in photosynthetic energy, caused by premature leaf loss due to damages caused by leaf mines [89]. The infestation of *C. ohridella* can also cause the production of seeds to diminish by up to 48% and halving each of their weight. In Albania, where horse chestnut is classified as an endangered species in the IUCN plant red list [90], further infestation of *C. ohridella* could case potential threat to biodiversity in the area.

The discovery of ovipositing on other *Aesculus* and *Acer* plants, the infestation of the leaf-mining moths can potentially pose a renewed threat to the *Acer* genus across Europe. Aside from its primary host plant, the common horse chestnut *Aesculus hippocastanum*, larvae of *C. ohridella* also have the ability to feed on other horse chestnut and maple trees, especially in areas where the common horse chestnut plants are heavily infested [15] [16] [17].

This has been observed in areas where the infestation of horse chestnut plants is particularly prevalent. For example, A larva of *C. ohridella* is able to feed and fully develop on *A. turbinata* (Japanese horse chestnut) [15] [16], *A. octandra* (American species yellow buckeye), *A. glabra* (Ohio buckeye), *A. sylvatica* (painted buckeye), *A. pavia* (red buckeye), *A. platanoides* (Norway maple) and *Acer pseudoplatanus* (sycamore maple) [15] [17] [91]. An example of *C. ohridella* infestation on *A. pseudoplatanus* is shown in figure 2.8. Other minor host plants include common red horse chestnut (*Aesculus* × *carnea*), Indian buckeye (*A.*

indica), Chinese buckeye (*A. chinensis*), California buckeye (*A. californica*); or occasionally plants outside plant genera *Aesculus* and *Acer*; they are non-conductive to larval development and not seriously affected by *C. ohridella* [91].

However not all species that *C. ohridella* lay eggs on support larval development that would lead to damage of the plant. Certain *Aesculus* species are potentially more resistant to the leaf-mining insects, including, such as the genotype *A. hippocastanum L. Mertelik06* (M06) [92]. It was suggested that the susceptibility and resistance of *C. ohridella* among the genus *Aseculus* varies, in relation to their genotypes [93] [94]. Of the 13 species of *Aesculus* native to the northern hemisphere, *A. hippocastum* is the most susceptible to *C. ohridella*, followed by *A. turbinata* and *A. glabra*, with *A. flava* (yellow buckeye), *A. pavia* and *A. sylvatica* being less likely to be affected [93] [95].

Occasionally appearing alongside leaf mines caused by *C. ohridella* is an infection caused by the fungus *Guignardia aesculi*. This disease causes irregular brown and red lesions on the surfaces of the leaves, leading to the common name of horse chestnut leaf blotch, as shown in figure 2.9. At a glance the blotches may be mistaken for leaf mines caused by *C. ohridella*, yet they are not see-through when the leaf is held up to a light source (*C. ohridella* leaf mines on the other hand are see-through). The blotches appear as water-soaked spots in the beginning, before turning reddish-brown with yellow outlines which merge into the leaf tissue. The lesions are concentrated on the tips or edges on the leaves, and the larger spots can cause the leaves to distort or even drop prematurely [96] [97].

The pathogen is believed to be originating in North America and was introduced by accident into the United Kingdom and across Europe. Spores of the fungus emerge around April every spring and infect new leaves; and by May black spots scattered within the lesions are produced by the fruiting body pycnidia, which spread the disease throughout the season. Humidity favours the pathogen and the infection can be exacerbated by poor air circulation. The blotches are considered a primarily aesthetic issue which are not necessarily harmful to the health of the infected plants. Chemical management is rarely used to control the pathogen, unless tree growth is hindered by severe infections; removal of fallen leaves and improvements on air circulation will help reduce future infections.

Figure 2.8: C. ohridella mines on sycamore maple (Acer pseudoplatanus) [98]

Figure 2.9: Horse chestnut leaf blotches [99]

Jagiełło *et al.* proposed a method to access the total amount of foliage damage caused by both *C. ohridella* and *G. aesculi* on horse chestnut saplings [100]. The saplings were planted specifically for the experiment, and they were photographed overhead over two months. The extent of damage was based on the classification among healthy, damage, and background areas. Jagiełło *et al.* suggested that a leaf mine infestation and fungal infection trigger similar defence response in the hose plants, but a different defence would be elicited when *C. ohridella* and *G. aesculi* coexist on the same plant [100].

Currently due to lack of samples available, the horse chestnut leaf blotch, as well as *C. ohridella* mines on secondary host plants are not considered in this project, which focuses on *C. ohridella* mines exclusively on horse chestnut leaves.

2.2.2 Apple leaf miner

Lyonetia clerkella (Lepidoptera: Lyonetiidae), also known as the apple leaf miner, is a species of moth commonly found across Europe, as well as in Siberia, the Far East, northern Africa, the Middle East, Turkey and certain parts of Asia [101] [102]. *L. clerkella* mainly preys on fruit trees from in the *Betulaceae* (birch) and *Rosaceae* (rose) families [103], the latter includes *Malus*, more commonly known as apples (*Malus pumila*). It also affects *Prunus* (plum) plants, including *Prunus avium* (wild cherry); and occasionally *Crataegus* (hawthorn) [104].

Similar to other leaf-mining insects, *L. clerkella* does not cause significant damage to the health of plant, although it can dramatically decrease the aesthetic value of the infested trees, which are often planted as ornaments. In addition, as larvae tunnel between upper and lower epidermis of leaves, this can lead to defoliation and a loss of yields, especially in saplings and young trees. Pesticides are not considered useful pest control for *L. clerkella* [105].

Unlike the blotch mines of *C. ohridella*, the long, narrow and smoothly-curved mines of *L. clerkella* winds through the leaf, not impeded by leaf margin or venation [106] [107] [108], as shown in figure 2.10 (b). Each egg is laid with an ovipositor within the leaf cuticle, leaving a small scar without any visible egg shell. The initially transparent mine turns green and feeds on the leaf tissue,

forming a whitish mine, broken up with brown or black frass in the centre [106] [109]. Sections of the leaf can be cut off and die due to the mine crossing itself and throughout the leaf. When a larva finishes feeding, it emerges from the upper epidermis. A small silk cocoon is then spun by the maturing larva which suspends from the underside of the leaf, from which pupation takes place [103].

Believed to be nocturnal, adult moths are silvery or white in colour (an example of which is shown in figure 2.10 (a), with an average wingspan of 7 – 9 mm [110]. Up to three generations of apple leaf miners appear every summer, with the last generation (from August to early October) being particularly noticeable with the huge number of caterpillars. Severe infestation can be cause parts of the leaves to die and fall out. During late Summer apple trees can be heavily infested, but can be prevented with the removal and destruction of damaged leaves. A map of distribution for *L. clerkella* leaf mines in the United Kingdom is shown in figure 2.11.

(a) *(b)*

Figure 2.10:
(a) Adult Lyonetia clerkella at resting position [110]
(b) Leaf mine of L. clerkella on Malus pumila [106]

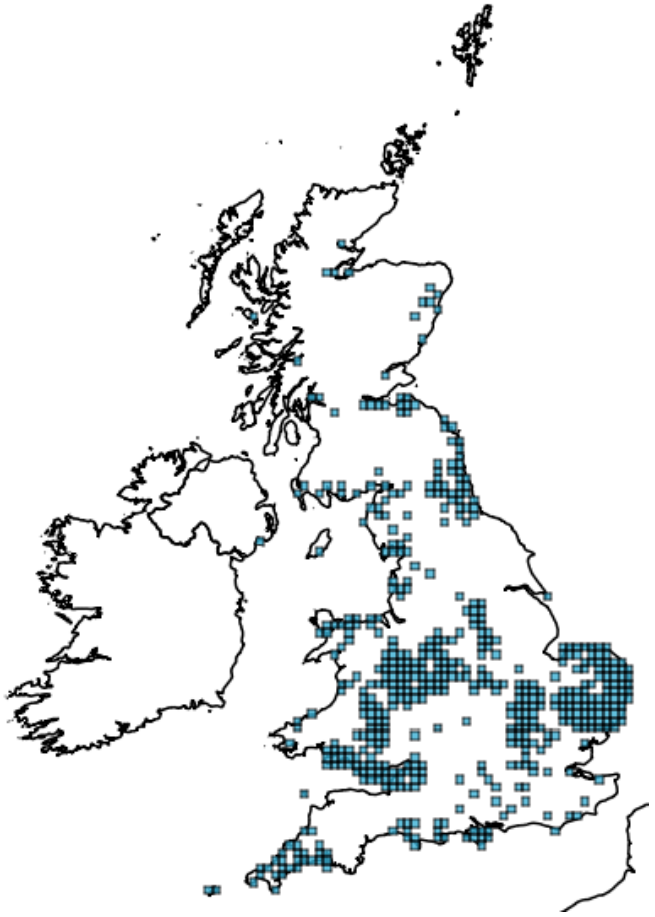


Figure 2.11: Distribution map for L. clerkella in the United Kingdom [111]

2.2.3 Holly leaf miner

Unlike the horse chestnut and apple leaf miners, holly leaf miners, *Phytomyza ilicis* (Diptera: Agromyzidae), belongs to the order of *diptera* (flies). First described by de Meijere in 1926, the holly leaf miner, as its name would suggest, feeds mainly on holly plants from the genus *Ilex* [112]. It is widespread throughout Europe but can also be found in North America.

There is only one generation per year, with up to three mines of a single leaf. Example of *P. ilicis* leaf mines are shown in figures 2.12 (a) and (b). Adult flies laying eggs as an ovipositor, at the base of petiole or midrib on the underside of holly leaves in June [113] [114]. As it hatches, the larva slowly feeds on and

tunnels in the midrib towards tip throughout the months of September to November, creating a yellowish linear blotch mine in an irregular shape [114]. The larva continues to mature, with its first moulting in December or January, and reaching a maximum size in March. During the last larval stage from March to May, it pupates within the cuticle [115]. Before its pupation, the larva makes a larger block mine in the parenchyma of the leaf. Dissimilar to the original mine, it does not have frass in its centre and has no discolouration at all [113]. *P. ilicis* in different life stages are shown in figure 2.13.

Mines of *P. ilicis*, and therefore the holly leaves they are on, are prone to attacks by the Eurasian blue tit *Cyanistes caeruleus*, as well as a number of parasitic wasps. These attacks prevent the larvae from maturing into adults [116]. *C. caeruleus* opens a holly leaf by tearing with its beak, removing the larva in the process; the parasitic wasp *Chrysocharis gemma*, on the other hand, lays eggs through the cuticle of the leaf into the body of a larva, before the parasite feeds on the leaf miner and kills it; while *Sphegigaster flavicornis* attacks the pupa of a young leaf miner by laying eggs within the host pupa and feeding on it. A third species of wasps, *Pleurotropis amyntas*, though less common, targets both the pupa of *P. ilicis* and larvae of other parasitic wasps [116]. Though it lowers the aesthetical value of infested plants, infestation of *P. ilicis* has little impact on the trees' health. Removal of affected leaves help control the damage but could lead to significant defoliation, which in turn is more damaging. Pesticides are not considered effective as they cannot penetrate the thick, glossy surface holly leaves and would simply run off the foliage.

(a) *(b)*

Figure 2.12: (a) – (b) Leaf mines of P. ilicis on Ilex aquifolium [112] [119]

Figure 2.13: Phytomyza ilicis in different life stages: (a) adult & (b) larva [117] [118]

2.2.4 Tar spots

Rhytisma acerinum (Rhytismatales: Rhytismataceae) is a plant pathogen which mainly affects tree species in the genus *Acer*, more commonly known as maple [120]. Norway maple (*Acer platanoides*), silver maple (*Acer saccharinum*), sycamore maple (*Acer pseudoplatanus*) are often affected by the ascomycete fungus [121] [122]. *R. acerinum*, the teleomorph of the fungus lives parasitically on leaves as a local infection. *Melasmia acerina* is the anamorph of the pathogen, which favours young trees in humid environments.

The fungus leaves yellow chlorotic spots on tree leaves in late spring and carry on evolving into black lesions with yellow borders on the upper epidermis. These spots are also known as stromata, and they continue to develop until the end of summer of early autumn [123]. Its brown-black appearance leads to the more commonly-known name of "tar spots". Example of tar spots can be seen in figure 2.14. On Norway maples, the spots are around 15 mm in diameters [123]. Infected leaves will fall and overwinter, with the fungus producing needle-liked spores internally. As spring approaches, the spores are ejected from the fallen leaves, before being carried to new host plants by wind, thus starting a new cycle.

Although heavy infection often decreases their aesthetic value, the tar spots usually does not have adverse effects on the long-term health of the host plants,

as tar spots do not interfere the growth of buds. An accepted management method is to remove debris of fallen leaves bearing the spots during the winter months, before the fungus matures. This practice may reduce the reoccurrence of the disease in the following spring, but its effectiveness is somewhat dubious, since spores from other trees could easily be transmitted by wind.

Figure 2.14: Rhytisma acerinum on Acer pseudoplatanus

2.3 Summary

This chapter provided an overview of related research in the development of automated identification of plant diseases using an image-based approach. Some of the digital image processing techniques and their use of neural networks as classifiers. Existing research on automatic species identification of leaf mines using digital image processing is limited; however, the techniques used in other leaf-based CAT systems are transferable because of the similarities in features such as colours and sizes. Similarly, the use of neural networks and other techniques from pattern recognition can be adapted to identify leaf mines.

This chapter also detailed background information on the life cycles of horse chestnut leaf miner (*Cameraria ohridella*), apple leaf miners (*Lyonetia clerkella*),

and holly leaf miners (*Phytomyza ilicis*). The three species of leaf-mining insects described in this chapter all have distinct characteristics in terms of shape and appearance, which provide challenges for the detection of leaf mines. The mines of *L. clerkella*, unlike the blotch mines of *C. ohridella* and *P. ilicis*, are long and thin. *P. ilicis* leaf mines, meanwhile, are located on the glossy leaves of holly plants. Holly leaves are also wavy and has spiny edges, which tend to affect the image acquisition process.

It was noted that infestation of the so-called horse chestnut leaf blotch (*Guignardia aesculi*) sometimes occurs alongside leaf mines caused by *C. ohridella*, and can be mistaken as each other due to their similar appearances. However, this is not explored further in this thesis as samples are not available. The plant fungus (*Rhytisma acerinum*) was also discussed in this chapter, as the fungus was introduced to explore the possibility of adapting the proposed approach for detecting fungal spots in additional to leaf mines.

Chapter 3

Digital image processing

Digital image processing can be defined as "the processing of digital images by means of a digital computer" [3, pp. 1 – 3]. One of its goals is to enhance objects of interest, while diminishing the less desirable features and background noises [3, pp. 689 – 690]. Image processing and image analysis are closely related to the fields of pattern recognition and artificial intelligences (AI), alongside the interdisciplinary area of computer vision [3, pp. 1 – 3] [124, pp. 291 – 292]. The fundamental steps of digital image processing are illustrated in figure 3.1.

Many image-based computer-aided taxonomy systems incorporate a range of techniques, including image acquisition, restoration, enhancement, compression, representation, segmentation, description and object recognition [124, pp. 1 – 2]. The objective of this chapter is to discuss the scope of the field that is image processing, focusing on the basic techniques related to this thesis, such as image segmentation, which includes the techniques used to divide an image into constituent parts, and the application of morphological algorithms to extract useful information from digital images.

3.1 Colour models

Digital imaging can be divided into the following categories – binary, greyscale, colour, volume, range, and multispectral [125, pp. 20 – 22]. For the purpose of this thesis, only binary, greyscale and colour images are further discussed. An example of a colour image is shown in figure 3.2 (c), while (a) and (b) are the result of binary and greyscale transformation of (c), as performed in MATLAB.

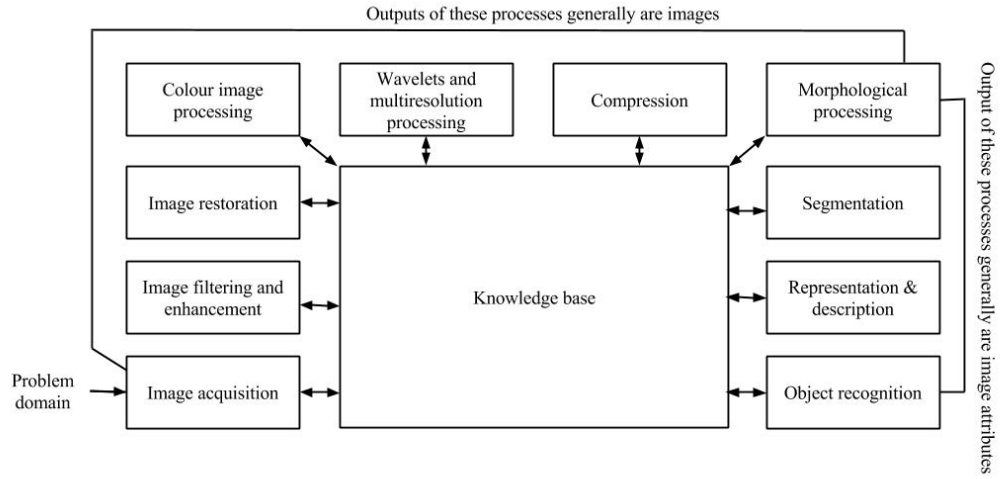
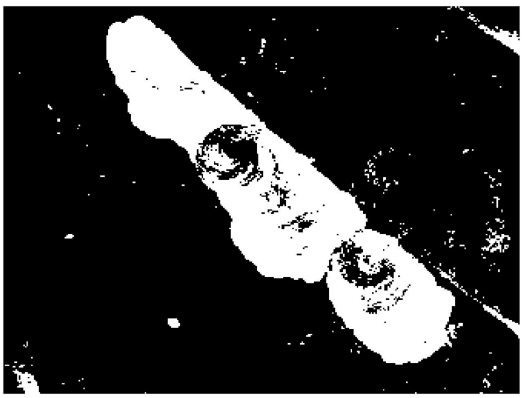


Figure 3.1: Fundamental steps of digital image processing [3, pp. 25 – 28]

Pixels in binary images take only two values – "0" (black) and "1" (white), making this format suitable for simple processing and extracting geometric properties from objects. As for greyscale images, each pixel value corresponds to the light intensity, and the brightness graduation can be differentiated, unlike in binary images. An 8-bit image has a brightness variation from 0 (black) to 255 (white).

Colour image processing, on the other hand, consists of two predominant areas: pseudocolour and full-colour processing [3, pp. 394 – 395]. The basic principle of pseudocolour image processing is taking monochrome images and assigning colours to grey values based on a specified criterion [3, pp. 414 – 415]. The primary application of pseudocolour, or false colour, is for human visualisation and the analysis of greyscale events in images. Pseudocolour images are often used in multispectral image processing and in remote sensing to aid the process of feature extraction. An example of a colour image of a *C. ohridella* leaf mine is shown in figure 3.3 (a), while (b) and (c) are the result of greyscale and pseudocolour transformation of (a).

As for full-colour processing, one of the major reasons of using colour images is that humans can generally differentiate thousands of colours and intensities, compared to a few dozen shades of grey [3, pp. 414 – 415]. Full-colour processing handles tasks by working with colour pixels directly as vectors, or processing as individual component images (full-colour images have at least three components, which can be modelled as separate monochrome images) [3, pp. 424 – 426].



(a)

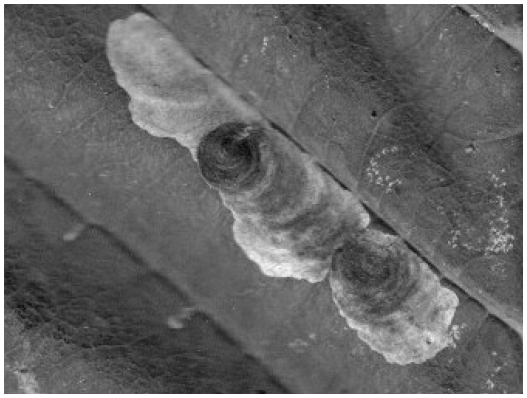
(b)

(c)

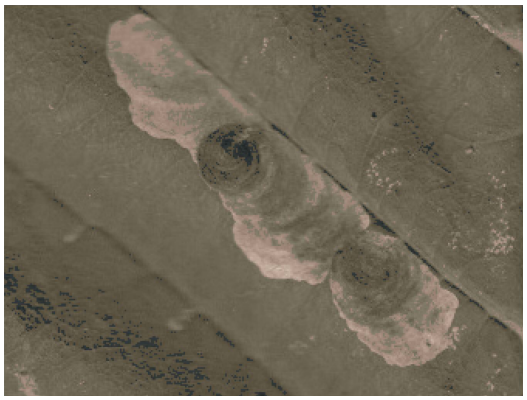
Figure 3.2: (a) Binary, (b) greyscale, and (c) colour (original) image

The original RGB image is a scanned picture of a horse chestnut leaf, cropped to focus on a single leaf mine.

(a)



(b)



(c)

Figure 3.3: (a) Full-colour, (b) greyscale, and (c) pseudocolour image

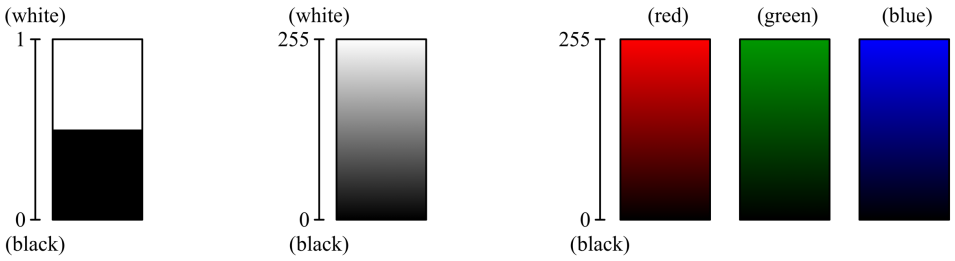
The original RGB image is a scanned picture of a horse chestnut leaf, cropped to focus on a single leaf mine.

Until relatively recently, full-colour images were mostly processed as individual component images due to the lack of availability of applicable sensors. This approach remains quite common in computer-aided taxonomy, as standard greyscale image processing methods can be applied to the component images individually, which then form a composite processed colour image [3, pp. 424 – 426]. Various representations of images are shown in figure 3.4.

Full-colour processing largely depends on the visible spectrum, referring to the segment of electromagnetic (EM) spectrum that can be detected by human eyes [3, pp. 395 – 401]. Although this is a fairly limited portion of the EM spectrum, the cones in human eyes are particularly sensitive to red, green and blue lights [3, pp. 395 – 401], thus the notion of primary colours is devised – red (R), green (G), and blue (B) are known as primary colours of light. The secondary colours, magenta (M), cyan (C), and yellow (Y), are produced by adding the primary colours; while white light is produced by mixing all the primary colours, or mixing a pair of opposition primary and secondary colours in the right intensities. The primary and secondary colours of light and pigments are illustrated in figure 3.5. These observations underline the definition of colour models.

Colour models, sometimes known as colour spaces or colour systems, are standardised ways in which colours can be represented systematically [3, pp. 401 – 402]. They are largely composed within a specification under a coordinate system, where every colour is allocated a single point in a subspace in that system. Most colour models are either hardware oriented, or are specially designed for colour manipulation; for example, the RGB (red, green and blue), CMY (cyan, magenta and yellow), and similar models are better suited for generating colours, rather than describing them. The HSI (hue, saturation and intensity) or HSV (hue, saturation and value) models, on the contrary, are ideal for developing image processing algorithms that effectively describe colours.

Compared to the RGB and its comparable models, the HSI model provides a practical description of colours in terms of how human eyes interpret colours – human eyes process colours according to the reflection of light when looking at a colour object, taking into account the hue, saturation and brightness [126], which is particularly important where subjective input from users is applicable.



(a) *(b)* *(c)*

Figure 3.4: Value at each pixel in (a) binary, (b) 8-bit greyscale, and (c) RGB colour images [125, pp. 20 – 23]

(a) *(b)*

Figure 3.5: Primary and secondary colours [3, pp. 395 – 400]

(a) Primary and secondary colours of lights (additive primaries)

(b) Primary and secondary colours of pigments (subtractive primaries)

3.1.1 RGB and CMY colour models

Based on a Cartesian coordinate system, each colour within the RGB colour model can be represented in its primary spectral components of red, green and blue [3, pp. 406 – 407]. Assuming all colour values are normalised to be in the range of [0, 1], colour C, can be represented by:

$$C = \alpha R + \beta G + \gamma B \tag{3.1}$$

where α, β and γ are coefficients, which illustrate proportions of the red, green and blue correspondingly. Consequently, each RGB image is comprised of three monochrome intensity images. When the coefficients add up to 1 (i.e. $\alpha R + \beta G + \gamma B = 1$), this equation can be referred to as a unit trichromatic equation [127].

The secondary colours of light can be created by combining two of the primary colours – cyan (blue and green), magenta (red and cyan) and yellow (red and green). An RGB to CMY conversion can be performed with:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.2}$$

Additionally, a fourth colour, black (K), is sometimes added to the CMY model, forming the CMYK colour model, which can be created by adding equal amounts of cyan, magenta and yellow.

While the RGB colour model is mostly used for sensing and representing digital image due to its compatibility, the CMY (cyan, magenta and yellow) model, on the other hand, is widely used in colour printers, photocopiers and other devices that deposit colour pigments on paper. Both models are ideal for hardware implementations, due to the availability of compatible devices.

3.1.2 HSI and HSV colour models

As mentioned previously, colour models that use cylindrical coordinates, such as the HSI (hue, saturation and intensity), are more similar to how human eyes

perceive colours [126]. The graphical and cylinder representations of the HSI colour model are shown in figures 3.6 (a) and (b).

Hue is the colour attribute that describes pure colour at their angular dimension (0 to 2π); saturation interprets the amount of pure colour diluted by white light, i.e. the amount of grey from 0% to 100%; value (or brightness) is an achromatic notion of intensity of the colour from 0% to 100% [3, pp. 418 – 424]. When considered collectively, hue and saturation can also be known as chromaticity; hence brightness and chromaticity together can characterise a colour, as shown in figure 3.7. Assuming the RGB values are normalised to the range [0, 1], and that the angle θ is measured with respect to the red axis of the HSI space, as depicted in figure 3.7, the hue H component of each RGB pixel is given by [3, pp. 407 – 414]:

$$H = \begin{cases} \theta & if\ B \leq G; \\ 360 - \theta & if\ B > G \end{cases} \quad \text{and} \tag{3.3}$$

$$\theta = \cos^{-1}\left\{\frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}}\right\} \tag{3.4}$$

The saturation S component can be obtained by:

$$S = 1 - \frac{3}{(R+G+B)}[\min(R,G,B)] \tag{3.5}$$

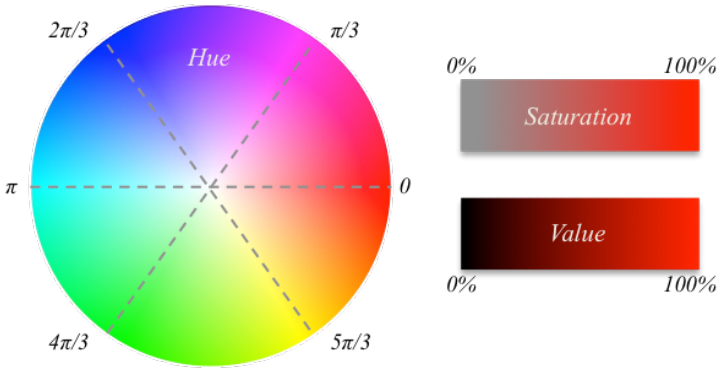
As RGB images are made up of three constitute intensity images (which represents red, green and blue), intensity can be extract directly:

$$V = \frac{1}{3}(R+G+B) \tag{3.6}$$

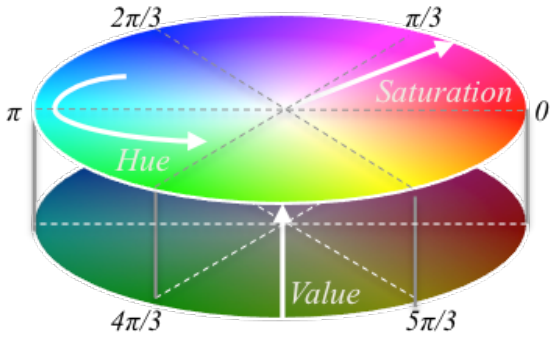
The conversion between RGB and HSI models is widely adopted in different applications of colour image processing. Particularly with images of leaves, the main reasons of choosing HSI colour model over RGB to improve accuracy of lesion region segmentation include [27]:

1. Illumination change – image quality can hugely depend on lighting, which is unlikely to be controllable when collecting data in a natural condition
2. Influence of vein – veins normally have shallower colouring than leaves, and the colour may change depending on stages of plant diseases
3. Unstable lesion characteristics – symptoms of leaf diseases are developed throughout different stages of plant diseases and rely on influences from lighting, water, nutrition and other environmental factors

Figure 3.8 (a) shows an image of an RGB image of a *C. ohridella* leaf. Figures 3.8 (b) through (d) show its hue, saturation, and intensity component images.



(a) Graphical explanation of hue, saturation and value



(b) Cylinder representation of the HSI colour model

Figure 3.6: Various representations of the HSI colour model

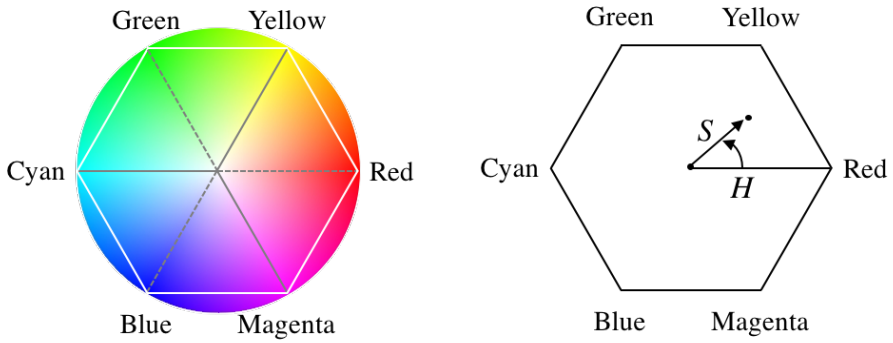


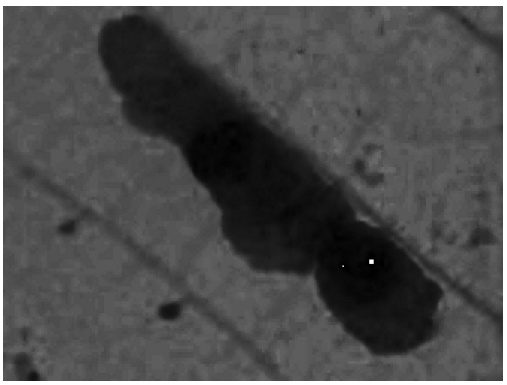
Figure 3.7: Hue and saturation in the HSI/HSV model [3, pp. 407 – 414]

Hue is given by the angle from the red axis,
and the length of the vector indicates the saturation.

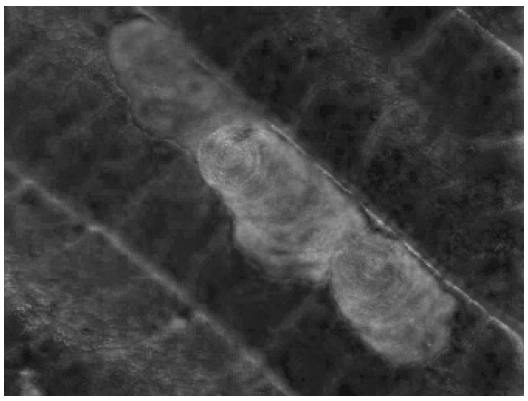
			RGB colour model (R, G, B)	HSI colour model (H, S, V)
Primary colours	Red		(1, 0, 0)	(0, 100%, 100%)
	Green		(0, 1, 0)	(2π/3, 100%, 100%)
	Blue		(0, 0, 1)	(4π/3, 100%, 100%)
Secondary colours	Cyan		(0, 1, 1)	(π, 100%, 100%)
	Yellow		(1, 1, 0)	(π/3, 100%, 100%)
	Magenta		(1, 0, 1)	(5π/3, 100%, 100%)
	Black		(0, 0, 0)	(0, 0%, 100%)
	White		(1, 1, 1)	(0, 0%, 100%)

Table 3.1: Colours in RGB and HSI representations

(a) Original RGB image



(b) Hue component image



(c) Saturation component image

(d) Intensity component image

Figure 3.8: Example of RGB and HSI component images

The original RGB image is a scanned picture of a horse chestnut leaf, cropped to focus on a single leaf mine.

3.1.3 Colour transformations

As discussed in the introduction to section 3.1, full-colour image processing fall into two main categories – by processing each component image individually, and dealing with colour pixels directly as vectors, with the latter being more common due to the relatively low cost in comparison. For the results of individual colour component and directly processing colour pixels as vectors to be equivalent, all processes have to be applicable for both scalars and vectors; and it is necessary for the operations on each vector components to be independent of one another [3, pp. 424 – 426]. Full-colour images, including those in the RGB and HSI colour spaces, can be treated as individual component images. Earlier application of digital image processing often processed each RGB image as three monochrome intensity images (depicting red, green and blue respectively).

In principle, human eyes respond to colours more accurately as objects tend to "stand out"; nevertheless, means of procuring and storing full-colour images tend to be more expensive. The amount of additional processing involved in interpreting the extra information carried by colour images and handling vector data can also be considered as a complication, due to the extra cost and longer processing time [129, pp. 21 – 22]. As such, colour images are often transformed into greyscale or binary images for further processing. Weeks *et al.*, for instance, discarded red and green components from the RGB images, before isolating the green component and transposing it to a greyscale image [130].

When processing images of leaves infested with mines and diseases, the leaf mines and tar spots can be considered as the foreground "objects", and the leaf itself can be treated as the background. These "mostly green pixels" do not contribute any significant information regarding the leaf mines or tar spots, they are regarded as irrelevant in further steps and can be removed. Discarding the green component of RGB images separates an image into regions according to their significance. The regions must be mutually exclusive, as each region is required to meet consistency conditions in specific regions [131].

Similarly, segmentation can be performed on HSI images – hue is the only colour attribute, as saturation and intensity lack extra information [23], the background

can be effectively removed by maximising contrast between veins and leaf surfaces, thus changing the disturbance of illuminance [27].

3.2 Image segmentation

Image segmentation forms the basis of image processing, and is usually an essential component of image analysis systems [132]. It is the process of subdividing an image into smaller components, so that regions of interest are identified or detected [3, pp. 689 – 690]. Different regions must not overlap each other and connected regions are heterogeneous [125, p. 369].

Many algorithms of image segmentation concern either the discontinuity or similarity of intensity values [3, p. 689 – 690]. The former approach (edge-based segmentation) partitions an image based on sudden changes in intensity, such as edges; while the latter (region-based segmentation) partitioned an image according to predefined criteria, which includes thresholding [3, pp. 689 – 690] [46].

3.2.1 Intensity thresholding

Intensity thresholding is an effective way to transform colour or greyscale images to their binary equivalents, by assigning pixels to two or more groups. It is particularly applicable for situations where the background on which the objects are is sufficiently constant in varying conditions. In computer vision and digital image processing, this is also referred to as Otsu's method.

Otsu's method is a typical approach for global thresholding, which selects a relevant threshold of grey level to separate objects from their background, in this case between the logical value of '1' and '0' in pixels. It is an optimum thresholding method which aims at reducing the error by maximising between-class variance. [3, pp. 742 – 747] [133].

A digital image can be represented by the two-dimensional function $f(x, y)$, where x and y are the spatial coordinates, and the amplitude of f at any pair of coordinates *(x, y)* is the intensity at that certain point [3, pp. 1 – 3]. It can also be defined as a discrete representation of data that contains both spatial (layout)

and intensity (colour) information [124, pp. 1 – 2]. In order to transform a colour or greyscale image to a binary one, a threshold level is required to determine whether the pixels are image foreground or background. A general rule of intensity slicing is, with two colours assigned to each side of a threshold, any pixel below the plane will be coded in one colour, while any pixel above the plane will be coded with the other colour.

A binary image is an image where each pixel is assigned to one of the two discrete logical values – '1' or '0'; hence an image with only two distinct colours could be created. Binary images carry no textual content; yet they are very useful for shape, size and location of the regions in the image. Solomon and Breckon referred to pixels with the logical value '1' as the *image foreground* pixels, and those having the logical value '0' as the *image background* pixels [124, pp. 197 – 198]. An *object* or a region in a binary image consists of a group of connected pixels – which means a foreground pixel must be connected to at least one neighbouring pixel to the north, east, south, or west to be considered as a pixel within the group.

Using Otsu's method, a full-colour or an intensity image can be transformed into a binary image using a threshold in the range [0, 1], by replacing all pixels with intensity greater than the threshold with '1' (white), and all other pixels with '0' (black) [134]. To create a binary image $b(x, y)$ from an intensity image $f(x, y)$, using a threshold value of T, the process can be represented as [124, pp. 265 – 266]:

$$b\,(x,y) = \begin{cases} 1 & if\ f(x,y) > T \\ 0 & if\ f(x,y) \leq T \end{cases} \tag{3.7}$$

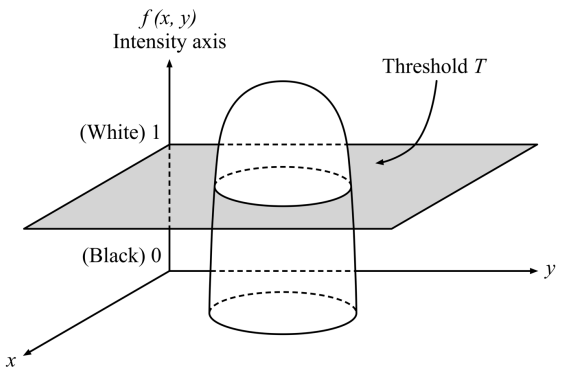


Figure 3.9: Geometric interpretation of intensity slicing [3, p. 415]

Figure 3.9 illustrates how intensity slicing can be interpreted as a 3D function: if greyscale, white and black can be given by $[0, L-1]$, $[f(x,y)=0]$ and $[f(x,y)=L-1]$ respectively; while the plane at $f(x, y) = l_i$ serves as the threshold in this case.

A frequently employed technique of intensity thresholding involves analysing the histogram of intensity levels in a digital image; if the image is composed of relatively uniformly illuminated objects and an evenly toned background, the minimum of distribution can be interpreted as an adequate threshold value [135]. For example, in figure 3.10, assuming the peak on the left of the idealised bimodal histogram corresponds to dark foreground objects, while the other peak is the result of the light background. The valley in between the peaks represents the relatively smaller number of pixels around the edge of the objects, from which the suitable threshold can be determined [129, pp. 81 – 83].

Otsu's method is also applicable for multilevel thresholding. This assigns pixels to more than two groups with a multimodal histogram. For an image $f(x,y)$ with two types of light object and a dark background, multilevel thresholding classifies a pixel (x,y) as belonging to the background if $f(x,y) \leq T_1$; and the two object classes if $T_1 < f(x,y) \leq T_2$ and $f(x,y) > T_2$ respectively. The segmented image can be given by:

$$g(x,y) = \begin{cases} a & if\ f(x,y) \leq T_1 \\ b & if\ T_1 < f(x,y) \leq T_2 \\ c & if\ f(x,y) > T_2 \end{cases} \tag{3.8}$$

where a, b, and c are three separate intensity values. Figure 3.11 shows an example of an intensity histogram that is partitioned by dual thresholds. The distances between the peaks in a multimodal histogram directly influence the accuracy of the thresholding (peaks that are further apart are more easily separated). Other factors that affects the accuracy of thresholding includes: the noise content; the size of the object(s) in relation to the background; the uniformity of the illumination; and the consistency of the reflective properties of the image [3, pp. 38 – 741]. Although theoretically Otsu's method can be extended to an arbitrary number of thresholds, it becomes less reliable as number of classes increases.

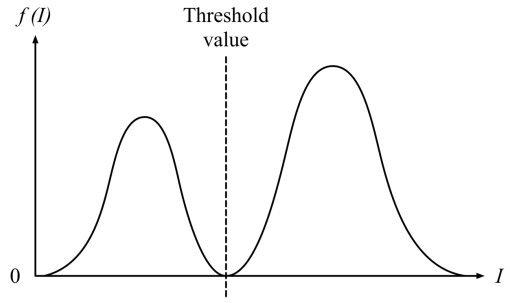


Figure 3.10: An idealised bimodal histogram of intensity levels in an image, which can be partitioned by a signal threshold

This histogram shows the number of pixels at each intensity value found in that image.

The pixel intensities are grouped in two separate clusters, which indicates foreground (left) and background (right) pixels respectively.

The optimum threshold for separating these two groups is between the two peaks, at the minimum of this distribution.

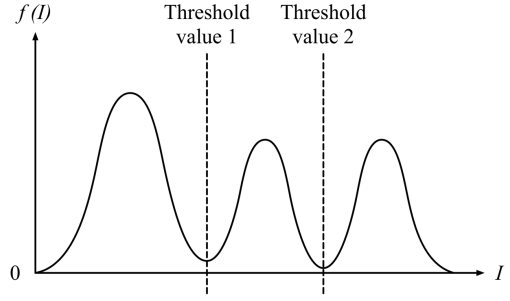


Figure 3.11: An idealised multimodal histogram of intensity levels in an image, which can be partitioned by dual thresholds

This histogram has three dominant modes, for instance, two dark objects against a light background.

Similar to the histogram in figure 3.10, the threshold values are given by the intensity values that separate the peaks.

When working with images of leaves, using basic thresholding technique with a single threshold, the intensity component of the HSI colour system can be transformed into binary images. The pixels are subdivided into "leaf" and "background" regions. For images with multiple leaf mines or tar spots, more than one thresholds may be needed.

In cases where a whole leaf is present, potentially the system can be refined to compare lesion area with regard to the leaf – it is possible to devise a corresponding rank table for damage caused, with a simple estimation of percentage of area infested with leaf mines or plant pathogens.

However, as difficulty in segmentation increases with the number of thresholds, other approaches may be more suited for partitioning an image into multiple classes. Liao *et al.* [136] suggested an improvement of Otsu's method to optimise thresholding of an image – using the maximised intraclass variance, a recursive algorithm was put forward to obtain the optimal threshold efficiently. When the intensity distribution of the background and objects are significantly noticeable, it is possible to apply a global threshold to the whole image. To improve its performance, image smoothing and edges are often used to improve global thresholding results.

3.2.2 Edge detection

Edges are often used in conjunction with other segmentation algorithms, such as thresholding and smoothing. Edge detection is based on sudden local changes in intensity, and is closely associated with extraction of boundaries [137]. Strictly speaking, the *boundary* of a finite region is a "global" concept, meaning the entire image is considered; and an *edge* is a "local" concept, meaning it is only a small portion of the image [3, pp. 68 – 71]. However, for binary images, edges and boundaries are sometimes interchangeable depending on the operator and type of connectivity. The three fundamental steps of edge detection are: (1) image smoothing for noise reduction; (2) detection of edge points – a local operation where potential edge points are identified and extracted; and (3) edge localisation – narrow down the candidate edge points to the true edge points [3, pp. 700 – 705].

In a binary image where objects are placed against a background, all pixels are assigned the number '1' or '0', for foreground and background pixels respectively. The fundamental principles to connecting the edges are: (1) the location of edge points locally, for instance within a 3×3 region; (2) the knowledge of points on the edge or boundary; and (3) a global approach that is suitable to the entire edge image [3, pp. 692 – 693].

Edges (or edge segments) are defined as sets of connected edge pixels, which are pixels where the intensity of an image $f(x, y)$ changes suddenly; and edge detectors are methods conceived specifically to identify edge pixels [3, pp. 692 – 738]. Ideally edges would be smooth and free of noise, yet blurriness and noise are inevitable in practice. The former depends on how the digital images are taken, such as the lenses of a digital camera; and the latter is limited by electronic components of the imaging systems used [3, pp. 700 – 705]. Figure 3.12 (a) shows the image extracted from figure 3.12 (b), which depicts a horizontal intensity profile. Figure 3.12 (b) shows the first- or second-order derivatives of the intensity. The intersection between the zero-intensity axis and the line between extrema of second derivatives is known as the zero-crossing of the second derivative [3, pp. 700 – 705].

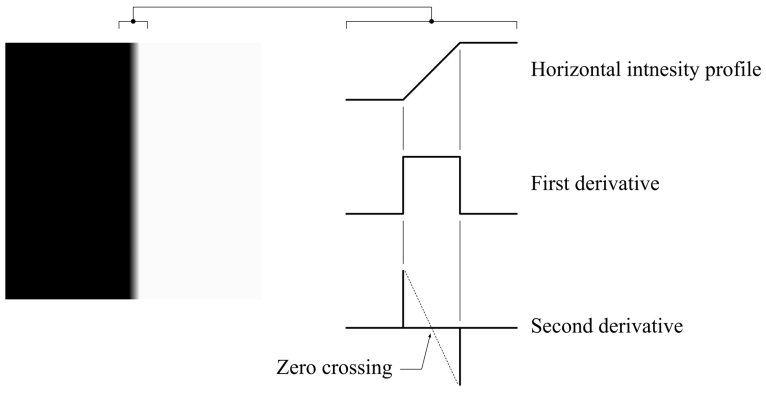


Figure 3.12: (a) An ideal vertical edge, and (b) its horizontal intensity profile [3, p, 703]

The following three operators use first-order derivatives to detect abrupt changes in intensity. A 3 × 3 region of an image is shown in figure 3.13. The Roberts cross-gradient operators are one of the earlier attempts to use 2-D masks with diagonal preference [138]. It uses two masks of size 2 × 2, as illustrated in figure 3.14 (a), when considering a size 3 × 3 region. The Prewitt operators, on the other hand, provides a more accurate approximation than the Roberts operators [139]. The entire image can be filtered with two masks of size 3 × 3, both with a weight of 1 in the centre coefficient, as shown in figure 3.14 (b). More complicated to implement than the Prewitt masks but superior in noise-suppression noises, the Sobel operators are often preferable when dealing with derivatives [3, pp. 706 – 714] [140]. They use two 3 × 3 masks with a weight of 2 in the centre coefficient instead of 1, but both operators, shown in figure 3.15, give equal results for vertical and horizontal edges. For diagonal directions, the 3 × 3 masks have to be modified to improve performance, as shown in figure 3.15.

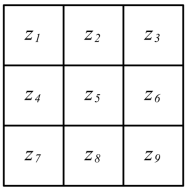
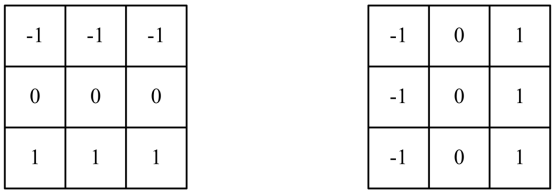


Figure 3.13: A 3 × 3 region of an image
(The z's indicates the intensity values)

(a)



(b)

-1	-2	1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

(c)

Figure 3.14: Various masks to compute the gradient at z_5 [3, p. 708]

(a) Robert operators

(b) Prewitt operators

(c) Sobel operators

0	1	1
-1	0	1
-1	-1	0

-1	-1	0
-1	0	1
0	1	1

(a)

0	1	2
-1	0	1
-2	-1	0

-2	-1	0
-1	0	1
0	1	2

(b)

Figure 3.15: The modified masks for detecting diagonal edges [3, p. 710]

(a) The Prewitt mask

(b) The Sobel mask

The Roberts, Prewitt and Sobel operators use one or more masks to filter an image. Though relatively simple to implement, they do not take into consideration the noise or edge characteristics. In an attempt to improve the above methods, Marr and Hildreth suggested that the change in intensity corresponds to image scale, hence edge detection calls for operators of different sizes; and that a rapid change in intensity would give a peak or trough in the first derivative, and a zero crossing in the second derivative, as seen in figure 3.16 [141]. Marr-Hildreth edge detector uses a Laplacian of a Gaussian (LoG) filter to find the zero crossings [3, pp. 714 – 725]. The LoG function can be expressed as:

$$\nabla^2 G(x,y) = \left[\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.9}$$

The Canny edge detector is the most complex of the techniques described in this chapter, but also has the best performance [142]. The three criteria of the Canny edge detector are (1) to lower the error rate; (2) ensure the edges are as close to the true edges as possible; and (3) only one point should be returned by the edge detector for each true edge point.

Many common edge detection methods are affected by slight alterations in illumination of neighbouring pixels, causing disruption to the continuity in intensity values. Maintaining connectivity is essential in edge detection, hence edge detection is routinely followed by mathematical morphology to link the edge pixels, so meaningful edges and boundaries can be assembled.

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

Figure 3.16: A 5 × 5 mask which is an approximation of the LoG function, also known as the Mexican hat operator because of its shape. In practice, the negative of the mask is used [3, p. 715].

3.3 Morphological image processing

Mathematical morphology is a useful tool when extracting components to represent the image or to describe its general shape [3, p. 630]. In terms of mathematical morphology, sets refer to objects in an image. For binary images, the sets are in a 2-dimensional space Z^2, where each element of the set refer to the (x, y) coordinates of a pixel; and for greyscale images, the sets are in a 3-dimensional space Z^2, where two components of each element corresponds to the coordinates, and the third refers to its intensity value [3, pp. 628 – 630].

Morphological operations are formulated based on structuring elements – which are small sets of binary-valued array or sub-image [3, pp. 628 – 630]. When the image undergoes erosion, every detail less significant than the structuring element is filtered from the image [3, pp. 631 – 633]; whilst dilation is the complete opposite, where the shape of structuring element acts as a guide to thicken objects in a binary image [3, pp. 633 – 635].

3.3.1 Basic morphological algorithms

Erosion and dilation are the primary components of morphological image processing, and are the basis of other morphological algorithms [3, p. 630]. The erosion of set A by structuring element B at all points z is defined as [3, pp. 631 – 633]:

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \tag{3.10}$$

Figure 3.17 shows an example of the erosion of A by B, which is the set of values of z that satisfy equation (3.10).

The dilation of set A by structuring element B is when $\hat{B}$ and A overlap by at least one element, which can also be represented by [3, pp. 633 – 635]:

$$A \oplus B = \left\{z \,\middle|\, \left[\left(\hat{B}\right)_z \cap A\right] \subseteq A\right\} \tag{3.11}$$

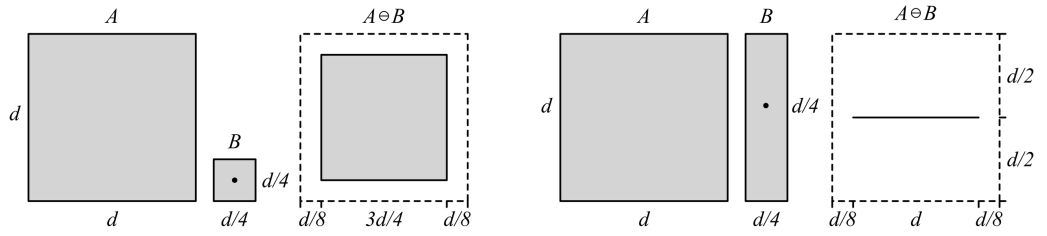
Figure 3.18 shows an example of the dilation of A by B. As illustrated by figures 3.17 and 3.18, erosion "filters" image details that are smaller than the structuring

element; while dilation does the opposite – it thickens objects, the manner of which is determined by the shape of the structuring element.

Erosion and dilation are duals of one another:

$$(A \ominus B)^c = A^c \oplus \hat{B} \quad \text{and} \quad (A \oplus B)^c = A^c \ominus \hat{B} \tag{3.12 – 13}$$

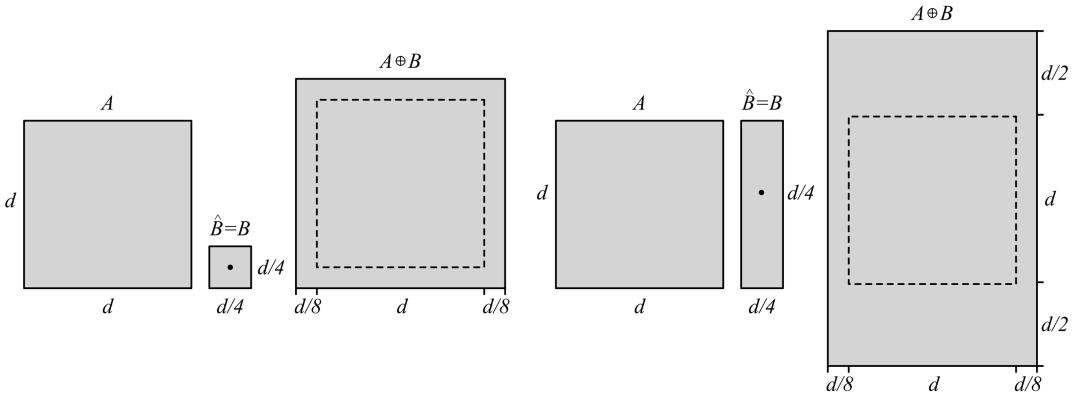
Equation (3.13) shows that the erosion of A by B is the complements of the dilation of A^c by $\hat{B}$, and vice versa.



(a) *(b)*

Figure 3.17: Erosion [3, p. 631]

(a) Set A, square structuring element B, and erosion of A by B (A⊖B)
(b) Set A, rectangular structuring element B, and erosion of A by B (A⊖B)



(a) *(b)*

Figure 3.18: Dilation [3, p. 634]

(a) Set A, square structuring element B, and dilation of A by B (A⊕B)
(b) Set A, rectangular structuring element B, and dilation of A by B (A⊕B)

3.3.2 Morphological boundary extraction

The algorithms discussed in this subsection are commonly used in conjunction with the image segmentation techniques in the previous section. When processing binary images, morphological algorithms are often for extracting boundaries and connecting components. An example of boundary detection is shown in figure 3.19. The boundary of set A, $\beta(A)$ can be extracted by the erosion of A, by structuring element B, and subtracting the result from set A [3, pp. 642 – 643]:

$$\beta(A) = A - (A \ominus B) \tag{3.14}$$

As mentioned previously, dilation bridges gaps between objects. Conditional dilation is frequently used in conjunction with boundary extraction to fill in holes, that area of background regions surrounded by connected borders of foreground pixels [3, pp. 643 – 645]. In binary images, all the pixels within the hole are replaced with '1's, indicating the area as foreground. For instance, if elements in set A are 8-connected boundaries, and each of these boundaries encloses a hole, which is of background pixels. Depicted in figure 3.20, the holes can be filled by [3, pp. 643 – 645]:

$$X_k = (X_{k-1} \oplus B) \cap A^c \qquad k = 1, 2, 3, \ldots \tag{3.15}$$

where X_0 is an array of 0s the same size as set A, and B is the symmetric structuring element. When $X_k = X_{k-1}$, the process stops at iteration k. $X_k \cup A$, therefore, contains the pixels indicated by the boundary and the filled holes.

When transforming a scanned image or digital photograph into a binary image, if the contrast between the objects on the image and their background is not significant enough, pixels within the object could be erroneously given the value of '1' and vice versa. When the object(s) is known to be the focus of an image, *a priori* knowledge indicates it should in the middle of the image, instead of nearing the borders. With the additional information of edges and boundaries, the false background pixels within the object can be reassigned to foreground with the hole filling method, and the false foreground pixels outside the object can be reassigned to background.

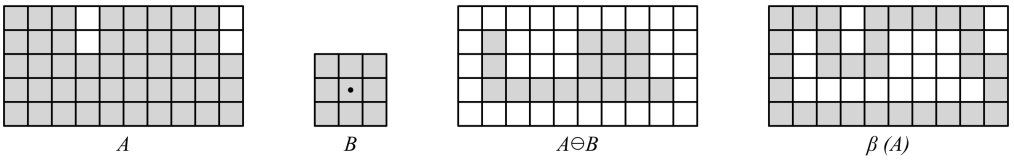


Figure 3.19: Boundary detection using erosion [3, p. 642]

(a) Set A, (b) structuring element B, (c) erosion of A by B, and (d) boundary given by the set difference between A and its erosion

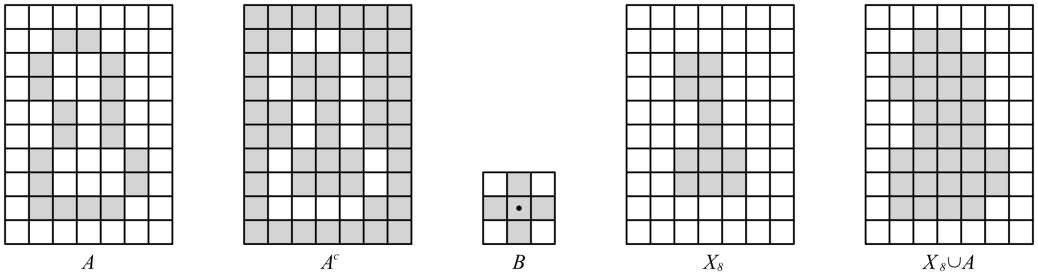


Figure 3.20: Hole filling [3, p. 644]

(a) Set A (shaded area), (b) complement of A, (c) structuring element B, (d) set k after the 8th iteration, (e) final result – the union of (a) and (d)

Chen *et al.* proposed a method to extract the boundary of objects in binary images that can be adapted to improve the detection of leaf mines [143]. Although this approach was originally devised for extracting liver boundaries from computer tomography (CT) images, computer-aided diagnostic (CAD) tools are the counterparts of computer-aided taxonomy within the medical field, and this technique can be applied to images of leaf mines and plant diseases with slight modification.

The detect-before-extract (DBE) method devised by Chen *et al.* identifies all boundary points of the liver based on the area bounded by '1's before applying a

Catmull-Rom B-spline to generate an initial liver contour. Dilation and erosion are used before interpolation to refine the initial contour. The structuring elements proposed by Chen *et al.* are included in figures 3.21 (a) through (f).

This approach further narrows down regions of interest by minimising inclusion of unwanted objects such as nearby organs (e.g. kidney), gastrointestinal tract, spinal cord, etc. It uses *a priori* knowledge to eliminate cornering regions. Using average grey-level (AG) and average feature (AF) maps formed using information contained within the remaining regions, modifications can be made to the initial contour to improve accuracy. Chen *et al.* suggested the correction be done iteratively using a deformable contour model. The model is subject to a set of constraints based on the differences of grey level and feature value of a particular pixel and average correspondence of a pixel from the normal liver region [143].

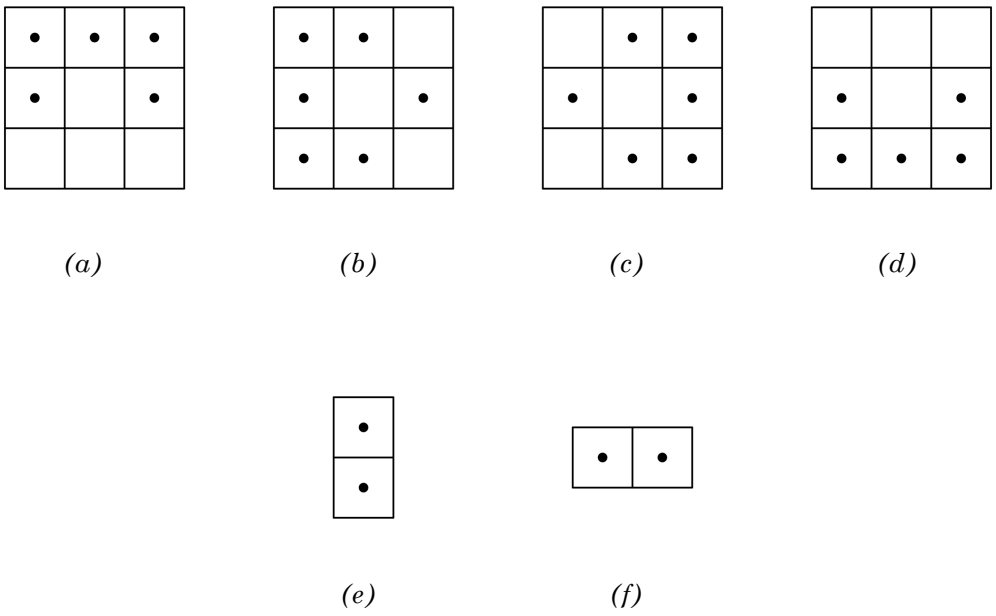


Figure 3.21: (a) – (d) Chen et al. proposed this set of structuring elements of morphological opening operations [142]

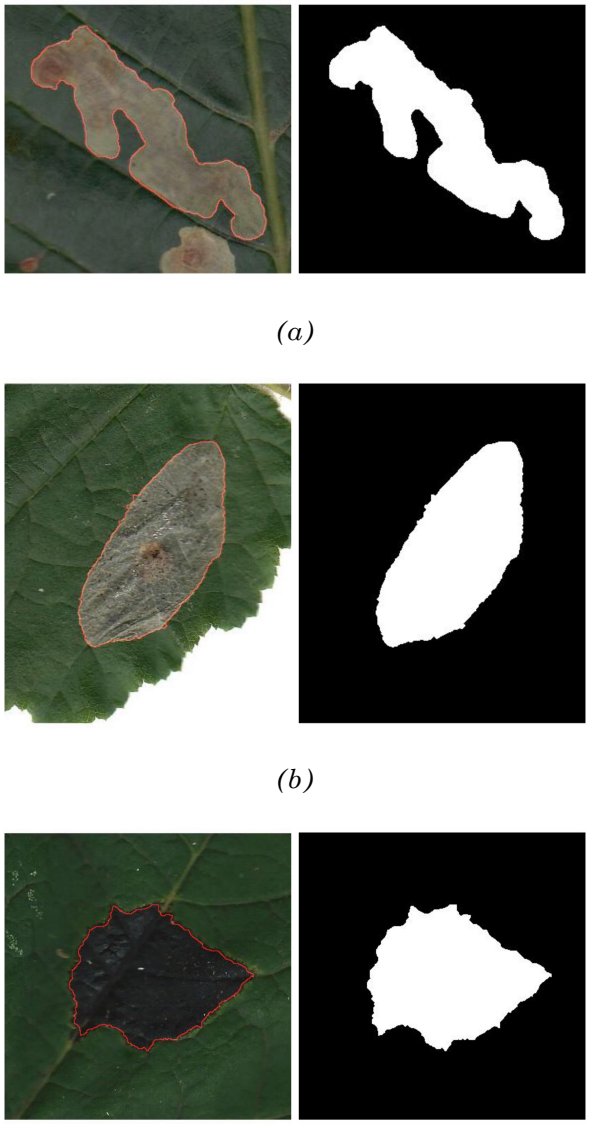
(e) and (f) are the structuring elements of dilation and erosion operations

A similar approach can be applied to binary images to leaf mines – objects (i.e. leaf mines or tar spots) are bounded by '1's, and morphological processes can be used before interpolation to refine the initial boundary. A pixel indicating the boundary, B, should satisfy the condition that any neighbourhood centred at B must contain at least a '1' and a '0', except when in a hollow within the object. Similarly, regions of interest can be slimmed down by minimising inclusion of unwanted objects, in the case of leaf mines, veins, minor blemishes on leaves or issues with the scanning process (e.g. dust trapped between the leaf and scanner glass). Diving the binary into $M \times N$ regions, as a result of *a priori* knowledge, it is only logical for the object (leaf mine or tar spot) to be roughly in the centre, meaning cornering regions can be eliminated.

3.4 Summary

This chapter elaborates on the basic image processing techniques that were used in the proposed system for detecting leaf mines and tar spots, including colour image processing, image segmentation using intensity thresholding, edge detection, and various morphological operations. Figure 3.22 shows the result of the various image segmentation techniques described in this chapter. Firstly, the RGB images were converted into HSI; then transformed into binary images using intensity thresholding. The edges of the objects in each image are obtained using the Sobel operators. The Sobel masks were chosen over the Roberts and Prewitt operators due to their noise-suppression qualities. The images on the left are scanned images of leaves of horse chestnut, crab apple and sycamore plants, which were cropped accordingly to focus of a single leaf mine or tar spot in each image. The edges are superimposed on the original RGB images, shown by the red lines. Their corresponding binary images, with the pixels within the edge filled using dilation, are shown on the right.

The implementation of these techniques will be discussed in chapter 5. The relationship among colour models, as well as that between full-colour and greyscale images, is essential for the first steps of the proposed semi-automated identification system. Image segmentation and edge detection are fundamental elements in digital image processing, and the techniques introduced in this chapter represents the bases of the approach developed in the following chapters.



(a)

(b)

(c)

Figure 3.22: Edge of leaf mines overlaid with their original RGB images and corresponding binary images

(a) C. ohridella on a horse chestnut leaf
(b) Phyllonorycter on a crab apple leaf
(c) R. acerinum on a sycamore leaf

The original RGB images are scanned pictures of (a) horse chestnut, (b) crab apple and (c) sycamore leaves respectively, which are then cropped accordingly to focus on individual lesion areas.

The red lines represent the edge of the lesion area in each image, as computed in MATLAB.

Chapter 4

Automated classification

Within the discipline of pattern recognition [124, pp. 291 – 292], automated classification is typically incorporated in computer-aided taxonomy systems or routine identification alongside image processing techniques [144]. Some of the techniques discussed in the previous chapter help segment digital images into individual regions and extract attributes from them, so that specific objects or patterns can be identified. Choosing a suitable method is therefore vital to any classification system.

This chapter provides an introduction to some of the common automated classification methods, which include the Bayes classifier, cluster analysis, neural networks, and evolutionary algorithms. A few of the pattern recognition algorithms discussed here will be further explored in chapter 6, to investigate the possibility of classifying leaf mines based on the morphological features extracted from images of leaves. It is worth mentioning some of the algorithms described in this chapter are not only efficient as the final classifier in routine identification, some are also used to sort pixels into various groups in the image processing stage.

4.1 An overview of classification

Despite being labelled as "automated", most computer-aided taxonomy systems are in fact semi-automated. Fully automated systems are largely idealistic – these

hypothetical systems would have the capability of identifying all existing taxonomic species, achieving 100% accuracy infinite times and requiring no user inputs. Such systems would benefit any research in biodiversity; yet they are mostly theoretical [145]. Semi-automated classifiers depend on a certain degree of intercommunication with users apart from original input, including making interim decisions and training the system. Inputs from users help guide the classifier, to make accurate judgements throughout the process.

Procedures in a typical classifier includes: class definition, which is problem specific and depends entirely on the project; data exploration, which determines the possibilities of using various features (both local and global) that could be used to distinguish between data samples; feature selection and extraction; building the classifier using a set of training data; testing the classifier, which utilises the remaining data as a testing set, with the results determining the reliability of the system [124, pp. 291 – 292]. Figure 4.1 shows an example of the processes involved in classifier design.

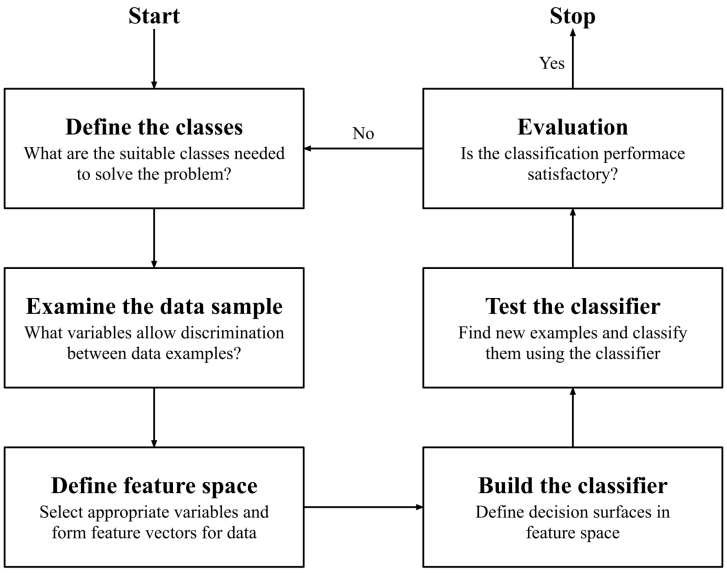


Figure 4.1: Flow diagram representing main steps in classifier design [124, pp. 294 – 296]]

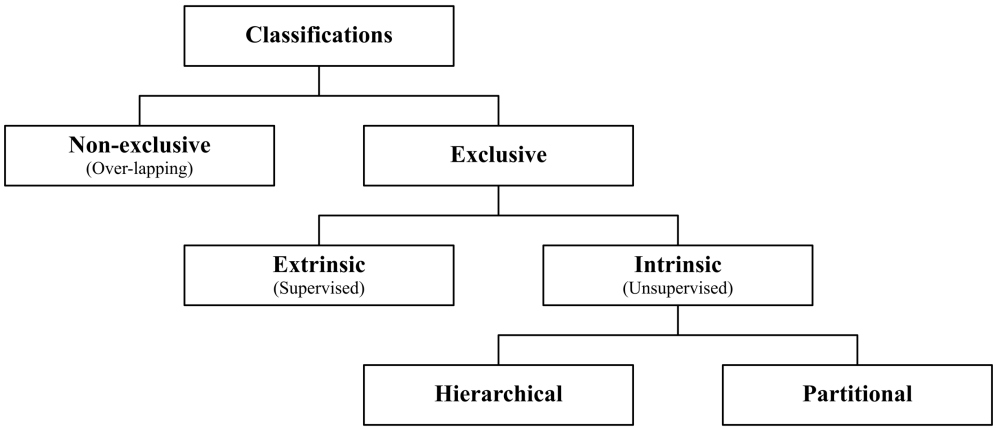


Figure 4.2: Tree of classification types [146, p. 56]

Classification can be either exclusive or nonexclusive – exclusive classification refers to the clear partition of a set of objects, so that objects of one subset are mutually exclusive to objects from another; while in nonexclusive classification, also known as overlapping classification, an object can simultaneously belong to multiple categories [146, pp. 55 – 58]. Figure 4.2 shows a decision tree that illustrates how classification can be divided into categories.

Exclusive classification can further be divided into intrinsic and extrinsic – intrinsic classification is unsupervised, and relies only on the proximity matrix, which is the measurement of distance or similarity between items; while extrinsic classification is supervised, and uses both the proximity matrix and category labels on the objects. The classifier required by this project is exclusive and extrinsic. Intrinsic classification is subdivided into hierarchical and partitional. Hierarchical classification groups objects according to a hierarchy, and partitional classification sorts objects into a predetermined number of groups according to their similarity. Hierarchical and partitional are two main techniques for clustering analysis, which will be further discussed in the following section.

4.2 Clustering analysis

Clustering analysis groups or organises objects (sometimes knowns as "individuals" or "subjects" etc.) into subsets, based on context or indices of

proximity between pairs of objects [146, pp. 1 – 6]. A cluster can be defined as a set of objects that are similar; whereas entities from a different cluster are not alike, such that the distance between any two objects within the same cluster should be significantly less than the distance between an object within the cluster and any object in a separate cluster [146, pp. 1 – 6] [147]. An index of proximity, or affinity, between pairs of patterns must exist for viable clustering methods.

Within the context of clustering, a set of objects can be divided into two standard formats – pattern or proximity matrices [146, pp. 8 – 10]. An $n \times d$ pattern matrix contains a set of n objects, each represented by a set of d measurements. Patterns and features are defined by row and columns respectively. A proximity matrix, on the other hand, is based on an index of proximity demonstrated by pairs of patterns [146, pp. 11 – 12]. It is the accumulation of proximity indices in which each row and column is a pattern, assuming all patterns have the same degree of proximity. As the proximity index is a measurement of association and interrelationship between patterns, it can either be a similarity or a dissimilarity. The proximity index between the i^{th} and k^{th} objects can be represented by $d(i,k)$ must comply with these three properties [146, p. 14]:

1.	(a)	$[d(i.i)] = 0$, for all i	for a dissimilarity	(4.1)
	(b)	$[d(i.i)] \geq max_k\, d(i,k)$, for all i	for a similarity	
2.		$[d(i.k)] = [d(k.i)]$, for all (i,k)		
3.		$[d(i.k)] \geq 0$, for all (i,k)		

4.2.1 Applications of clustering

Cluster analysis classify objects by exploring the structure of data, making it different from discriminant analysis, decision analysis or pattern recognition [146, pp. 1 – 6]. Decisions in pattern recognition are based on pattern class labels, whereas cluster analysis uses pattern class labels to verify classification results [146, pp. 241 – 245]. Clustering algorithms are considered more time-efficient, reliable and consistent compared to manual grouping processes, where individual analysts might identify and interpret data differently. However, clustering results can be misinterpreted because of certain factors, such as data type, normalisation, or scale.

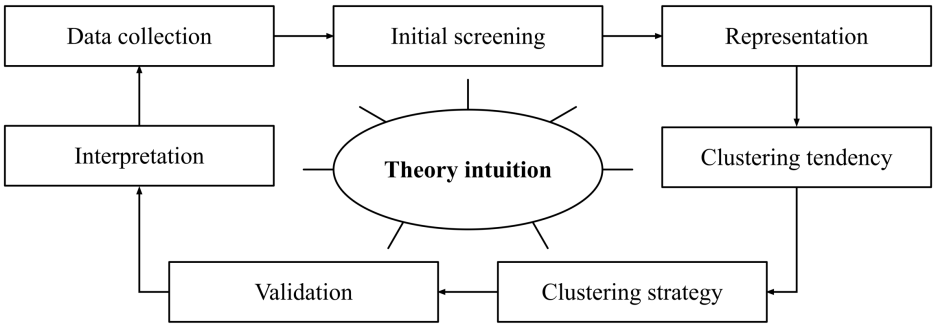


Figure 4.3: Clustering methodology [146, p. 144]

Clustering algorithms often include: data collection, which influences the methodology used to analyse the data; initial screening, such as normalisation, in order to prepare the data for further analysis; representation, whereupon the proximity index is chosen, and data is transformed and projected suitably; clustering tendency, an optional step which helps decide whether clustering would be suitable for the data; clustering strategy, choosing between hierarchical and partitional clustering ; validation, which evaluates clustering results; and interpretation, which draws conclusion about the data [146, pp. 135 – 137]. Figure 4.3 shows the relationships among the major steps in cluster analysis.

Cluster algorithms are paramount in solving problems in both image processing and pattern recognition. The applications of clustering often include feature selection [148], unsupervised learning [149], clustering of various kinds of data [150] [151], automatic indexing and classification [152] [153], speech recognition [154], image segmentation [155] and image registration [156]. The development of clustering methodology has been an interdisciplinary collaboration, including taxonomists and engineers. *K*-means clustering in particular, has been incorporated into CAT systems for sorting pixels into groups [35] [36].

4.2.2 Clustering algorithms

Many clustering algorithms are based on two popular techniques: hierarchical and partitional. Hierarchical clustering is exclusive and intrinsic; whereas

partitional clustering algorithms are categorised as exclusive, intrinsic and partitional. Sneath and Sokal created the acronym SAHN (Sequential, Agglomerative, Hierarchical, Nonoverlapping) to refer to exclusive, intrinsic, hierarchical, agglomerative algorithms [6]. Each clustering technique is best suited for different domains of applications – agglomerative hierarchical clustering focuses on the proximity index among the objects; while iterative square-error partitional clustering is nonhierarchical, and deals with data that takes the form of a pattern matrix.

A hierarchical clustering algorithm transforms a proximity matrix into a sequence of partitions, in which each partition is nested into the next [146, pp. 58 – 60]. This method is often depicted using a dendrogram, as seen in figure 4.4, which lists the clusterings in a systematic manner, so that every level defines a clustering and helps identify clusters.

Partitional clustering, on the other hand, organises n patterns within a d-dimensional space into K clusters, to the effect that patterns within a particular cluster are more similar to each other than patterns in another cluster. A clustering criterion, either local or global, is chosen, after which all possible partitions containing K clusters are evaluated, and the partition that optimises the criterion is selected. A global criterion identifies any cluster by a prototype, assigning patterns to cluster as per the most similar prototypes; whereas a local criterion utilises local structures to form clusters [146, pp. 89 – 92].

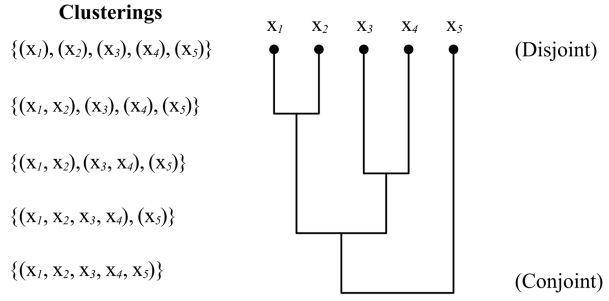


Figure 4.4: Example of a dendrogram [146, p. 59]

The square-error criterion is a popular strategy. Its objective is to minimise the square-error, or maximise the variation between clusters, for a particular partition with a fixed number of clusters [146, pp. 82 – 96]. Considering a set of n patterns, in d dimensions, that are partitioned into K clusters $\{C_1, C_2, \dots, C_K\}$, there are n_k patterns in cluster C_k:

$$\sum_{k=1}^{K} n_k = n \tag{4.2}$$

and the mean vector of cluster C_k can be given by:

$$\mathbf{m}^{(k)} = \frac{1}{n_k} \sum_{i=1}^{n_k} \mathbf{x}_i^{(k)} \tag{4.3}$$

where $\mathbf{x}_i^{(k)}$ is the i^{th} pattern in cluster C_k. The square-error, or within-cluster variation, for cluster C_k can be defined as the sum of the squared Euclidean distances between any two patterns in the same cluster, with $\mathbf{m}^{(k)}$ as the cluster centre.

$$e_k^2 = \sum_{i=1}^{n_k} (\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)})^T = (\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)}) \tag{4.4}$$

Partitional clustering plays an important role in biology, social and behaviour sciences. When constructing taxonomies, partitional clustering techniques can efficiently represent and compress a large amount of data. Dendrogram are unsuitable for representing partitional clustering because of the large dataset [146, pp. 89 – 92].

4.2.3 Image segmentation by clustering

Apart from the image segmentation approaches described in the previous chapter, clustering is also a technique that is used to partition an image into regions. Compared to image classification, image segmentation is sometimes considered more difficult, as the number of classes is typically unknown, unlike in classification problems. For basic image segmentation, in which "light" and

"dark" objects are represented by background and foreground pixels respectively, the threshold can be chosen using a grey-label histogram [146, pp. 225 – 227]. This approach is similar to one-dimensional partitional clustering, meaning each pixel corresponds to a pattern, and pixels are separated into groups according to only grey values. The same concept can be further adapted to multiple dimensions, given pixels can be represented by corresponding *d*-dimensional feature vectors, including grey values, gradients, or textures. Texture, in particular, is suitable for clustering.

In terms of machine vision and image processing, texture does not necessarily refer to the physical surface texture, but the spatial variation in pixel intensities in digital images [157]. The exact definition of texture is debatable [158], even though texture property is occasionally used as a guide to recognising image regions. It is a feature that plays an important role in image analysis, including several studies in automated inspection problems, which use textures to detect detects in images [159], textile [160] and lumber wood [161].

Common approaches of classification and segmentation of textures include: utilising grey-level histograms [162], in which channels respond to grey-level changes; and spatial filtering using partitional clustering, assessing a structure of n patterns in d-dimensional, where n is the number of pixels in an image, and d is the number of texture features [163] [164] [165] [166]. Another approach is the use of *K*-means clustering to minimise square-error. This method partitions the data according to a segmentation of the image. One of the statistics for assessing the validity of cluster k, S_k is defined as:

$$S_k = \frac{\min_{\substack{l \\ l \neq k}} \sum_{j=1}^{d} (m_j^{(k)} - m_j^{(l)})^2}{\left\{ \frac{1}{n_k} \sum_{i=1}^{n_k} \sum_{j=1}^{d} (x_{ij}^{(k)} - m_j^{(l)})^2 \right\}^{1/2}} \tag{4.5}$$

where n_k is the number of patterns in this cluster $m_j^{(k)}$, is the cluster centre, d denotes the number of features and $x_{ij}^{(k)}$ is the value of the j^{th} feature for the i^{th} pattern in cluster k [146, pp. 101 – 118, pp. 227 – 231]. A clustering is valid when the value of S_k is greater than a threshold for all clusters. Generally, a larger number in S_k indicates well-isolated clusters.

4.3 Neural networks

As mentioned in chapter 2, neural networks are popular choice of tool for training the classifiers within computer-aided taxonomy systems. Neural networks have been used in conjunction with digital image processing since the early 1990s due to their adaptability in solving a wide variety of problems [167]. Egmont-Petersen *et al*. suggested that neural networks can be trained to perform various image processing tasks, such as preprocessing, feature extraction, segmentation, object recognition, image understanding, and optimisation [168].

Although neural network algorithms do not accurately model biological connectionism, their development was inspired by networks of real neurons in the brain [169, pp. 2 – 6]. In a typical artificial neural network, a collection of neurons is connected by synapses, which allow signals to be transferred among neurons; although "neurons" and "synapses" in artificial neural networks are somewhat more straightforward than their biological equivalents [169, pp. 2 – 6].

McCulloch and Pitts proposed a simple model of neuron, presented as a binary threshold unit. Figure 4.5 shows a schematic diagram of the McCulloch-Pitts neuron. It generates an output of either 0 and 1 when a weighted sum of inputs is computed by a neuron, according to a certain threshold [170]:

$$n_i(t+1) = \theta\left(\sum_j w_{ij} n_j(t) - \mu_i\right) \tag{4.6}$$

The state of neuron i can be either *firing* and *not firing*; time t represents the time unit elapsing in each processing step; and $\theta(x)$ is the Heaviside function, also known as the unit step function:

$$\theta(x) = \begin{cases} 1 & if\ x \geq 0; \\ 0 & otherwise \end{cases} \tag{4.7}$$

The weight w_i refers to the strength of synapse which connects neurons j and i. The McCulloch and Pitt model indicates that a simple neuron, in principle, can be used in universal computation for any suitable weights w_{ij}, meaning the McCulloch-Pitts neuron is capable of performing computation similar to any ordinary digital computing system [169, pp. 2 – 6].

A simple single layer feed-forward network, as well as ones with multiple layers, can be referred to as perceptrons. Figures 4.6 (a) and (b) show examples of a simple single layer feed-forward network and a two-layer network respectively. Rosenblatt, who coined the term, further proved that perceptrons are capable of learning, which meant they are able to simulate specific functions [171]. While neural structures in the brain are far more complex than artificial ones, they are mostly feed-forward. Meanwhile networks with loops of connections, whether direct or indirect (i.e. networks that are not considered feed-forward) are generally known as recurrent networks [169, pp. 90 – 92].

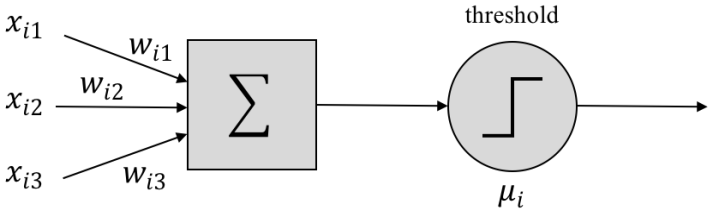
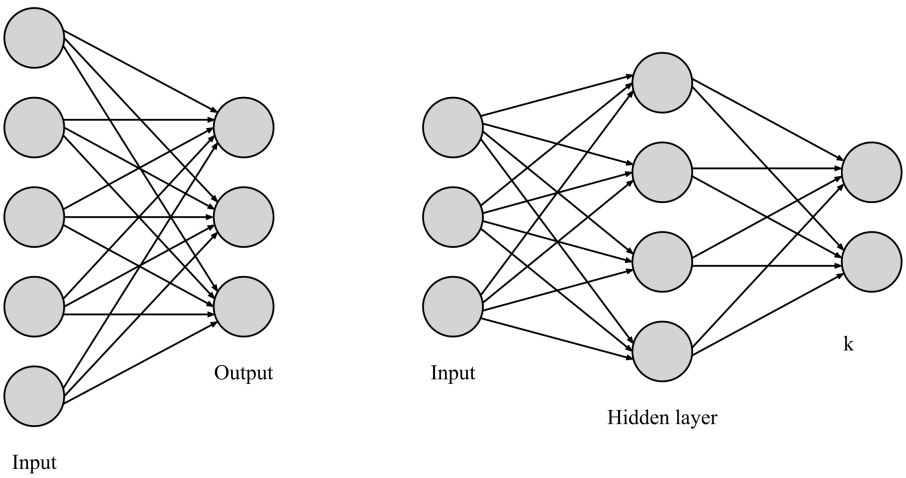


Figure 4.5: Schematic diagram of a McCulloch-Pitts neuron [169, p. 3]



(a) *(b)*

Figure 4.6: Feed-forward networks
(a) A simple perceptron, and (b) a two-layer perceptron

4.3.1 Feed-forward networks

In any feed-forward network, there is a set of input terminals, which feeds input patterns into the rest of the network; zero or more intermediate layer of hidden units; and a final output layer [169, pp. 90 – 92]. The hidden layer is known as such since it is concealed from the inputs and outputs. By definition connections in feed-forward networks are unidirectional, having asymmetric connection matrices w_{ij}. Simple perceptrons are feed-forward networks with a single layer; while an N-layer network has N layers of connections and $N - 1$ hidden layers, not counting the input terminals [169, pp. 90 – 92] [172, pp. 11 – 26]. For a simple perceptron, $g(h_i)$, the activation function can be given by [169, pp. 90 – 92]:

$$O_i = g(h_i) = g(\sum_k w_{ik}\,\xi_k) \tag{4.8}$$

where ξ_k is a set of N inputs and O_i is the output layer, which is an explicit function of the input. For a simple deterministic thresholding unit, $g(h) = sgn(h)$, with the assumption that the targets ζ_k^μ have values of ± 1 [169, pp. 92 – 100]. A desired outcome is where the actual output pattern O_i^μ equals target pattern ζ_k^μ for each i and μ. An example of a two-layer feed-forward network is shown in figure 4.7.

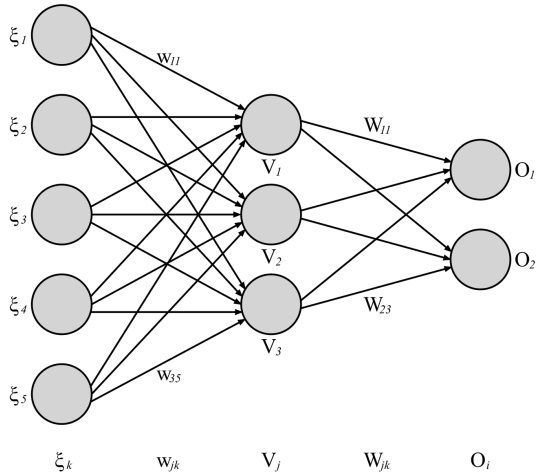


Figure 4.7: A two-layer feed-forward network; with the notation for units and weights [169, p. 116]

Each input pattern can also be represented by ξ_k^μ, where ξ^μ is a pattern vector in the same N-dimensional space; and the weights w_{ik} become a weight vector, where $w = (w_1, w_2, \dots, w_N)$. If a plane in the ξ space separates the patterns with $\zeta^\mu = +1$ and $\zeta^\mu = -1$, the problem is said to be linearly separable, and thus can be solved by a simple perceptron [169, pp. 92 – 100] [173]. As for multi-layer networks, they can be made solve a particular function by changing the weights w_{ik} [169, pp. 115 – 120].

4.3.2 Training algorithms

Through learning, whether supervised or otherwise, perceptrons can adjust themselves to produce desirable outcomes by sequentially modifying the network weight to produce such function through training, whether supervised or otherwise [172, pp. 11 – 26]. A network is typically trained over a number of training pairs, consisting of input and output vectors [172, pp. 43 – 59]. Initially weights are small random numbers, as the network could not learn if all the weights are of the same value and the desired outputs require unequal values.

Training can be divided into two groups: supervised and unsupervised. With supervised learning, learning is based on comparing the output of the network in question, with known correct answers; while in unsupervised learning, the only information lies in the correlations of the input data with the network aiming to create categories via learning; and to produce output signals according to the input category [169, pp. 8 – 10]. Unsupervised learning is more akin to biological neural structures compared to supervised learning, as brains do not train by comparing actual outputs with desired answers, before feeding the errors back through the network [172, pp. 11 – 26].

In unsupervised learning a training set consists of only input vectors, and with the absence of target vectors, outputs are not compared with any desirable outcomes. The objective of these training vectors is to adjust network weights so that consistent outputs can be produced by extracting statistical properties of the training set and grouping similar vectors into classes [174]. The Gaussian mixture models and Kohonen networks are examples of unsupervised learning algorithms, where measurements are not accompanied by class labels. Many

training algorithms are inspired by and evolved from a particular unsupervised learning model proposed by Hebb, in which the weight is increased if both the input and output neurons are activated [175]. This strengthens the frequently used paths in the network and facilitates learning.

In supervised training an input vector is coupled with a target vector to produce a training pair, with the latter representing the desired output [172, pp. 11 – 26]. When a network is trained with a number of training pairs, the difference between each actual output and their corresponding target output vectors are calculated, and fed back through the network to adjust the weight. This process, known as back propagation, will be further elaborated in section 4.3.3.

Eventually the weights are modified for each vector and the error of the training set is minimised to an acceptable level [172, pp. 11 – 26]. A supervised learning algorithm consists of an instance of data i, an attribute vector X_i, and target vector Y can be represented by:

$$E = \frac{1}{2}\sum_i (y_i - Y_i)^2 \tag{4.9}$$

where X_i denotes an attribute vector which is processed with a network: in order to produce an output y_i, which has the same form as target vector Y. Through the modification of the parameters of the network w (frequently by minimising the total square error).

4.3.3 Back propagation

Back propagation, sometimes known as error back propagation or back propagation of errors, was invented by Bryson and Ho [176], Werbos [177], Parker [178] and Rumelhart *et al.* [179] [180]. Back propagation is an algorithm that is based on a simple gradient descent which changes the weights w_{pq} in a feed-forward network, so that it can learn a training set of input-output pairs $\{\xi_k^\mu, \zeta_i^\mu\}$ [169, pp. 115 – 120]. It uses the forward coefficients W_{ij}, although errors δ are being propagated backwards instead of signalling forwards. Back propagation can also be bidirectional, as depicted in figure 4.8.

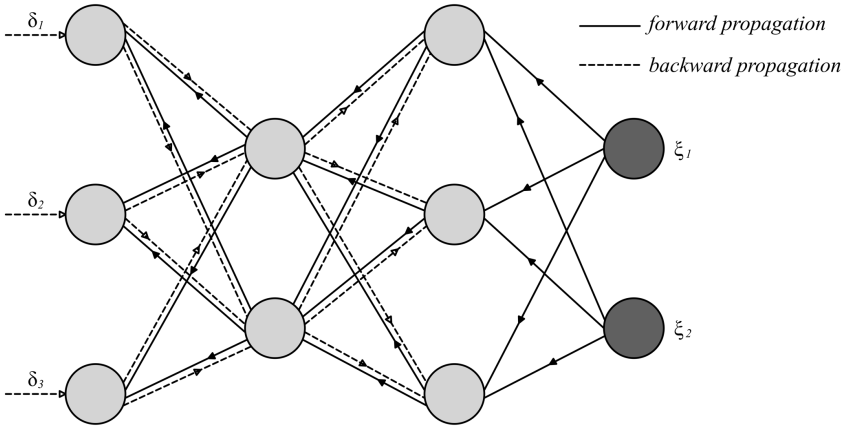


Figure 4.8: Back propagation in a three-layer network [169, p. 118]

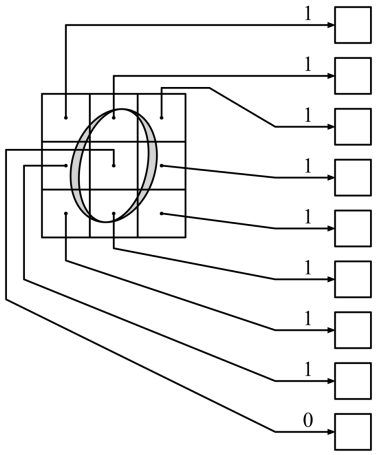


Figure 4.9: Image recognition [172, p. 48]

A pattern of '0's and '1's forming a binary image of the number '0' is the set of inputs of this training pair; while the output could be a number representing zero or a second set of '0's and '1's.

To train the network to recognise numbers 0 to 9,
10 training sets are required.

In general, back propagation takes the form [172, pp. 43 – 59]:

$$\Delta w_{pq} = \eta \sum_{pattern} \delta_{output} \times V_{input} \tag{4.10}$$

where p and q are the two ends of connection, referred to by *output* and *input* respectively; and V represents the suitable input-end activation from either a hidden unit or a real input. For the final layer of connections, δ can be given by:

$$\delta_j^{\mu} = g'(h_j^{\mu})\left[\zeta_i^{\mu} - O_i^{\mu}\right] \tag{4.11}$$

As for other layers, δ has the form:

$$\delta_j^{\mu} = g'(h_j^{\mu}) \sum_i W_{ij} \delta_i^{\mu} \tag{4.12}$$

For any network without backward or lateral connections, error propagation can be calculated after weights are initialised to small random values [169, pp. 115 – 120]. Each pattern ξ_k^{μ}, chosen and applied to the input layer ($m = 0$), such that

$$V_i^0 = \xi_k^{\mu} \qquad \text{for all } k \tag{4.13}$$

The signal propagated forwards through the network:

$$V_i^m = g(h_i^m) = g(\sum_j w_{ij}^m V_j^{m-1}) \tag{4.14}$$

for each i and m; and where V_i^m is the output of the i^{th} unit in the m^{th} layer, V_i^0 would be the same as the i^{th} input ξ_i and V_i^M would be the last output. The deltas for the final output layer can be calculated by comparing the actual outputs V_i^M and the desired outputs ζ_i^{μ} for pattern μ:

$$\delta_j^M = g'(h_j^{\mu})\left[\zeta_i^{\mu} - V_i^M\right] \tag{4.15}$$

By propagating the errors backwards, deltas for previous layers can be calculated:

$$\delta_j^{m-1} = g'\left(h_i^{m-1}\right) \sum_j w_{ij}^m \delta_j^m \qquad \text{for } m = M, M-1, M-2, \ldots, 2 \tag{4.16}$$

The connection can be updated using the gradient descent algorithm:

$$\Delta w_{ij}^m = \eta \delta_i^m V_j^{m-1} \qquad \text{and} \qquad w_{ij}^{new} = w_{ij}^{old} + \Delta w_{ij} \tag{4.17 – 18}$$

4.3.4 Image processing using neural networks

Convolutional neural network (CNN) is a class of deep neural networks which is commonly applied in visual image analysis. Unlike other neural networks discussed in the previous subsections, CNNs use a three-dimensional structure – width, length and depth [181]. This architecture allows the image to be scanned several pixels at a time, until the entire image is mapped. The resulted feature map contains features that belong to a required class and this first stage is known as convolution [181] [182] [183]. CNNs can perform numerous rounds of convolution before moving to the next layer, known as pooling. In the second layer, pooling produces a "summary" of the most important features on the image, by "downsampling" the dimensionality of each feature and keeping the most important information at the same time [181]. The fully-connected layer is similar to the one in regular neural networks, which determines the class of the image. A typical CNN structure is shown in figure 4.10.

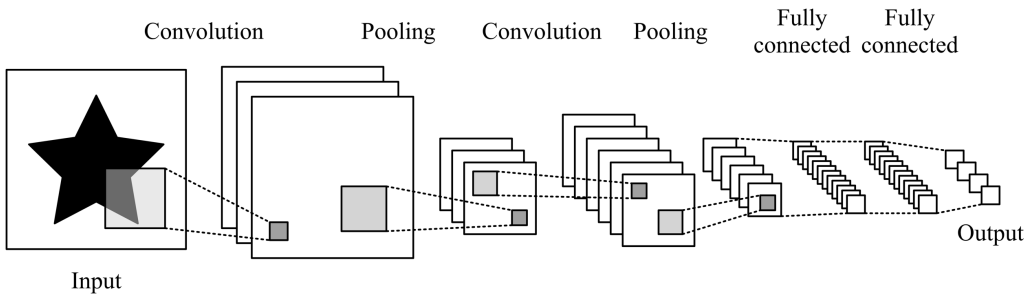


Figure 4.10: Typical CNN structure

CNNs use a three-dimensional matrix as an input feature map, with the size of the first two dimensions corresponding to the number length and width of the image in terms of pixels; and the third dimension has a size of 3 for full-colour RGB images, representing the three channels of red, green and blue [182]. Convolution then obtains tiles of the feature map, and applies filters accordingly to form new features, which then results in an output feature map, containing the convolved features. Size of the extracted tiles and the depth of the output maps are the parameters of the convolution process.

Common algorithms for the pooling stage include max pooling and average pooling [182]. The former uses the maximum value for each tile, while the latter takes an average value in order to determine the presence of a certain feature. The resulting fully connected layers of a CNN perform classification based on the feature extracted with convolutions. In fully connected layers, every node in the first layer is connected to every node in the second layer. The performance of CNNs is dependent on the parameters chosen for the purpose, such as number of layers and the number of feature maps in each layer. The accuracy for classification can be achieved through experimentation with sets of parameters.

As CNNs do not process entire images at the same time (unlike global thresholding), they are suitable for segmenting pixels into multiple classes. CNNs have been considered as an alternative to the image segmentation techniques described in chapter 3, in order to separate the pixels into three groups – leaf mines or tar spots; leaf; and the background (for example the background of a flatbed scanner). This will be further discussed in chapter 6.

4.4 Evolutionary algorithms

Evolutionary algorithms have been used for optimising image processing systems and neural networks [184] [185], or to perform clustering tasks [186]. They have also been implemented for segmentation purposes [187] and morphological classification of images [188]. Nickolay *et al.* proposed a method to optimise the parameters in image processing systems automatically using evolving algorithms, in order to increase the efficiency of the system and to simplify its usage [184].

Evolutionary algorithms (EAs) include a range of algorithms that are based on a major principle of evolution – "survival of the fittest". It is a phrase coined by Spencer in the 1860s, itself derived from genetic inheritance and Darwinian strife for survival [189]. By incorporating mechanisms of natural selection and population genetics, evolutionary algorithms can be used as an optimisation process. Dasgupta and Michalewicz attributed the development of EA-based tools to independent sampling from individuals, selection processes which preserve useful solutions, and the ease to modify partial solutions with "genetic" operators [190, pp. 3 – 4].

Generally speaking, an evolutionary algorithm maintains a population $P(t) = \{x_1^t, \dots, x_n^t\}$ for iteration t. Each individual represents a respective solution x_n^t, which is assigned a "fitness" according to user-specified criteria. By selecting the "fitter" individuals, a new population (iteration $t + 1$) is calculated, with individuals of the new population undergoing certain genetic transformations to form new solutions [190, pp. 4 – 6].

4.4.1 Genetic algorithms

One of the main paradigms of evolutionary algorithms is genetic algorithms (GAs), in which a population of potential solutions are maintained to facilitate a multi-directional search [190, pp. 6 – 8]. Like most computational systems that drew inspiration from nature, GAs can be adopted to solve technological problems, as well as used to understand and answer questions about nature [191, p. 35].

Genetic algorithms incorporate selection, crossover and mutation, where information formation is encouraged and exchanged in multiple directions. Typical elements of a genetic algorithm include: populations of chromosomes, selection according to fitness, crossover to produce new offspring, and random mutation of new offspring. It uses fitness functions, which assigns a score, to each chromosome in the present population to indicating the fitness [191, pp. 8 – 10].

For each iteration, a new population is created based on preceding generation according to these steps: (1) evaluation – individuals of existing population are

evaluated individually using a fitness function and assigned a value to denote their merit; (2) selection – individuals with better fitness are "selected" to create the next generation; and (3) mating – a new population is generated by applying genetic operators to selected individuals, including crossover and mutation. Generations are iterated until a satisfactory solution is reached.

This technique, dating back to 1950s but further developed in the 60s and 70s, often operated on binary strings [192] [193], with series of mutations involving crossover operators (where '0" becomes '1' and vice versa). Applications of genetic algorithms include: optimisation, such as numerical and combinatorial optimisation problems; automatic programming, in which GAs are used to evolve computer programming and design computational structures for specific tasks; machine learning, which includes classification and prediction tasks; evolution and learning, as a tool to study population genetics, and how species evolution affects each other; model and develop processes or systems in areas such as economics, ecology, sociology, immunology and robotics [191, pp. 15 – 16]. Pettey *et al.* proposed a new technique using genetic algorithms (GAs) as an alternative to *k*-means workload characterisation [194]. In this study, the GA-based technique outperformed *k*-means clustering models, which were discussed in section 4.2.2, by a factor of 10, successfully identifying both the number of workload classes and class centroids.

4.4.2 Evolving neural networks

Genetic algorithms have been incorporated into feed-forward neural networks to enhance network architectures, their learning and how the weights are calculated. Montana and Davis first used GA to evolve the weights in a fixed network, instead of back propagation [195]. Their results indicated that GA could outperform back propagation in certain problems, as well as the possibility to be used alongside back propagation to enhance its performance. Schaffer *et al.* suggested that GAs would be useful in finding weights in tasks, particularly for problems where back propagation is unsuitable [196].

Evolving network architectures can be primarily divided into two categories: direct encoding and grammatical encoding. The former, devised by Miller *et al.*,

uses simple fitness-proportionate selection and mutation in fast-forward neural networks [197]. Grammatical encoding, on the other hand, was first described by Kitano in the 1990s, as an alternative to direct-encoding [198].

Another application of genetic algorithms in neural networking is to evolve learning rules for neural networks. Chalmers [199] initially studied fully connected single-layer feed-forward networks. During training, network weights are modified according to the performance on the training data. To modify the weight for a single-layer feed-forward network, from input unit i to output unit j, can be given by:

a_i: activation of input unit i

o_j: activation of output unit j

t_j: training signal on output unit j

w_{ij}: current weight from i to j

Δw_{ij}: the change to make in weight w_{ij},
and $\Delta w_{ij} = f(a_i, o_j, t_j, w_{ij})$

Not only has genetic algorithms been used to optimise the performance of neural networks as classifiers, they have also been applied to image processing systems to optimise object recognition in a supervised learning context, which will be discussed in the next section.

4.4.3 Image segmentation using genetic algorithms

Genetic algorithms have been incorporated various computer vision systems to solve problems such as image segmentation, feature selection and object recognition. Unlike traditional image segmentation techniques, GAs can adjust to changes in image characteristics due to variable environmental conditions. GAs can provide adaptive capability within a computer vision by allowing the genetic system to modify a set of control parameters that affects the output of an existing system; by having the genetic component adjust the complex data structures within an algorithm or production rules for a computer vision application; or by allowing the genetic algorithm to make changes in the executable code of a program [200]. The performance of image segmentation can

be evaluated using global and local features of the image. An example of how image segmentation can be adapted using GAs is shown in figure 4.11.

Bhanu *et al.* proposed a method involving a closed-loop feedback control which allows the system to be adaptable, and a genetic learning system that optimises segmentation performance [200]. It utilises both global characteristics of the whole image and local features of separate object regions within the image. The result image of an adaptive image segmentation techniques can be used in region labelling and feature extraction, before being passed onto object recognition and classification. A flow diagram of the proposed system is depicted in figure 4.12.

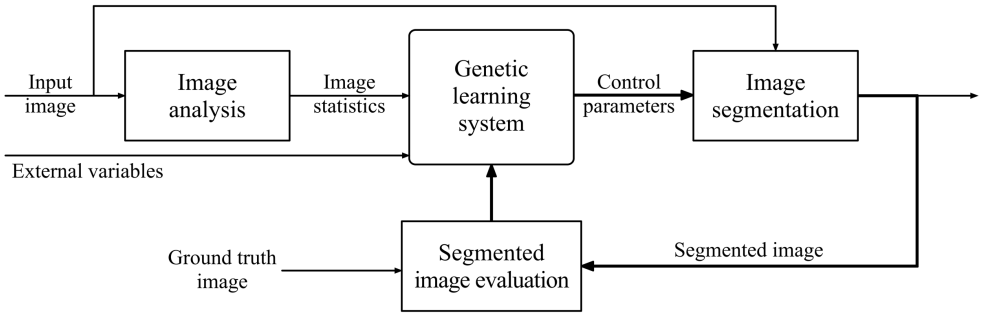


Figure 4.11: Block diagram of the adaptable image segmentation process [200]

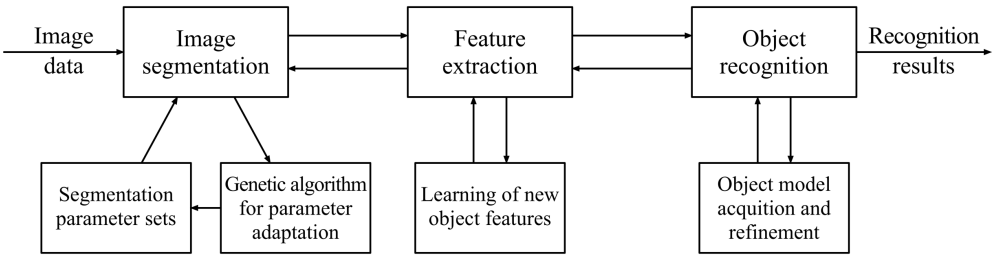


Figure 4.12: Conceptual design of the multi-level computer vision process proposed by Bhanu et al. [200]

4.5 Probability in machine learning

A selection of features is often used in image analysis and classification systems, as it provides a systematic way to compare data. Prominent features, such as the length or area of an object, are often employed as classification aids. The importance of each feature to the particular pattern class varies; and its effectiveness is dependent on its measurements, as well as the selection or number of features used.

4.5.1 Bayes' decision theory

By combining information obtained from features and *a priori* probabilities using Bayes' rule, the performance for object recognition can be enhanced [129, pp. 507 – 509]. The *a posteriori* probability of class C_i when feature x is observed, is given by:

$$P(C_i \mid x) = \frac{p(x \mid C_i)P(C_i)}{p(x)} \tag{4.19}$$

$P(C_i)$ is the *a priori* probability of class C_i, $p(x \mid C_i)$ is the class-conditional probability density for feature x in class C_i, and $p(x)$ is the probability density of feature x, which can be represented by:

$$p(x) = \sum_j p\big(x \mid C_j\big)P\big(C_j\big) \tag{4.20}$$

An increase in the number of features generally improves the performance of classification and lowers the classification errors. However, an error rate of 0 is not achieved by increasing the number of features indefinitely; when the total number of features exceeds an optimum, a percentage of features becomes statistically negligible and adds little valuable information. This introduces uncertainty to the system and inadvertently raises the error rates. Generally speaking, by increasing the number of training set patterns, a larger number of features can be used without compromising the classification performance [129, pp. 513 – 514] [201]. In data-dependent situations, certain features are more significant than others and have to be chosen via experimental tests [202].

4.5.2 Principal component analysis

A way to narrow down the number of features for classification is principal component analysis (PCA). PCA is one of the earliest techniques of multivariate analysis [203]; and an approach to represent data, which is closely related to cluster analysis. Pearson and Hotelling [204] [205] developed PCA with the objective to reduce the dimensionality of a large set of interrelated data, without giving up the variation present in the data set [206].

Correlated variables are condensed into single representation, known as the principal components (PCs), giving a satisfactory approximation of the original data. This essentially narrows down the number of components, as the first few PCs retain most of the variation present in all original variables [202], thus reducing the number of independent variables. As a multivariate method, PCA is suitable if the number of data is at least three or four times more than the number of components [207]; and Clemensat [208] suggested that PCA is more desirable if the number of data is more than ten times greater than the number of components. If correlation exists between any of the parameters, PCA can be a simpler and more effective way to classify the groups.

In order to narrow down the number of features, the first step is to normalise, or standardise, the data set by subtracting the mean and dividing by the standard deviation, if the units of all original variables differ from one another. The normalised data has zero mean, and the standard deviation of the data is 1.

The covariance matrix can be created by calculating the covariance of all original variable; and using the covariance matrix, eigenvalues and their corresponding eigenvectors can be obtained [203, pp. 8 – 22]. The eigenvalue of each eigenvector represents the quantity of variance it accounts for. A selection of eigenvalues is then chosen after arranging the eigenvector in a decreasing order of values, depending on cumulative variance or how accurate the approximation has to be. The more eigenvectors are used, the more variance are retained. As the first few PCs retain most of the variation present in all the original variables, there is negligible variance in the remaining PCs; taking all PCs into account, this allows the variance to be retained, even though the number of features are reduced [203].

4.6 Summary

The algorithms described in this chapter are often applied in conjunction with the digital image processing techniques introduced in the previous chapter. Clustering, neural networks, evolutionary and genetic algorithms, and probability-based classifiers are commonly used in computer-aided taxonomy systems and classifiers based on pictorial pattern recognition. Some of the pattern recognition algorithms, such as *K*-means clustering, convolutional neural networks, and genetic algorithms, can also enhance the image segmentation performance. The use of machine learning in segmentation is more complex than the thresholding and edge-based methods discussed in the previous chapter, and depending on the desired outcome, may produce superior results.

In particular, neural networks have been acknowledged as an effective classifier for the routine identification of plant diseases using digital images, as supported by the examples included in section 2.1. The use of a two-layer feed-forward neural network will be further discussed in section 6.4.

Chapter 5

Implementation

While implementing any digital image processing system, the principal requirement is to acquire samples in the form of digital images, before preparing them for further processing. The system's ability to analyse and interpret the images can be directly affected by their quality. For instance, if the images are out of focus, or if the lighting is insufficient, the system may not be able to accurately identify the objects in a particular image.

This chapter introduces various methods for collecting data and discusses their relative merits. Gathering of mines of *C. ohridella* and other leaf-mining insects is restricted to the latter half of the year due to phenology, environmental factors, growth rates, etc. Apart from collecting leaves with *C. ohridella* leaf mines for growth rate monitoring, leaf mines of other species were photographed using a digital camera to develop a database of high-quality reference and test images. The images then went through the preprocessing and processing steps, such as cropping, colour transformation, segmentation, edge detection and feature extraction.

5.1 Data collection

The prerequisite for any system that involves digital image processing is the acquisition of images, which can be broadly defined as "generating digital images using sensed data" [3, p.52]. This involves capturing objects or elements using sensors and sources of illumination. The energy from an illumination source is

reflected or absorbed by the objects or elements; and the sensor, or an array of sensors, produces an output based on the corresponding amount of light received by the sensor. An analogue-to-digital converter (ADC) then transforms the output into a digital image [3, pp. 46 – 52] [209].

For computer-aided systems using digital images, the most common image capturing methods are using suitable digital cameras and flatbed scanners. Photographing specimens or samples using digital cameras in a natural environment is one of the more popular methods to acquire images for digital image processing purposes [46] [210]. It had been used in a number of research approaches which investigate overall damage to agricultural plantations caused by leaf diseases [18] [21] [23] [25] [63]. Especially with the widespread availability of mobiles phones with camera functionality, digital photography seems to be a portable way to capture images in natural environment. However, live data collection is arguably more difficult and less likely to be controllable, hence limitations exist – the environmental, lighting and background conditions are difficult to control. Aside from digital cameras and smartphones, multispectral cameras or sensors placed in drones are occasionally used [211] [212]. Another data collection method is to take sample leaves from plants, place them on contrasting backgrounds and capture 2D images using digital cameras or flatbed scanners [22] [44] [45] [46] [47] [212]. In some cases, samples can be collected, sorted, preserved and retained for identification [213].

In this project, both digital cameras and flatbed scanners were used. Leaves were collected around York, mainly around the University of York campus throughout the summer months in 2014 to 2019. Some samples were then captured using digital cameras (iPhones 5, 6 and 7; Nikon 1 J5) in various settings. The iSight cameras used have 8- and 12-megapixels digital CCD (charger coupled device) sensors, and the Nikon 1J5 has a 20.8 megapixel BSI-CMOS (back-illuminated complementary metal-oxide semiconductor) sensor.

Leaves with existing *C. ohridella* leaf mines were also kept in airtight plastic containers indoor at room temperature, and scanned daily over a period of 7 to 10 consecutive days using flatbed scanners (Canon CanoScan LiDE 110 and HP Photosmart 5520 e-All-in-One Printer). The leaves were scanned daily in order to

observe the growth of area of leaf mine overnight. The scanned images or photographs were then transferred to computers (Apple Mac mini and MacBook Pro, macOS 10.9 Mavericks to 10.15 Catalina) via a USB 2.0 interface or via AirDrop (over Wi-Fi and Bluetooth). The resolution of the flatbed scanners was set to 300 and 600 dpi (dots per inch) as a suitable compromise of image quality and file size. Other resolutions were tested and considered but the result was inferior at lower dpi settings.

The first set of data was a set of 14 leaves of horse chestnut leaves collected in June 2014. Additional leaves were collected in the same method in throughout the summer months, and again in following summers. Several photographs of *Lyonetia clerkella*, *Rhytisma acerinum* and *Phyllonorycter* were also taken around the University of York, and were used for testing.

5.1.1 Image acquisition using digital cameras

Digital cameras mainly use either charger coupled device (CCD) or complementary metal-oxide semiconductor (CMOS) image sensors [125, p. 22] – 23] [214, pp. 41 – 42]. Light enters the camera through the lens and falls on the image sensor. The resolution determines the amount of detail the camera is able to capture – higher resolution equates to an increase number of pixels, which also means the camera is more capable of capturing finer details.

There are various ways for recording red, green and blue colours with a digital camera: (1) with three separate sensors, each having an individual filter; (2) by rotating a series of red, green and blue filters in front of the sole sensor, which records three images in rapid succession; (3) the sensor is separated into various red, blue and green pixels, and interpolation is used to estimate the true colour at that location [214, p. 45].

Factors such as lighting, background, orientation, and image quality can directly impact the accuracy of an automated identification system. These factors are more likely to be controllable under laboratory condition, with the aid of various tools including lighting equipment, plain backgrounds and overall set up. Some prefer to collect samples and store them in a controlled environment, as the

quality of images is more controllable in terms of lighting condition, position of the leaves in images, as well as the orientation. Controlling these factors improves the consistency of images and simplifies the image analysing process.

When photographing leaf samples in an indoor setting, samples are conventionally placed upon a white background, and a camera is set up at a fixed distance, perpendicular to the object in question. A single light source is often used to provide light over the object. Such process relies on uniformity of lighting in a controlled environment. Assuming the light source, camera and objects are placed apart, the following issues may affect the result:

- the variations in the angle of incidence mean parts of the leaf can be illuminated differently
- the reflectivity of various parts of the leaf
- parts of the objects and background are in the shadow
- the distance among the background, camera and object, as well as the directions relative to them

When photographing leaf samples in an outdoor setting using digital cameras, lighting becomes an even more complex issue. Sunlight is not a controllable light source, and its direction or intensity is not consistent. Artificial light sources, such as flashlight or light boxes, are more easily controlled. Additionally, in order to determine the precise distance between the camera and its object, both have to remain stationery, which is not particularly practical for handheld camera due to its weight or external factors such as wind. Despite these challenges, digital photography remains a highly favoured image acquisition method for many computer-aided taxonomy systems. This is due to the popularity of mobile phones with built-in digital cameras.

5.1.2 Scanning mechanism

Flatbed scanners, also referred to as reflective scanners, work by shining a white light source onto flat, opaque materials; and reading the intensity and colour of the light reflected from it [215]. Typically an object is placed on a glass window, face-down beneath the cover, which provides a controllable background; and as most leaves are primarily flat, scanning is an ideal way to digitalise such objects.

Flatbed scanners use the CCD array, with a fluorescent or xenon lamp to illuminate the document, and the image is reflected by mirrors. Since CCD elements are brightness sensitive, the pixels store only luminance information of the original image. As for the colour and chrominance information, three CCD elements for each pixel, and the red, green and blue components are combined to form an RGB scanned image [125, pp.22 – 23].

The colour depth, or bit depth, depends on the scanning array; high colour depth means that more colours are used and the quality of the scan will be better. The scanners used in this project, a Canon CanoScan LiDE 110 and HP Photosmart 5520, can digitalise objects with a depth of 48-bit and 24-bit respectively; and both have an 8-bit output for greyscale images [216] [217]. The maximum size of the object is limited by the size of the glass window – for both scanners used in this project, the maximum scan size is 216 x 297 mm (8.5 x 11 in).

Dots per inch (dpi) is the standard measure of spatial dot density of an image scanner or printer, which is the number of individual dots that can fit into a linear inch [3, pp. 59 – 65]. Dpi is different from pixels per inch (ppi) – an image of 100 × 100 pixels, when printed in a 1 inch square, would have a resolution of 100 ppi. For printing purposes, a higher dpi translates to better quality, as indicated by a larger number of dots. For scanned images, however, the dpi setting of the scanner correlates to the final size of the images in pixels. As the resolution of a digital image is measured in pixels, a higher dpi setting means more information is collected during the scanning process, hence a superior image quality. A setting of 300 dpi to 600 dpi would provide a quality scan with a reasonable file size. The maximum resolution for the Canon CanoScan and the HP Photosmart 5520 are 2400 × 4800 dpi and 1200 × 2400 dpi respectively [216] [217].

Figure 5.1 (a) shows a cropped image (560 × 420 pixels) of a decimal one penny (1p) coin, scanned at 300 dpi. A standard 1p coin has a diameter of 20.3 mm (0.7992 inches), each side of a 1p coin has an area of 323.6547 mm^2 (0.5017 square inches). Using the same approach described in this chapter, the estimated area of a 1p coin is 45564 pixels. This is indicated by the object area within the red outline in figure 5.1 (a). The green circle in figure 5.1 (b) has the same area and centre as the object in figure 5.1 (a), which has a radius of 120.4303 pixels.

(a) *(b)*

Figure 5.1: A decimal one penny (1p) coin scanned at 300 dpi.

The object outlined in red in (a) has the same area as the green circle in (b). At 300 dpi, the area of each side of a 1p coin is approximately 45564 pixels, with a radius of 120.4303 pixels.

Regarding the issues addressed in subsection 5.1.1, both the background and lighting elements are consistent in flatbed scanners; hence the result images are unlikely to be subjected to distortion, reflection or shadows, where compared to acquiring images with digital photography.

5.1.3 Image compression and file formats

Digital images contain large amount of data, leading to the necessity of image compression – the reduction of information while retaining an acceptable image quality [125, pp. 444 – 445]. A block diagram of general image compression is shown in figure 5.2. The higher quality, the larger amount of storage space and transmission bandwidth are required. Scanned images have a set numbers of dpi (dots per inch), which can be converted into ppi (pixels per inch), as demonstrated in section 5.1.2. Photos taken with smartphone cameras are typically around 8 to 12 megapixels, while single-lens reflex cameras create images with 20, or sometimes up to 60 megapixels. Due to their large file sizes, images with more than 12 pixels are not likely to be practical for the proposed approach – the software and hardware used for developing the system may not be able to process them efficiently, and the processing time would be considerably longer.

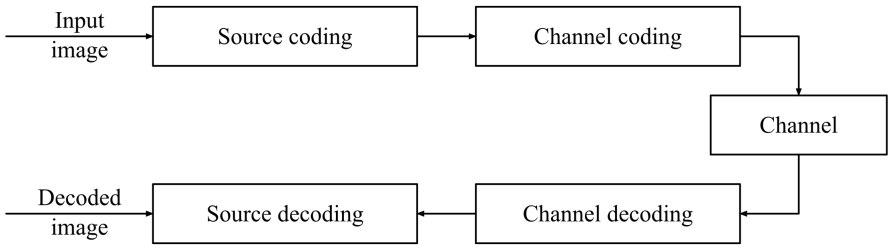


Figure 5.2: Image compression [125, p. 447]

Image compression can be divided into two groups – "lossy" and "lossless" [125, p. 447]. The former is known as irreversible compression, since parts of original image information is discarded during the process and could not be restored. In lossy compression, the reconstruction is an approximation of the original data, which is a trade-off for a higher compression ratio. Lossless compression is fully reversible, as all the information is preserved during the compression and decompression processes; none of the image data has been compromised, so that the reconstruction image is exactly identical to the original image.

One of the most popular image file formats that utilises lossy compression is JPEG (Joint Photographic Experts Group, .jpg or .jpeg), or its successor JPEG2000 [125, pp. 639 – 640]. The compression process mainly includes the steps of: preprocessing – decomposing image components into rectangular tiles, dc-level shifting and component transformation; core processing – wavelet transformation, quantising coefficients and entropy coding; bit-stream formation – precincts and code blocking. It is the default image output format of both scanners used. It is a joint international standard for the image compression on both colour and grayscale still images [125, pp. 488 – 494], which employs block-based discrete cosine transform (DCT), which allows the image to be cropped, rotated, transformed or converted into a greyscale image, as well as significantly reducing the file size. Block diagrams of JPEG encoder and decoder are shown in figures 5.3 and 5.4.

PNG (Portable Network Graphics) is a bitmapped image format which uses lossless compression [3, p. 541]. PNG supports 16 million colours, and was

designed specifically for transferring images via the Internet. It uses a 2-stage compression process – pre-compression, involving filtering; and a non-patented lossless data compression algorithm known as Deflate. PNG is considered an ideal format for picture editing, as images can be fully reconstructed from the compressed data. The processing time of PNG images is considerably higher compared to JPEG images, due to the reduced compression ratio.

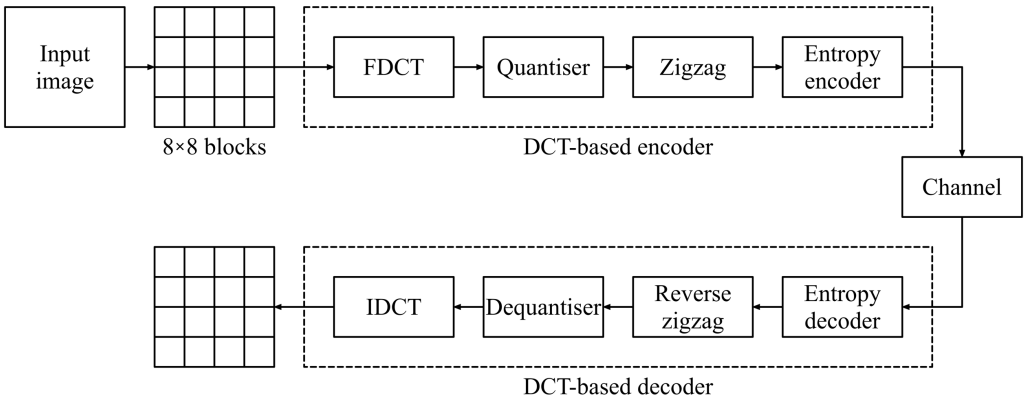


Figure 5.3: Baseline JPEG encoder and decoder [125, p. 489]

FDCT and IDCT refers to Forward and Inverse DCT respetively; quantised DCT coefficients are ordered according to a zigzag scan, which is based upon the observation that most of the high frequency coefficients are zero after quantisation.

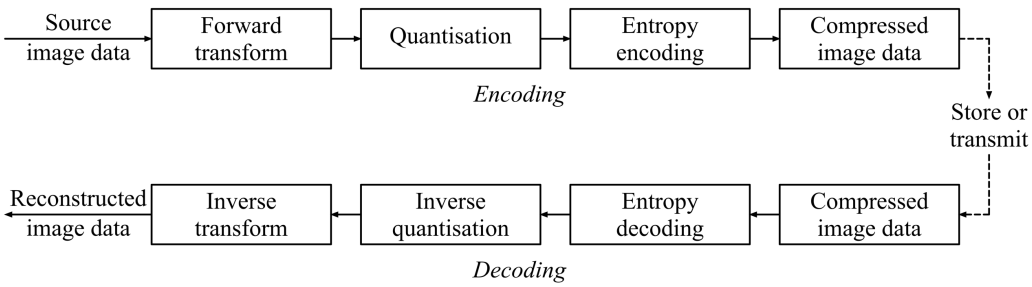


Figure 5.4: A general block diagram of the JPEG 2000 encoder and decoder [218]

TIFF (Tagged Image File Format) supports all resolutions, sizes, and colour depths [3, p. 541]. The quality is preserved in TIFF images but its larger file size limits its use. Raw image files are data from image sensors of digital camera or scanners before they are processed. RAW files retain the information required to produce a viewable image but they cannot be printed or edited with a bitmap graphics editor. They support all colour spaces with more shades of colours compared to JPEG files (raw images supports 12 or 14 bits of intensity information per channel, while JPEG supports 8 bits or 256 shades).

RAW image files contain data from the image sensor of a digital camera or an image scanner, with minimal compression. RAW images bypassed the processing steps of sharpening and noise reduction, which are typically automatically performed on JPEG images during the compression process. Unlike other image file formats, settings for white balance, colour saturation, contrast, and sharpness are not predetermined, either by the photographer or automatically, but instead are saved for further processing. Most RAW formats use lossless compression to reduce file size without compromising quality, but some use lossy compression for filtering and quantisation. Depending on the compression ratio, RAW files are approximately 2 – 6 times larger than their JPEG equivalent; for instance, using the Nikon 1 J5 and the Nikkor 10.0 – 30.0 mm VR lens at ISO 320, the file sizes for the RAW and JPEG images are 21.4 MB and 11.2 MB respectively at 5568 × 3712 pixels (20.7 megapixels). RAW images are converted into TIFF or JPEG file formats for storage and processing using a RAW converter [219] [220].

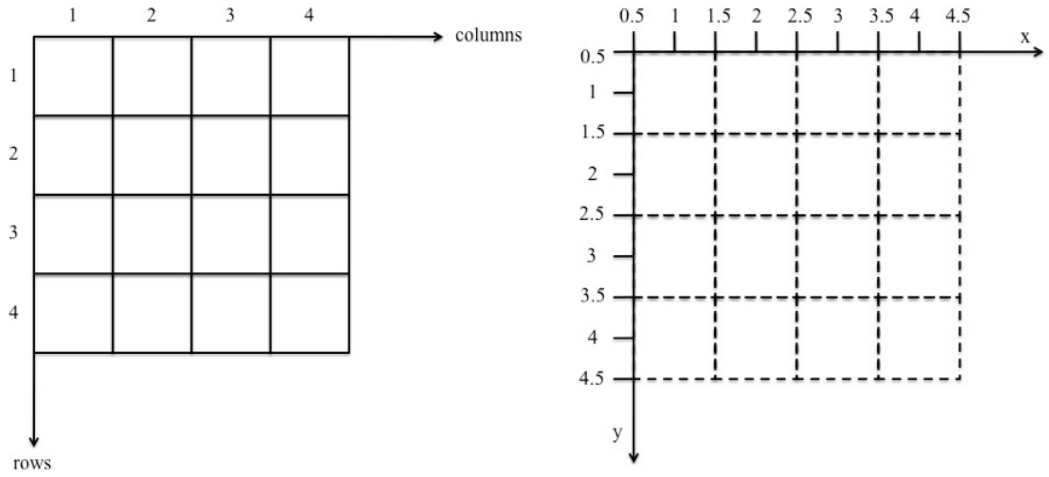
With modern photo editing software, the processing time for RAW images have been significantly reduced. However, it is an additional step that would not be required otherwise when using ready-to-use image formats, such as JPEG. Moreover, raw image files are not standardised and numerous raw formats are currently in use, unlike the JPEG format. For the purpose of this project, the use of RAW image is impractical without specialist hardware and software.

5.2 Image analysis

The proposed detection tool for identifying *C. ohridella* leaf mines was developed in MATLAB (MATrix LABoratory) versions R2014a through R2018b, with the

addition of various Toolboxes, including the Image Processing Toolbox, the Deep Learning Toolbox (known as the Neural Network Toolbox in previous versions), and the Statistics and Machine Learning Toolbox (the successor of the Statistics Toolbox) [221] [222] [223]. Developed by Math Work Inc. for digital signal processing, MATLAB is a programming language that allows matrix and array calculation, graph plotting, algorithm implementation, etc., and facilitates a quick and easy visualisation of results [224]. The MATLAB image processing toolbox supports a wide range of image formats, and provides a diverse range of algorithms and functions aimed at image analysis. Subsequently, two more species of leaf mines were introduced: *L. clerkella* and *P. ilicis*; as well as the plant pathogen *R. acerinum*.

In this chapter, pixels labelled "object area" indicate the pixels which can be regarded as leaf mines or tar spots, and the "leaf area" refers to the pixels which represents the leaf itself. In binary images, the pixels with a value of "1" (i.e. foreground) mainly includes the "object area" pixels, while the background pixels, with the value of "0", includes the "leaf area", and occasionally the area that is neither of the leaf or the "objects" (for instance, the background beneath the leaf).



(a) *(b)*

Figure 5.5: 2-D Cartesian coordinates in MATLAB [225]

Image data are represented in three numeric classes in MATLAB – double-precision floating-point (`double`), 16-bit unsigned integer (`uint16`) or 8-bit unsigned integer (`uint8`) [226]; and any image can be interpreted as a two-dimensional (2-D) function $f(x, y)$, with x and y as spatial coordinates [3, pp. 1 – 3], as shown in figure 5.5. For example, the intensity at coordinates (x, y) can be given by $f(x, y)$ for a binary image with m rows and n columns. With double-precision arrays, each colour component of an RGB image has the value between 0 and 1; for instance, (0, 0, 0) corresponds to a pixel in black, and (1, 1, 1) corresponds to one in white [227].

5.2.1 Image preprocessing

Image preprocessing involves the range of techniques to manipulate or improve image data, so that the images become better suited for further processing and specific applications. The goal for preprocessing is to enhance features vital to further processing, and suppress irrelevant or undesired information. Preprocessing methods include geometric transformations, such as rotation and scaling; image enhancement and restoration, both of which improve the appearance of an image visually [3, p. 25 – 28] [214] [228] [229].

One of the first preprocessing steps is to perform some basic image editing and manipulations, such as resizing, cropping and rotating, on the scanned images and photographs. For instance, it is essential to crop images into smaller portions, which puts emphasis on the subject (leaf mines and tar spots) and improves the framing of the images.

Cropping allows a smaller section of the original image to be focused on, while simultaneously reducing storage size and lowering processing time. It discards the parts of an image that are deemed unnecessary or undesirable for purpose, which could include background information that is irrelevant to the leaves, leaf mines and tar spots, as well as parts of the image that are blurry or rendered not suitable for further processing because of the image acquisition method. Leaf mines of *C. ohridella* are clearly visible on the upper surface of affected leaves, thus can easily be identified on upper surface of *A. hippocastanum* leaves. A similar approach can be used on leaves with tar spots – the dark spots are in

contrast to the green leaves, making them suitable for intensity thresholding and edge extraction.

For scanned images of *C. ohridella*, there are usually multiple leaf mines on a single leaf, but some are not easily identified because of blurriness; or their location on the leaves (too close to the edges or leaf veins), or other issues encountered during the data collection or image acquisition stage.

This process was manually performed on Preview and Photos (previously iPhotos), which are both image software for MacOS. The cropping was based on the *a priori* knowledge that the leaf mine(s) or tar spot(s) should be located in the centre of the image. Additional *a priori* knowledge is given by distinct differences in colour between the "leaf" and "object" areas in the image. All the species of leaf mines used in this project, as well as the tar spots, can easily be distinguished from the background by their colour.

Unlike resizing, cropping does not change the resolution of the image, which is based on the digital camera used or the dpi setting of the scanner. Without altering the resolution, the size of each "object" area is not affected by the cropping, as it is only dependent of the number of pixels within that area. As with many image enhancing processes, cropping was performed based on visual interpretation, to ensure the "object" is in the centre of the cropped image.

5.2.2 Image segmentation based on colour spaces

After the scanned images or photographs have been prepared, they are stored in the same folder as other files that are run on MATLAB for ease of use. All the images acquired with the methods described in section 5.1 uses the RGB colour system by default, and are processed using the techniques discussed in chapter 3 – from transforming into HSI colour space, to segmentation and edge detection.

RGB images consist of three component images, one for each primary colour [3, p.402 – 406]. The three component images are monochrome intensity images (representing red, green, and blue) of the same dimension of their full-colour counterpart.Though normally combined on screens to form a composite colour image when presented on an RGB monitor, as shown in figure 5.6, it is possible

to process the red, green, and blue images on their own. Likewise, HSI images are composed of three stand-alone component images, for hue, saturation, and intensity. An image in RGB or HSI colour models is stored and represented as an *m*-by-*n*-by-3 data array, which defines red, green and blue (for an RGB image), or hue, saturation and intensity (for an HSI image).

When segmenting images based on colours, the HSI colour model has the convenience of decoupling colour and intensity information – colours in an HSI image are represented in the hue component image, and saturation is often used as a masking image to separate regions of interest in the hue image. The intensity component contains no colour information, and is rarely used in segmentation. When smoothing the intensity component of HSI images, the hue and saturation of each pixel are not affected [3, pp. 439 – 445].

Figure 5.7 (a) shows an image of an RGB image of a *C. ohridella* leaf. Figures 5.7 (b) through (d) show the red, green, and blue components of this image. Figures 5.7 (e) is the HSI equivalent of the image in figure 5.7 (a), using the processes described in chapter 3. Figures 5.7 (f) through (h) show the hue, saturation, and intensity components of the RGB image.

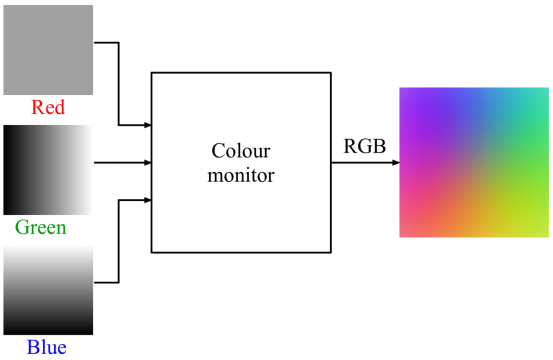


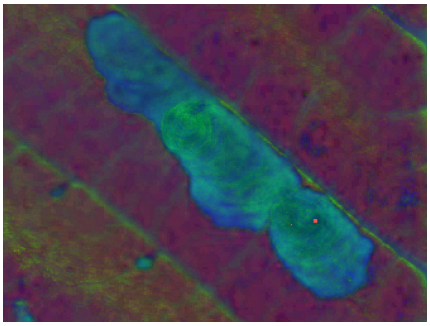
Figure 5.6: RGB colour rendition of the original colour scene using the three RGB component images [3, pp. 403 – 404]

(a) Original RGB image

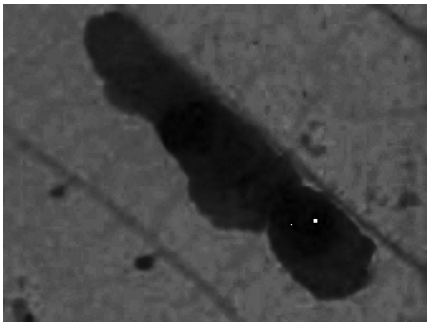
(b) Red component image

(c) Green component image

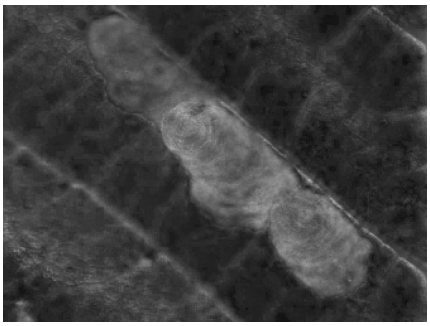
(d) Blue component image



(e) HSI image



(f) Hue component image



(g) Saturation component image

(h) Intensity component image

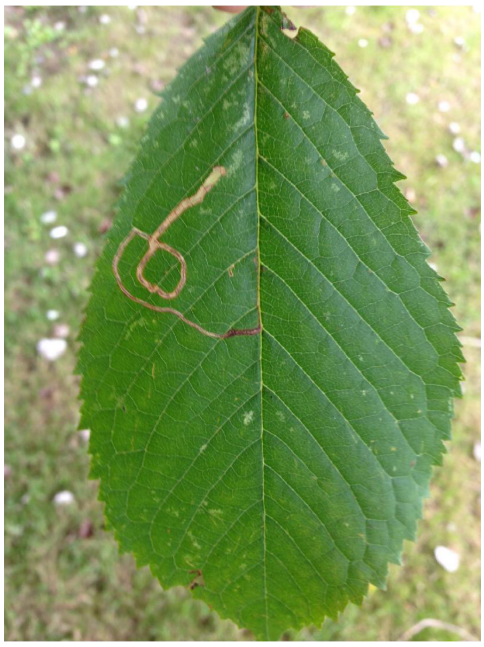
Figure 5.7: Original RGB and HSI images, with its red, green, blue, hue, saturation and intensity compoment images

To investigate the benefits of masking different channels in RGB and HSI images, a simple system was devised and tested using images of leaves from various species affected by a range of pests and disease. As green is the dominant colour of a healthy leaf, green coloured pixels in the images mostly correspond to the "healthy" areas on the leaf. These areas do not add value to the leaf mines in question and removing the green coloured pixels can reduce processing time and improve efficiency.

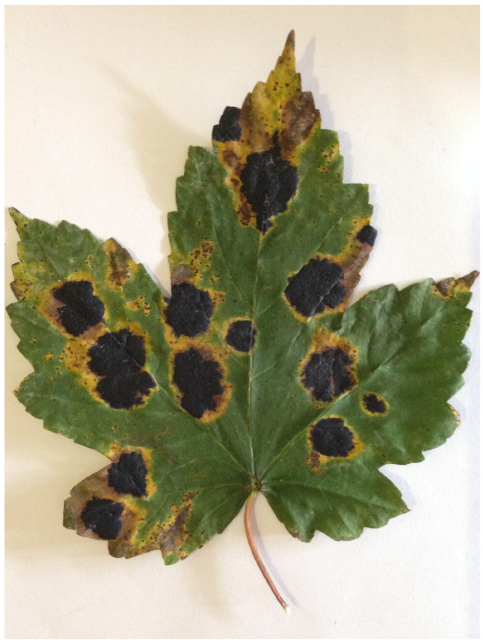
Since the hue component carries information of the pure colour of a pixel, by setting maximum and minimum values of hue, the colours within the range can be "masked" and removed from the picture. Iandge *et al*. adopted the concept of comparing the green component of "mostly green pixels" and the rest of the image [23] – if the green component of a certain pixel is smaller than that of the average of the "mostly green pixels", the RGB component of the pixel would be set to zero (i.e. removed) and the rest of the image could be further processed – conversion to binary image and edge extraction for example.

A full-colour photograph of an apple leaf is shown in figure 5.8 (a), and an example of a maple leaf is shown in figure 5.8 (b). These RGB images are used as the original inputs for the component images, binary masks, and histograms in figures 5.9 and 5.10. For the apple leaf image in figure 5.9 (a), its equivalent in the HSI colour space, and the three HSI component images are shown in figures 5.9 (b), (d), (g), and (j). The combined histograms of the hue, saturation, and intensity components are shown in figure 5.9 (c), with each of them shown individually in figures 5.9 (f), (i), and (l). The data in all the histograms are normalised to the range of [0, 1]. Figures 5.9 (e), (h) and (k) shows the binary hue, saturation and intensity masks, which are generated with the hue, saturation, and intensity component images.

Similarly, the RGB image of the maple leaf in figure 5.10 (a) is transformed into an HSI image, shown in figure 5.10 (b). The hue, saturation, and intensity components are shown in figures 5.10 (d), (g), and (j). The histogram in figure 5.10 (c) is the amalgamation of the histograms in figures 5.10 (f), (i), and (l). The binary masks in figures 5.10 (e), (h), and (k) are produced with the same method as the ones in figures 5.9 (e), (h), and (k).

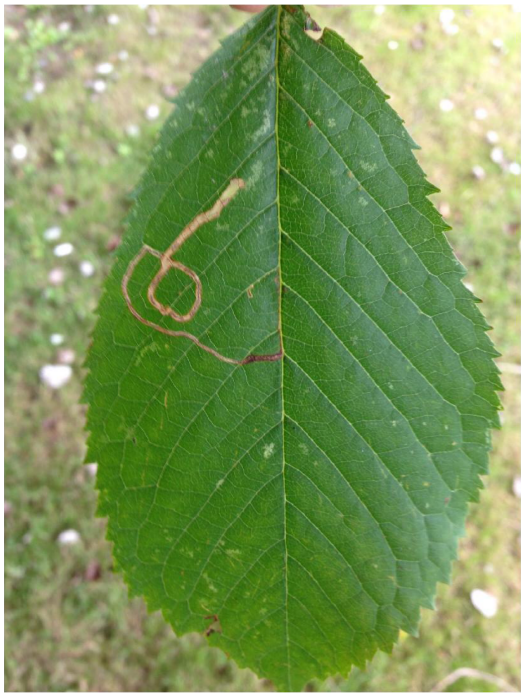


(a) Original image used in figure 5.9 –
a photograph of a Prunus padus leaf with a Lyonetia clerkella leaf mine



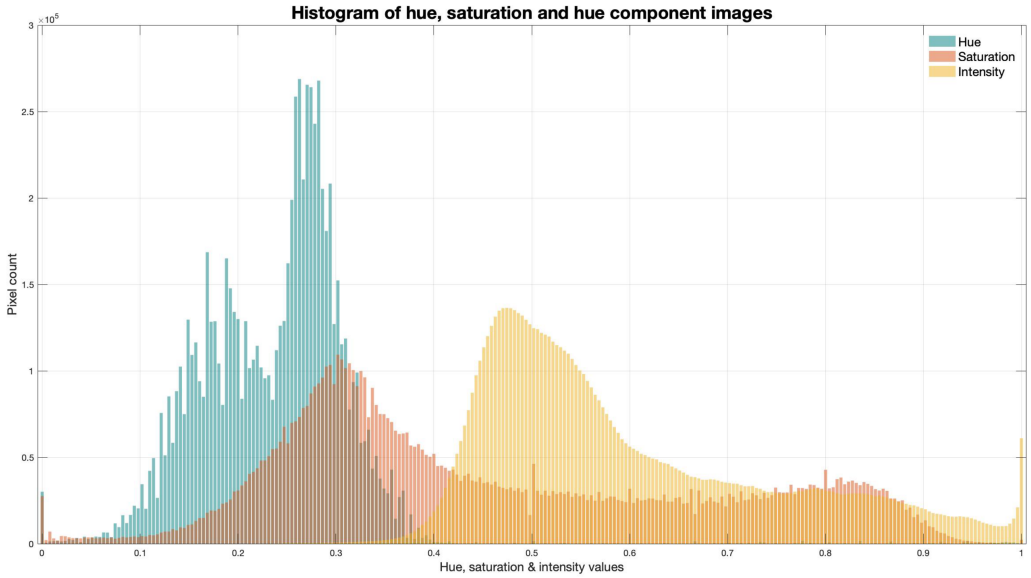
(b) Original image used in figure 5.10 –
a photograph of an A. pseudoplatanus leaf with tar spots

Figure 5.8: Original RGB images used in the follow figures



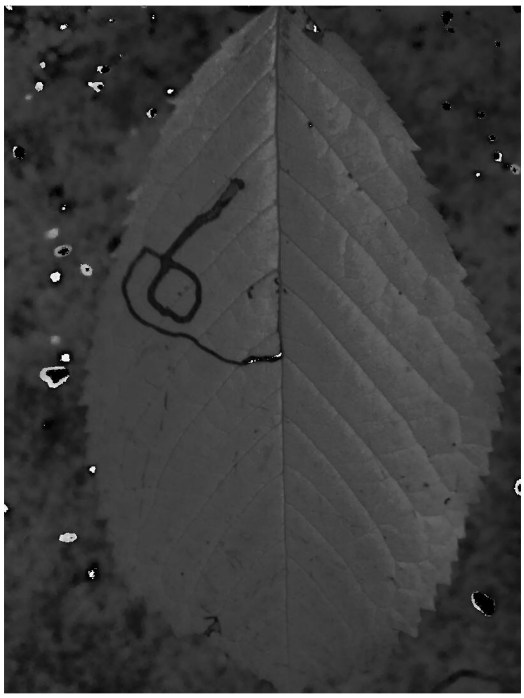
(a) Original RGB image

(b) Same image in HSI colour space

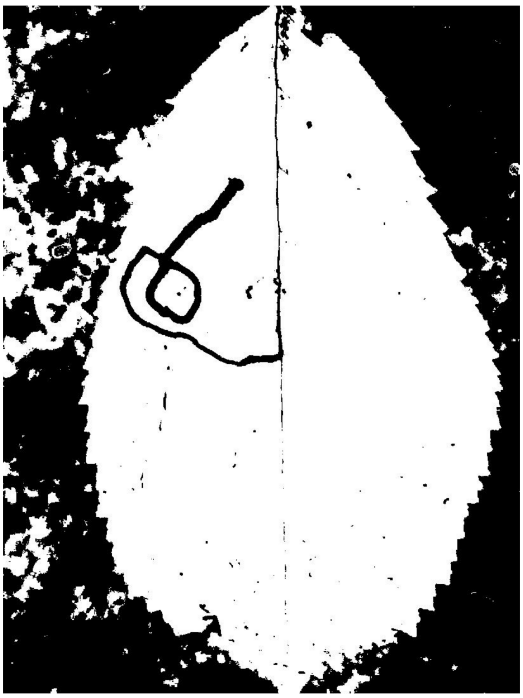


(c) Histogram of all 3 component images of HSI colour space

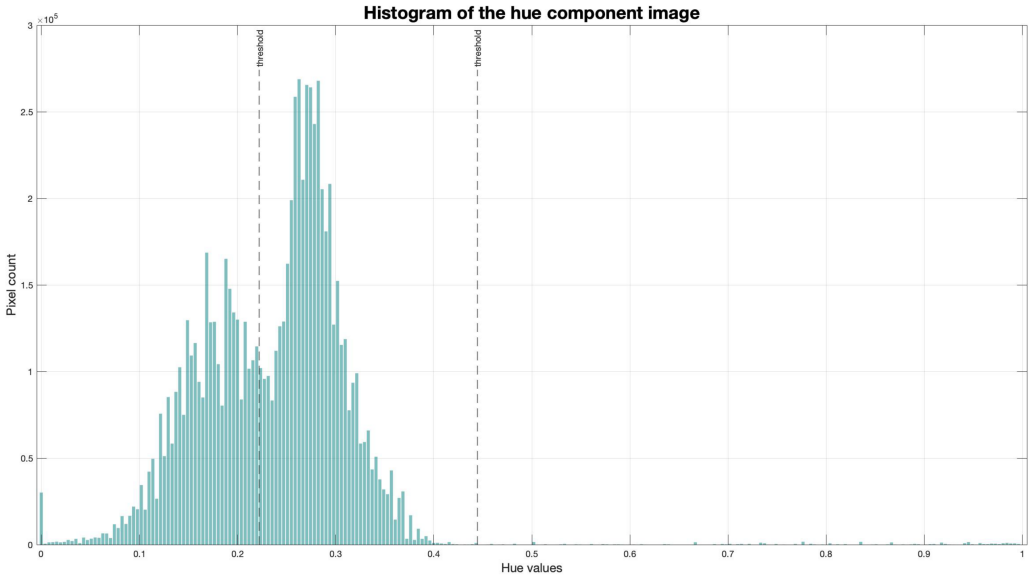
Figure 5.9 (a) – (c): L. clerkella leaf mine – segmentation and histograms using HSI colour space (part 1)



(d) Hue component image



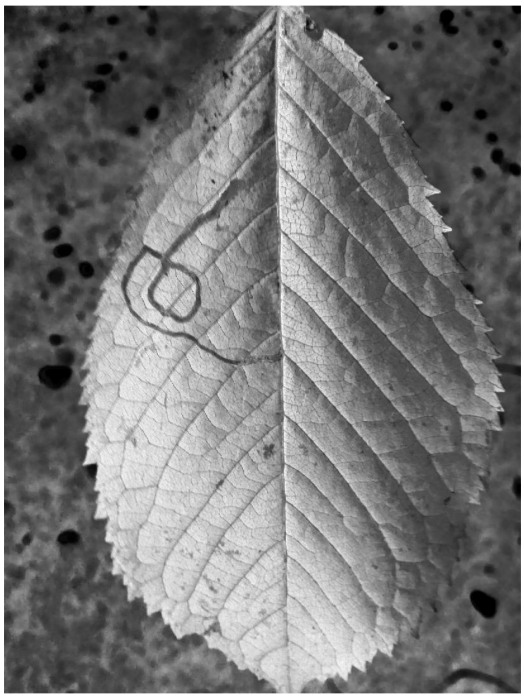
(e) Hue mask



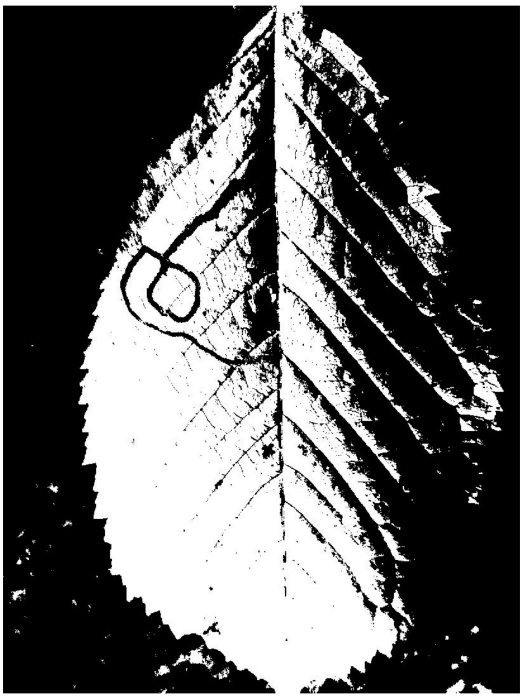
(f) Histogram of the hue component image

Threshold values for creating the hue mask: 80° and 160°

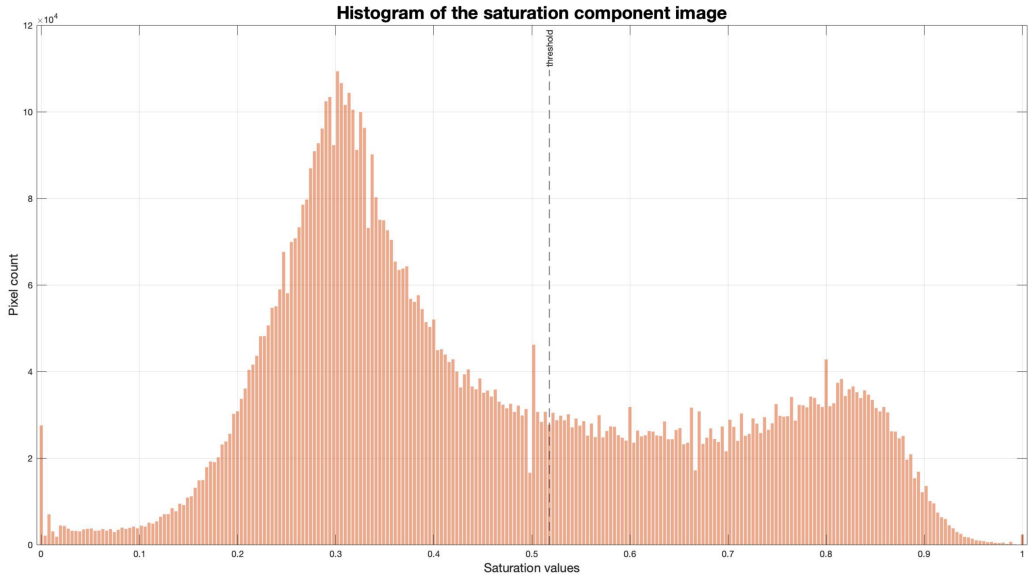
Figure 5.9 (d) – (f): L. clerkella leaf mine – segmentation and histogram using HSI colour space (part 2)



(g) Saturation component image



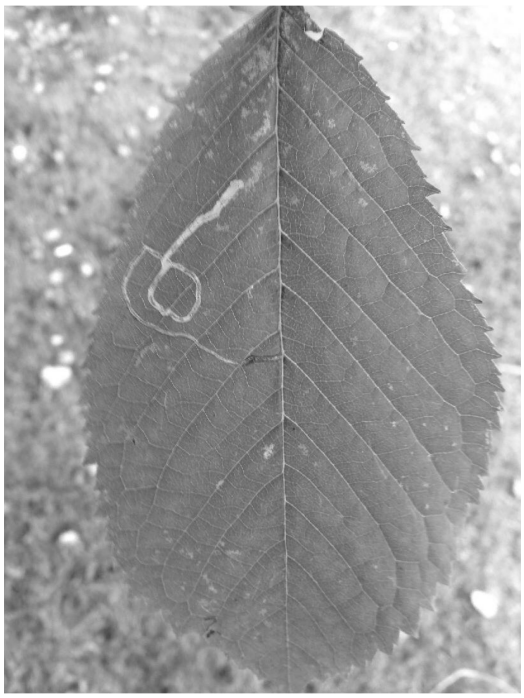
(h) Saturation mask



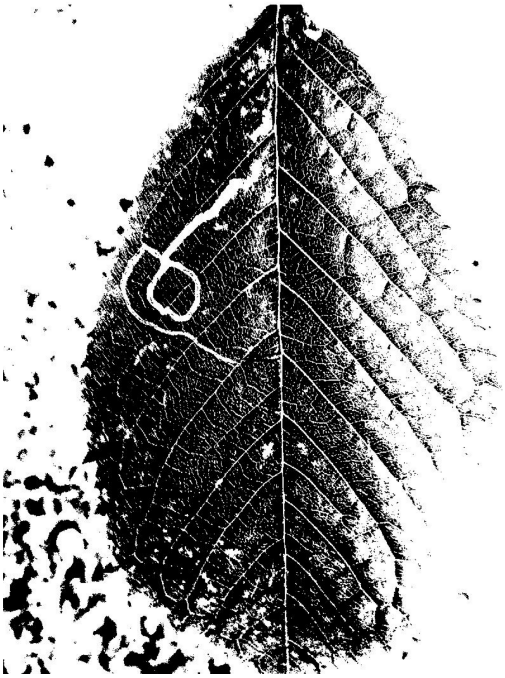
(i) Histogram of the saturation component image

Threshold value for creating the saturation mask: 0.5176

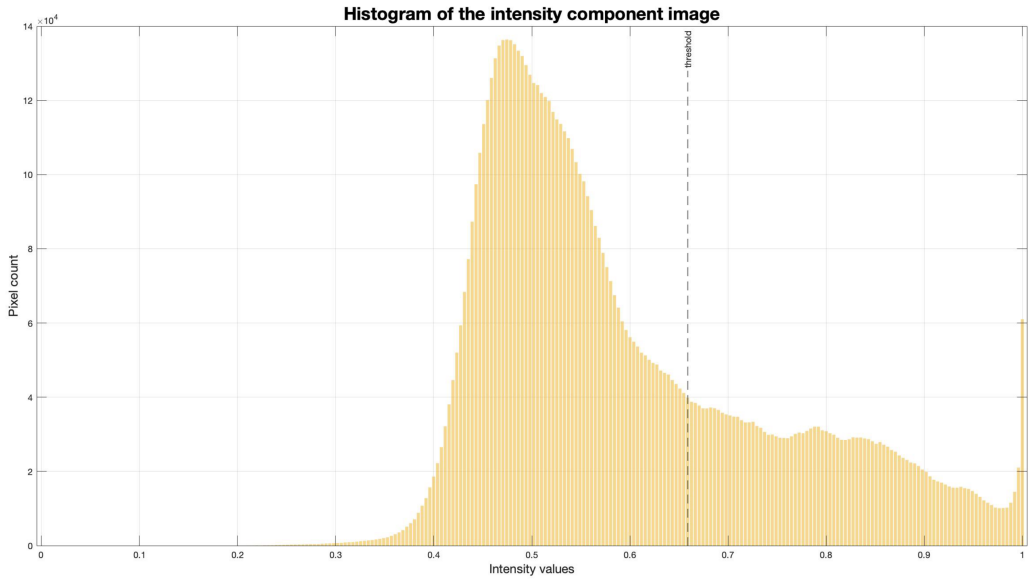
Figure 5.9 (g) – (i): L. clerkella leaf mine – segmentation and histograms using HSI colour space (part 3)



(j) Intensity component image



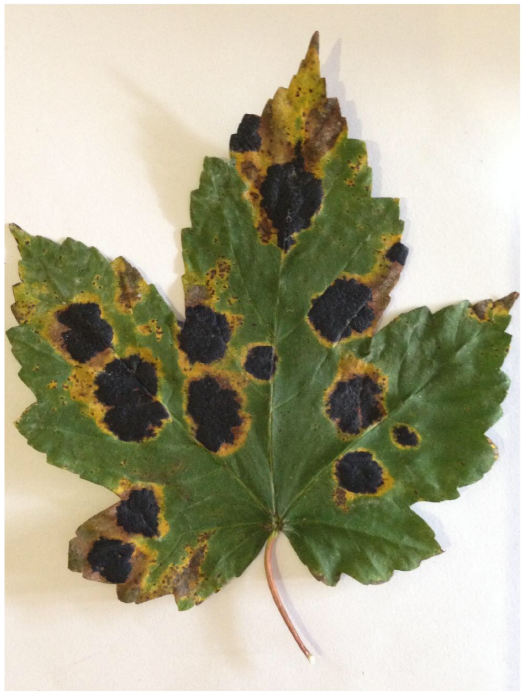
(k) Intensity mask



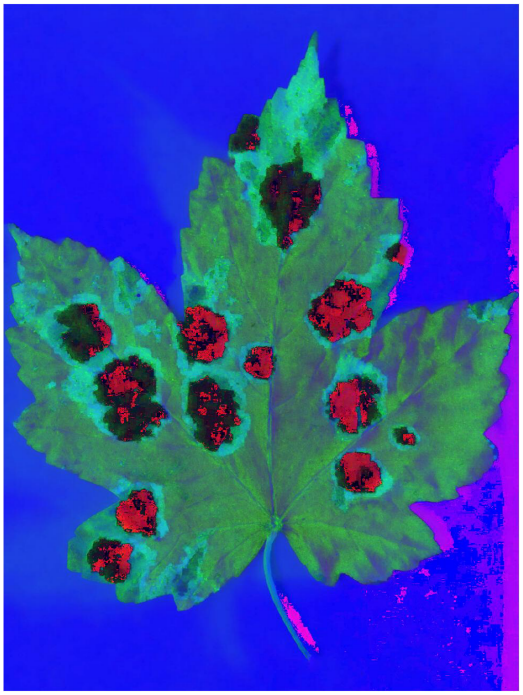
(l) Histogram of the intensity component image

Threshold value for creating the intensity mask: 0.6588

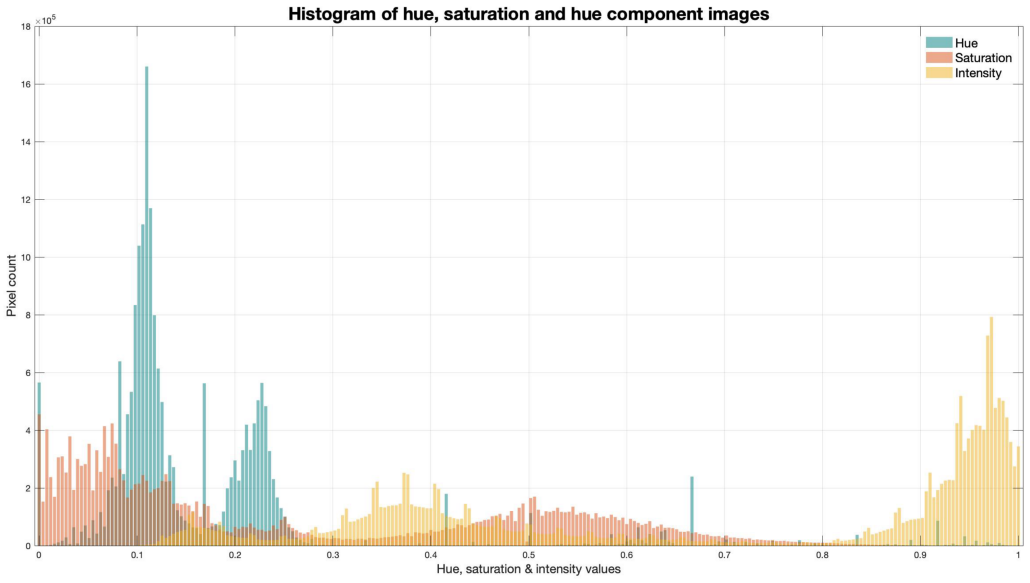
Figure 5.9 (j) – (l): L. clerkella leaf mine – segmentation and histograms using HSI colour space (part 4)



(a) Original RGB image



(b) Same image in HSI colour space

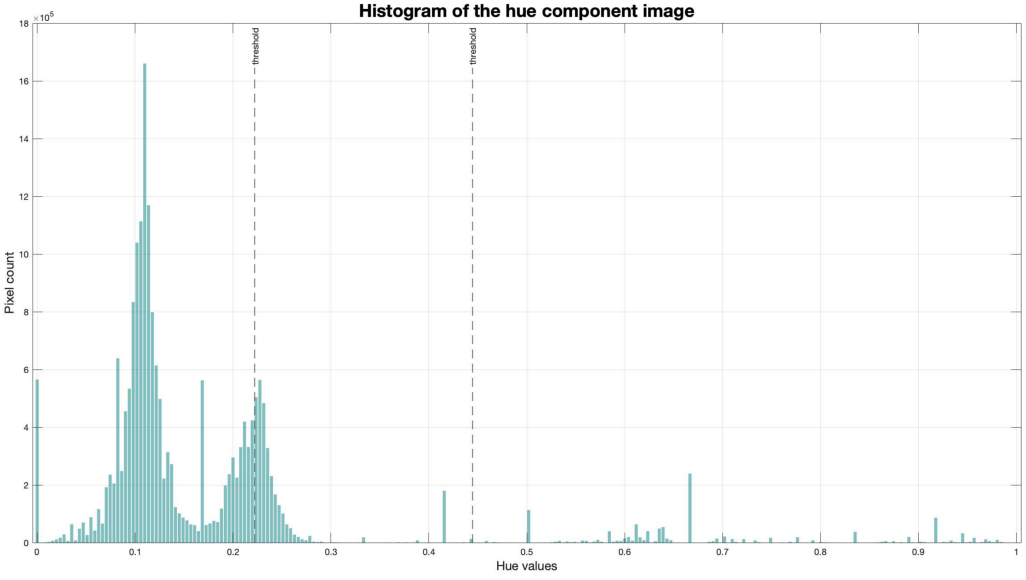


(c) Histogram of all 3 component images of HSI colour space

Figure 5.10 (a) – (c): Tar spots caused by R. acerinum – segmentation and histograms using HSI colour space (part 1)

(d) Hue component image

(e) Hue mask

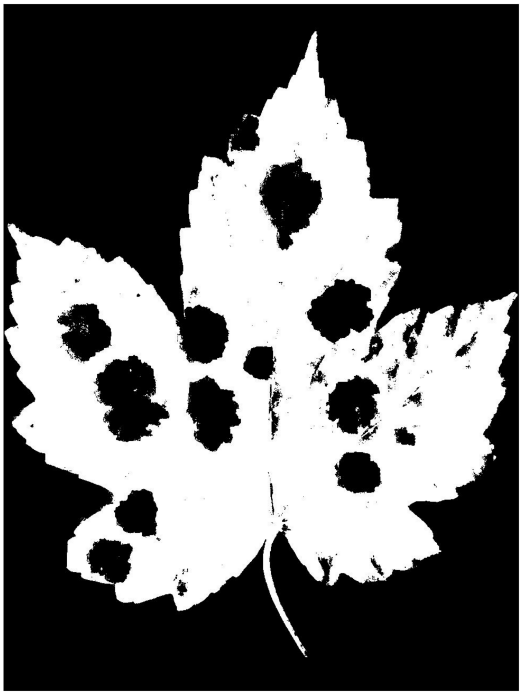


(f) Histogram of the hue component image

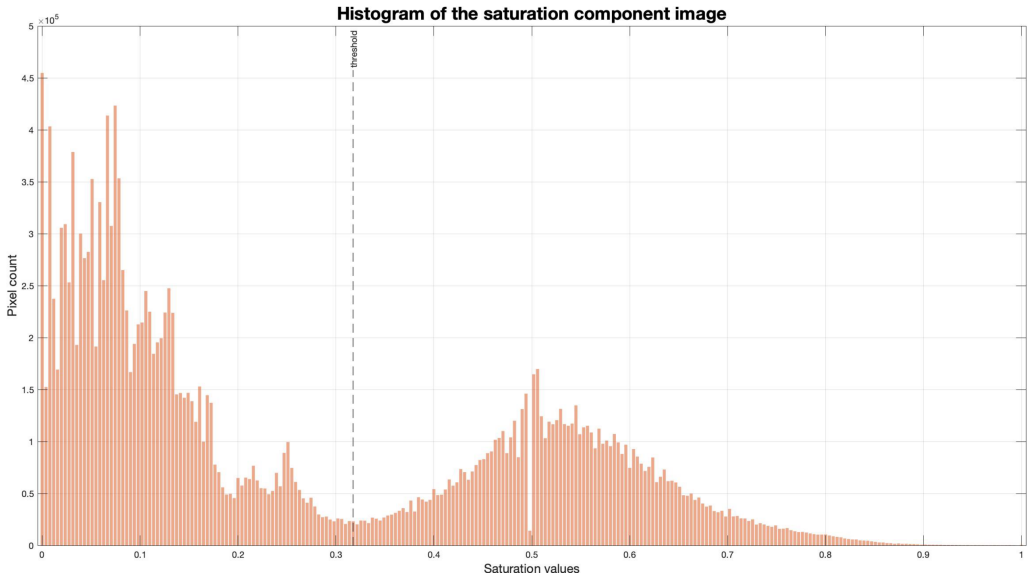
Threshold values for creating the hue mask: 80° and 160°

Figure 5.10 (d) – (f): Tar spots caused by R. acerinum – segmentation and histograms using HSI colour space (part 2)

(g) Saturation component image



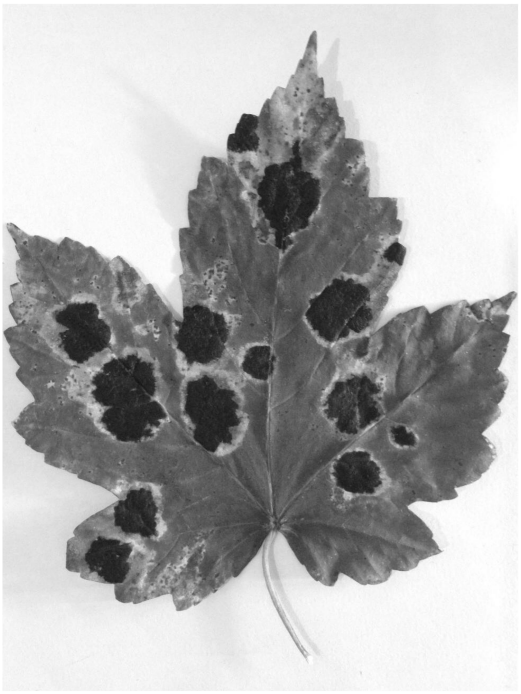
(h) Saturation mask



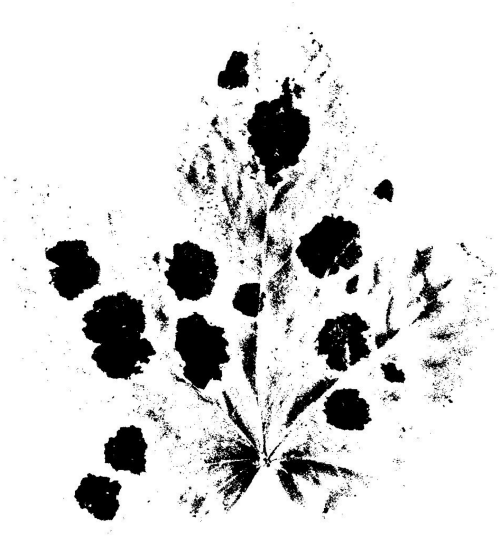
(i) Histogram of the saturation component image

Threshold value for creating the saturation mask: 0.3176

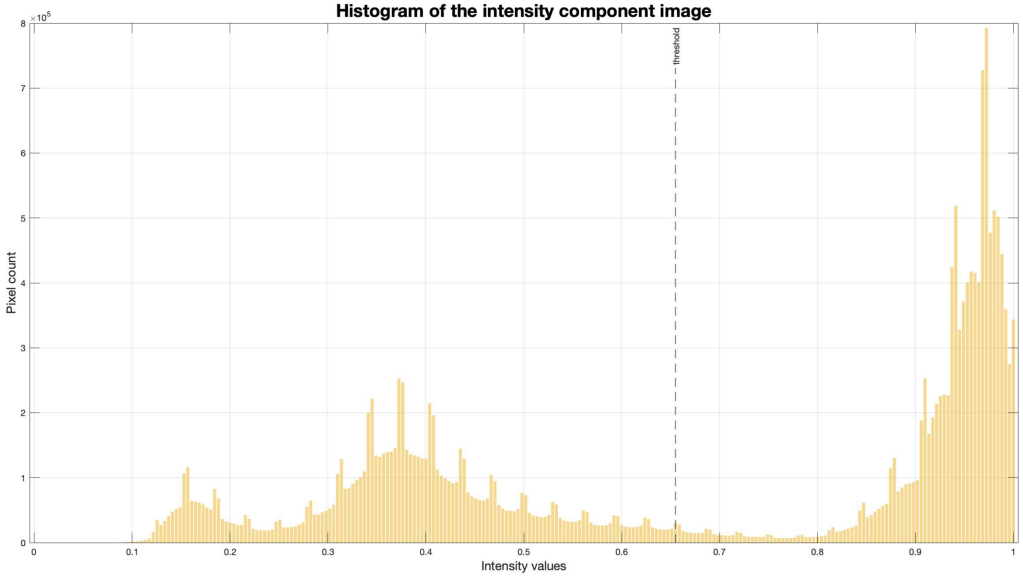
Figure 5.10 (h) – (i): Tar spots caused by R. acerinum – segmentation and histograms using HSI colour space (part 3)



(j) Intensity component image



(k) Intensity mask



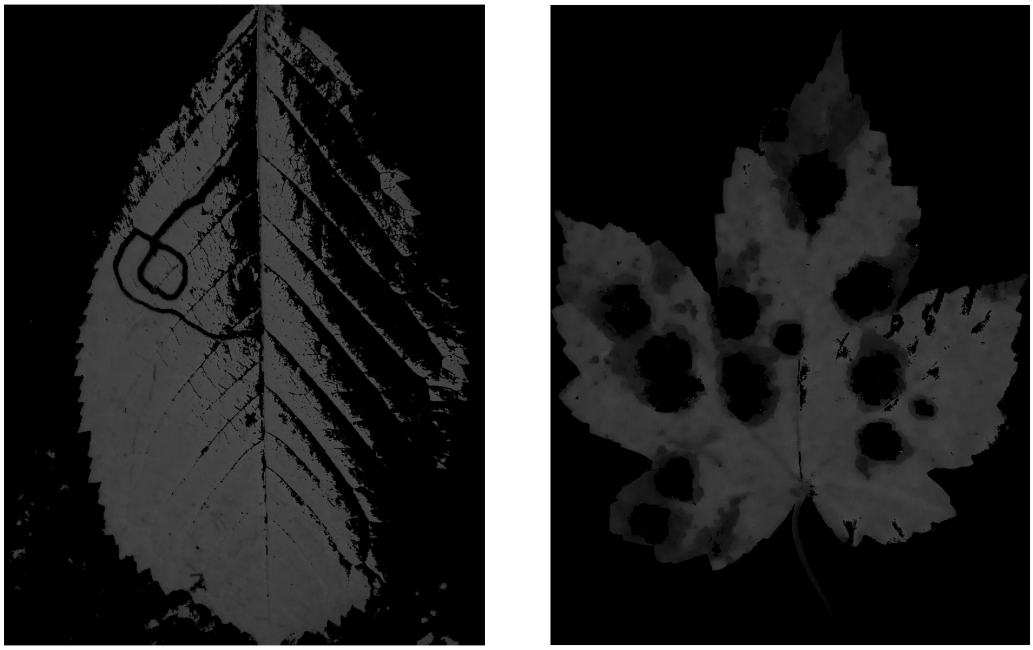
(l) Histogram of the intensity component image

Threshold value for creating the intensity mask: 0.6549

Figure 5.10 (j) – (l): Tar spots caused by R. acerinum – segmentation and histograms using HSI colour space (part 4)

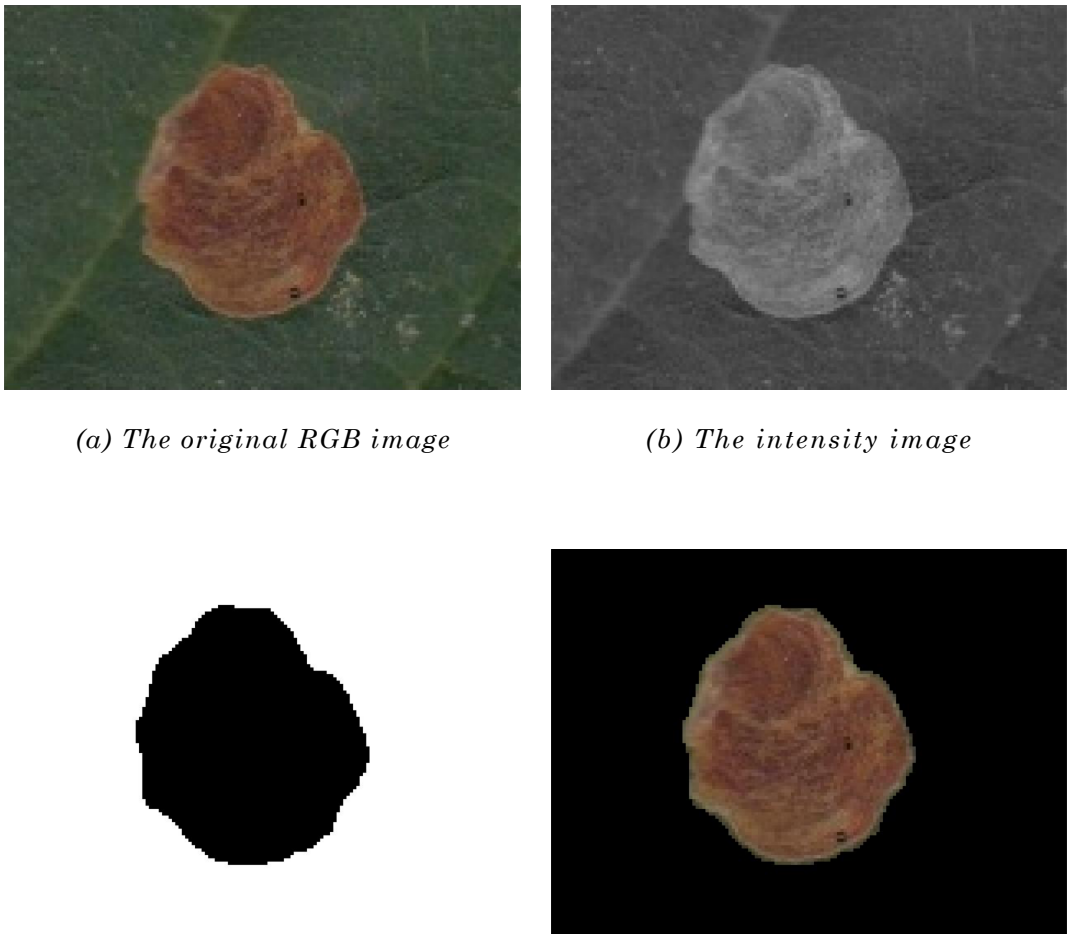
The binary masks are generated by thresholding the hue, saturation, and intensity component images respectively. The hue binary mask restricts the hue value to between 80° to 160° for the colour green. The binary masks are generated by thresholding the saturation and intensity component images. The threshold values are generally automatically in MATLAB using Otsu's method – any pixel with a value greater than the threshold was set to 1' (white), and the rest is set to '0' (black) [3, pp. 443 – 445].

Figures 5.11 (a) and (b) show the products of the saturation masks with the hue component images of figures 5.9 and 5.10. While capable of differentiating between "object" and "leaf" area, the backgrounds on which the leaves are placed upon are mistakenly grouped into the "object" area. This is mainly due to the chosen image acquisition method of digital photography, and can likely be helped with a smooth, solid background with adequate and constant lighting. Further thresholding techniques and edge detection has to be implemented in order to separate the background (where the leaf was laid upon), and the leaf itself.



(a) Product of figures 5.9 (e) and (g) *(b) Product of figures 5.10 (e) and (g)*

Figure 5.11: The products of the binary masks generated by thresholding the saturation component images in figures 5.9 and 5.10, and their hue component images



(a) The original RGB image

(b) The intensity image

(c) The binary mask

(d) RGB image with "masked" green pixels

Figure 5.12: Masking the green pixels

Figure 5.12 shows an example of the usage of a saturation mask (threshold value: 0.3647) to remove the green pixels of a scanned image of a *C. ohridella* leaf mine. Unlike the photographs used in figures 5.8 through 5.11, the image shown in figure 5.12 (a) was a scanned image and cropped using the steps described in section 5.2.1.

The image in figure 5.12 (a) was transformed into HSI colour space, and the intensity component, shown in figure 5.12 (b) was used for further processing. The saturation mask in figure 5.12 (c) was generated using the same method as the ones in figures 5.9 (h) and 5.10 (h). By selecting pixels which have low values in their blue and red channels, and have high values in the green channel, the

"mostly green pixels" can be removed, as shown in figure 5.12 (d). As the scanned image has uniformed background and lighting elements compared to the photographs used in figures 5.8 through 5.11, and because the scanned image was cropped before it was processed, the use of a saturation mask to remove the "background" pixels is more effective compared to the ones shown in figures 5.11 (a) and (b).

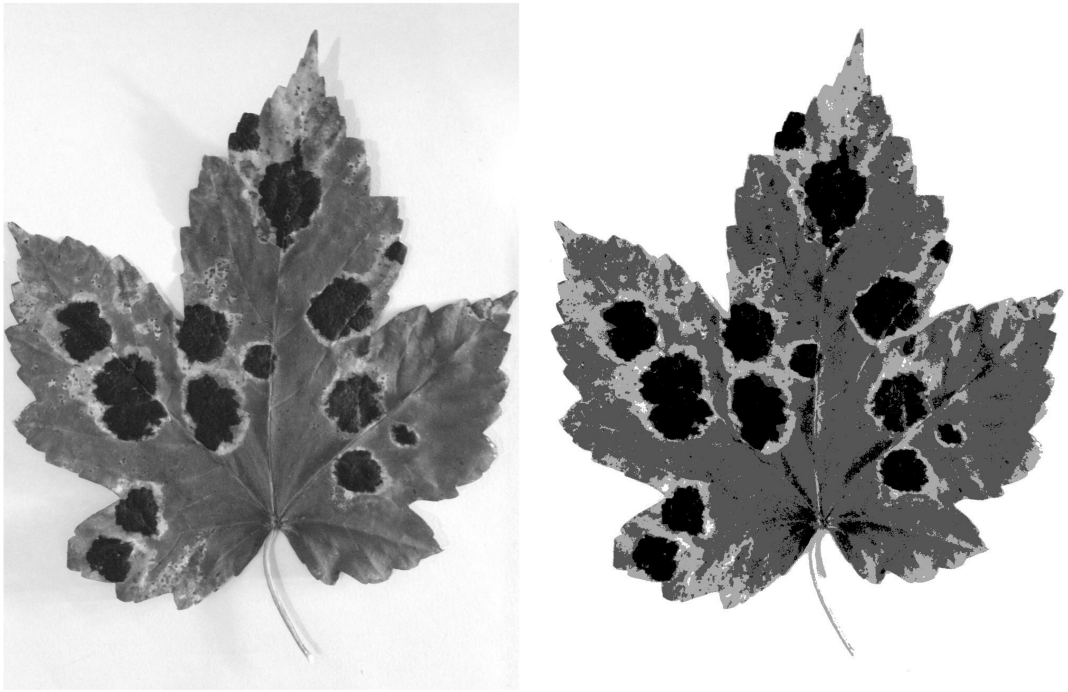
5.2.3 Image segmentation based on thresholding

It is worth mentioning here that the image segmentation techniques described in sections 3.1 through 3.3, as well as in this chapter, are inter-related. Therefore, the implementation of these techniques overlaps. For instance, the saturation and intensity masks discussed in section 5.2.2 were obtained using thresholding; and edge detection techniques were used to refine the binary masks, which will be further discussed in section 5.2.4. As mentioned previously, the HSI colour model has the advantage of separating colour and intensity information, making it ideal for greyscale image processing compared to the RGB colour space. The previous section described the processes of converting full-colour RGB images into HSI images, and the use of their monochrome component images; this section involves the transformation of greyscale images into their binary equivalent. Intensity thresholding using Otsu's method can be applied in MATLAB for global image thresholding and multilevel thresholding. The latter segments an image according to the threshold values determined using Otsu's method [230].

As with other image segmentation techniques, intensity thresholding divides an image into two (or more) sections; for binary images, the two sections are: image foreground, indicating the "object area"; and image background. When investigating the possibility of identifying leaf mines or tar spots, "object area" refers to an area with a leaf mine or tar spot; whereas the background, or "leaf area" refers to an area of the leaf not affect by leaf mines. Using Otsu's method, by comparing the intensity of the pixel and the predetermined threshold value, a certain pixel location would be assigned the value of '1' (white) or '0' (black) using an intermediate output of a single global threshold. In this instance, all the pixels within the "object area" would ideally be identified as image foreground, while all other pixels would be classed as image background.

Suppose an intensity histogram that corresponds to a resulting image from the previous subsection, the intensity values of the foreground and background pixels can be separated into two main groups, as discussed in section 3.2.1. For example, the histogram in figure 5.10 (i) has two main modes, which corresponds to light objects on a dark background. The global threshold value is affected by the following factors: (1) the distance between the peaks; (2) the amount of noise present in the image; (3) relative sizes of background and objects; and (4) the uniformity of illumination and reflectance. If the intensity distributions of foreground and background pixels are adequately distinct, a single global threshold can be applied over the whole image.

However, a constant threshold cannot be effectively applied over an entire image. An example of such scenario is the histogram in figure 5.8 (l). The three dominant modes of the histogram represent a more complex thresholding problem. The three classes, consisting of three intensity intervals, can be separated by dual thresholds. Figure 5.13 (b) shows the multilevel thresholding result of the intensity image in figure 5.13 (a) using two threshold levels.

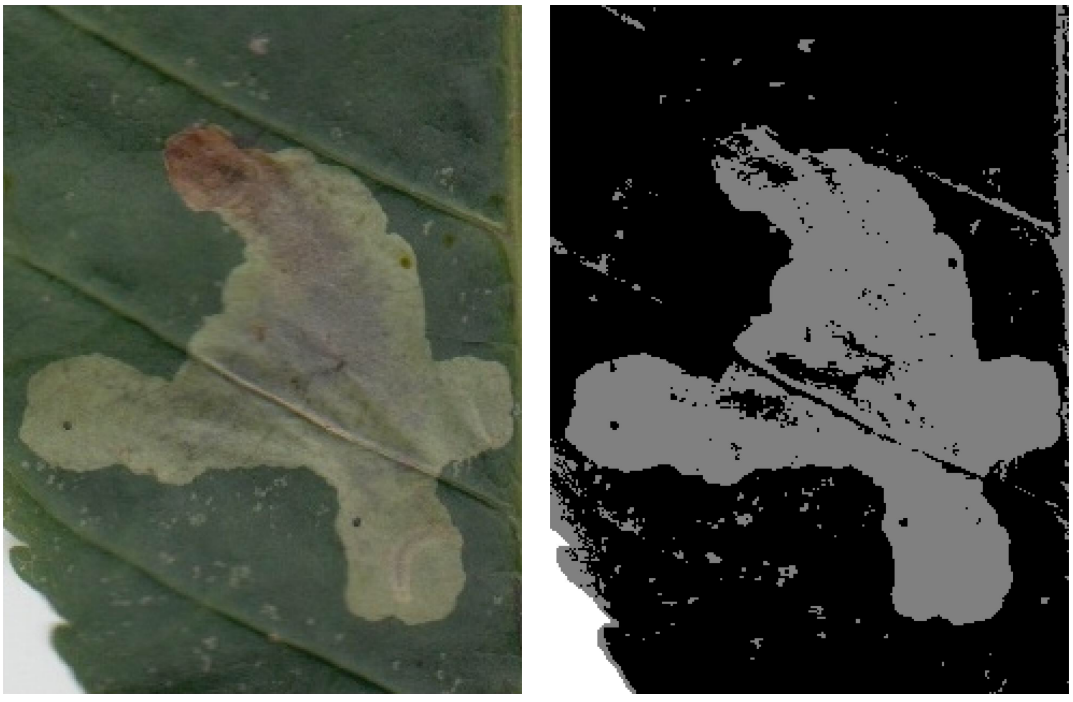


(a) Intensity image *(b) Thresholding result*

Figure 5.13: Multilevel thresholding

Using this approach, it is possible to segment an image using an arbitrary number of thresholds. In theory, this would be an ideal way to improve the segmentation method for ambiguous areas; in practice, however, intensity thresholding becomes less reliable as number of threshold increases, and it becomes a pattern recognition problem [3, pp. 752 – 756].

An original full-colour RGB image of a *C. ohridella* leaf mine is shown in figure 5.14 (a), and figure 5.14 (b) depicts the thresholding result using dual thresholds, in an attempt to separate the pixels into three classes – leaf mine (shown in grey), the leaf itself (black), and the background (white). As seen in figure 5.14 (b), the background was successfully separated into a different group from the rest; however, some leaf pixels were erroneously classified as part of the leaf mine and vice versa. To reduce the average error incurred when separating pixels into different groups, the images are smoothed before thresholding; and to decrease to influence of background and object sizes, the shape of the histogram is improved by focusing on edges between objects and background; Laplacian operators are applied to extend the valley between the peaks on a histogram.



(a) Original RGB image *(b) Thresholding result*

Figure 5.14: Multilevel thresholding of a C. ohridella leaf mine

In addition to preprocessing techniques like image smoothing and the usage of edges, intensity thresholding can be improved with variable thresholding; for instance, image partitioning complements Otsu's method by dividing the image into non-overlapping sub-images, and analysing the histogram of these sub-images. An application of multivariable thresholding is in colour imaging, where the red, green and blue components form a composite RGB image. Each pixel of the RGB image can thus be characterised by three separate value (r, g, b), and be represented by the 3D vector $\boldsymbol{z} = (z_1, z_2, z_3)^T$, with the RGB colours at a certain pixel being the components. To segment an image based on a parameter based on a specified colour range,

$$g = \begin{cases} 1 & if\ D(\mathbf{z}, \mathbf{a}) < \mathrm{T} \\ 0 & otherwise \end{cases} \tag{5.1}$$

where T is the threshold, $D(\boldsymbol{z}, \mathbf{a})$ is the distance between an arbitrary colour point $\mathbf{z}$, and the average colour within that specified colour range $\mathbf{a}$.

5.2.4 Edge-based segmentation

In the previous section, it was noted that edge detection is often used in conjunction with thresholding. Unlike region-based approaches such as thresholding, edges partition an image based on sudden changes in intensity. The approaches of image segmentation were discussed in section 3.2. The combination of methods from both region-based and edge-based approaches improves segmentation performance. Image segmentation based on morphology is also discussed in this section. This approach combines merits of region-based and edge-based techniques.

As mentioned in the previous section, some background pixels can be mistaken as "objects" and vice versa. These small "objects" could be dust or leaf lesion not related to the leaf mine or tar spot [196]. The falsely positive classified pixels are eliminated using morphological operations, as are objects on the border. An example is shown in figure 5.15. Objects connected with the border of the image are removed, as shown in figure 5.16. As *a priori* knowledge indicates that the leaf mine or tar spot of interest should be roughly in the centre of the image, therefore any objects connected to the border of the image can be removed [143].

The Sobel operator was chosen as a suitable edge operator for locating edge pixels following calibration and experiments. Pixels along the edge, *E*, has to satisfy the condition that any neighbourhood centred at *E* must contain at least a '1' and a '0', which indicates a local change of intensity. The abrupt change of intensity often associates with some physical boundaries of various objects in binary images, hence this edge detection method is most suitable for images with sharp edges and little noise.

(a) Binary image before removing a small object

(b) Binary image after removing a small object

Figure 5.15: Removing a small object from a binary image

Figure 5.16: A scanned image of a C. ohridella leaf mine

The red line represents the edge of the leaf mine of interest. Note that the system excluded the mine at the bottom of the image, due to its connection to the border of image

There are limitations to existing image segmentation methods: (1) the number of classes or regions is usually pre-defined, and (2) pre-processing techniques are incorporated to reduce or remove noise. Common edge detection methods are easily affected by the alteration in brightness of neighbouring objects, as edge detection cannot be achieved independently of direction. When the pixels of a binary image are divided into "object" (foreground) and background pixels, errors are inevitable. For example, stems or more prominent leaf veins can sometimes "blend into" lesions areas if the variance of colour is not significant, thus a pixel which should belong to the background could be mistaken as foreground and vice versa. Additionally, it is often necessary to connect all the edge pixels if continuity is affected by noise.

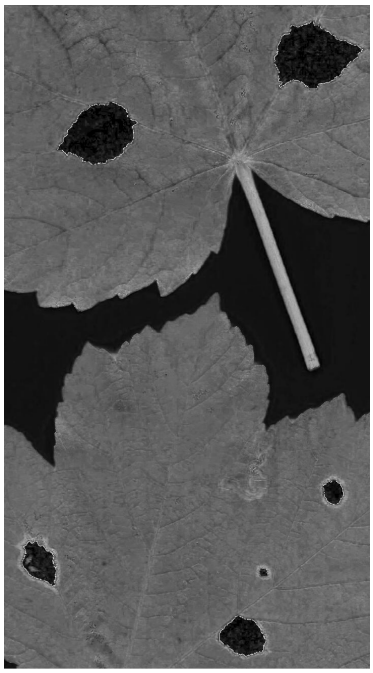
Morphological operations are then used to enhance traditional edge detection methods – with the use of structuring elements, erosion removes pixels from the edge, whilst dilation adds pixels to the edge of the object. A structuring element is added to the input image for both erosion and dilation, and can be in the shapes of diamond, disk, line, octagon, rectangle, square, cube, cuboid, sphere or arbitrary [231]. Figure 5.17 shows the flat diamond structure element chosen after experimentation. For images with multiple objects or ones with many ambiguous areas, user interaction can be used to simplify this process. This method is suitable for analysing leaves with multiple leaf mines or tar spots.

Figure 5.18 shows an example of how tar spots caused by *R. acerinum* are detected, in various stages. Figure 5.14 (a) shows the original RGB image and figure 5.18 (b) shows the hue component of full-colour in HSI colour space. Figure 5.18 (c) shows the binary image after intensity thresholding, but before the "object" areas were filled; and the result of edge detection using the Sobel operator is shown in figure 5.18 (d).

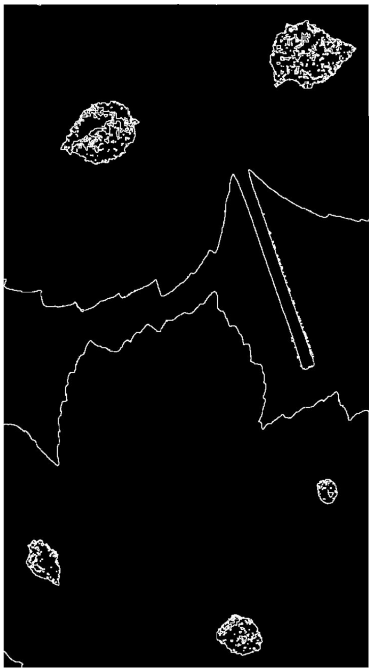
```
0     1     0
1     1     1
0     1     0
```

Figure 5.17: A flat diamond structure element object, containing 5 neighbours, the distance from the structuring element origin to the points of the diamond is 1

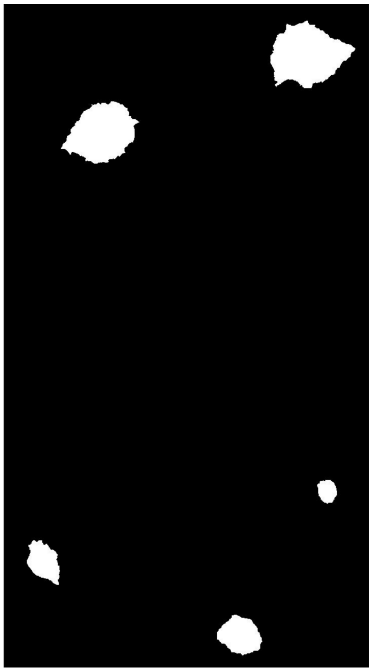
(a) Original RGB image



(b) Intensity image



(c) *Binary image, before the user selects the areas to fill*



(d) Binary image, showing the segmentation results

Figure 5.18: The process of identifying 5 spots caused by R. acerinum

The RGB image in (a) is a scanned image of a sycamore leaf.

5.3 Feature extraction

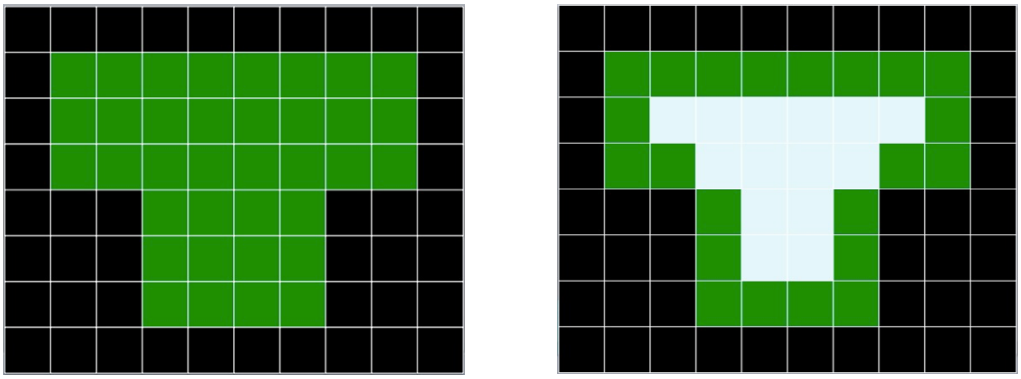
One of the objectives of this project is to extract features from images to facilitate classification in later stages. This is a systematic way to compare the samples, and to provide a quantitative approach for the classification stage and result analysis. Kang *et al.* [48] suggested that feature extraction on images allows suitable details to be identified and hence plays an essential part in automating species identification. Mayo and Watson [44] stated feature extraction is the most important factor for any image identification system involving machine learning; and O'Neill [2] indicated that feature extraction is particularly beneficial for analysing two-dimensional images. By extracting morphological features of corresponding domain and training the system, many automated systems went on being developed and succeed in identifying various animals and plant species.

Features can be categorised as global or local – global features consider the entire image; whereas local features focus on smaller portions on the image [22]. Taking both global and local features into account allows the images to be considered as a whole, while details in smaller sections were not excluded. Global features include: eigenspace matching [232], colour histogram [233] and receptive field histograms [234]. Different from global features, Wen *et al.* [22] characterised local features as ones that base on smaller sections of the image, such as edge, corner, entropy, curvature, region and ridge. There is no definite solution to whether global or local features are more desirable, as this depends on the aims and approach of the research. Local features are highly compatible with images taken in non-controlled environment, as suggested by Larios *et al.* [49]; and this argument was supported by Wen *et al.* [22]. Due to local features being less prone to occlusion, hence the effect of scaling and rotation could be diminished. Using exclusively local features, Wen *et al.* achieved an average success classification rate of 70% over six methods [22]. Mayo and Watson used a combined approach by incorporating local and global features [44], as did Flandin [47].

The geometric features obtained through the image processing section of the proposed system include: (1) area of leaf mine or tar spot (the total number of foreground pixels); (2) perimeter of leaf mine or tar spot (using edge detection methods); (3) centroid (the centre of the object in question); (4) length of major

and minor axis (the axes of the ellipse with the same central moments as the object); (5) orientation (the angle between the x-axis and major axis of the ellipse, for computing the distance among leaf mines). Perimeter and area, in particular, are the foundation in the analysis of growth rate.

There are two main methods to calculate the area of an object using MATLAB – based on the perimeter of an object; and a weighted estimation based on the shape of the object. The perimeter of an object is the total distance between each adjoining pair of pixels surrounding the border of the object region. The area of the object, therefore is the total number of pixels within the border [235]. The green pixels in figure 5.19 (a) made up the area of the region; while the green pixels in figure 5.19 (b) indicates the border, which must be continuous.



(a) *(b)*

Figure 5.19: Calculating area and perimeter in MATLAB [232]

(a) *(b)*

Figure 5.20: (a) 4-connected connectivity and (b) 8-connected connectivity [234]

With the second method, the area of a particular object is estimated by consider any non-zero numeric input as '1' [236]. It uses a 2×2 neighbourhood to determine the area of each pixel. Using this approach, the perimeter of an object based on connectivity between pixels, as demonstrated in figure 5.20. The connectivity used in two-dimensional images is either 4 or 8; with 4-connected connectivity, pixels are connected if their edges touch, and two or more adjoining pixels are considered part of the same object whether connected horizontally or vertically; whilst with 8-connected connectivity, pixels are connected if their edges or corners touch, and two or more adjoining pixels are counted as part of the same object if they are connected horizontally, vertically or diagonally [237]. The two methods may return slightly different results, as the first method returns the actual number of pixels of the object, while the latter provides an estimation of the total area of all objects in the image.

5.4 Summary

This chapter discussed the processes that were implemented in the proposed approach for detecting leaf mines and fungal spots from images. This includes the methodology in data collection; how the digital images were acquired; preprocessing techniques to prepare the images; various image segmentation methods, including the use of colour models, thresholding and edge detection; and how features could be extracted from the segmented images. Figure 5.21 shows a flow diagram of the image processing steps discussed in this chapter.

Section 5.1 provided an overview of data collection methods, and a brief comparison between image acquisition using digital cameras and flatbed scanners. In general, flatbed scanners are well-adapted for providing consistency in background and lighting. The use of digital photography, meanwhile, is suitable for its usability in collecting samples in the natural environment. The preprocessing steps and various image segmentation methodologies were included in section 5.2. The morphological features extracted using the steps described in section 5.3, can then be used as parameters to possibly classifying leaf mines based the growth of infestation. This will be further discussed in the next chapter.

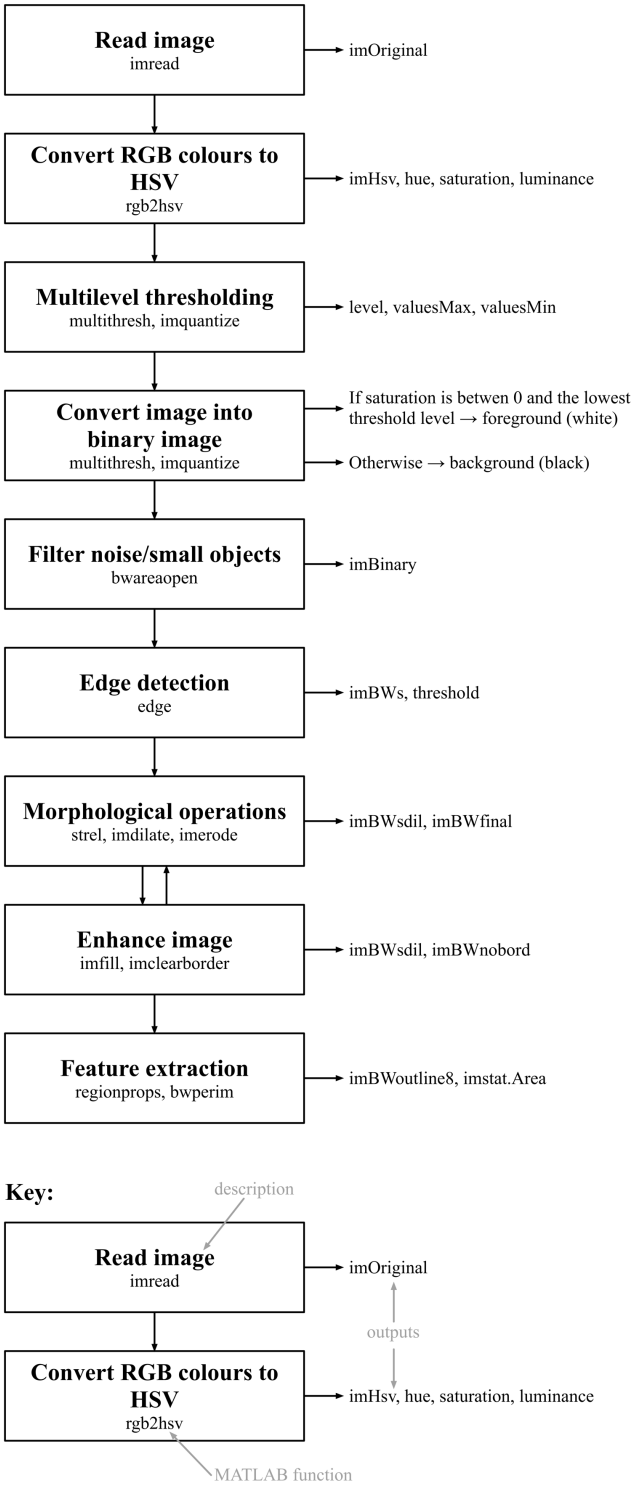


Figure 5.21: Flow diagram of the image processing section of the system

Chapter 6

Data analysis

The previous chapter described the implementation of digital image processing techniques in the proposed approach for detecting leaf mines and tar spots from digital images. This chapter presents the experimental results of the processes discussed in chapter 5, including data collection, image segmentation, edge detection, and feature extraction. By repeating the image processing steps, it is possible to measure the daily area of a certain leaf mine over several days. This chapter also investigates the possibility to estimate the growth of *C. ohridella* leaf mines, and the use of morphological features as parameters in a leaf mine classification system.

Continuing from the last chapter, "object area" refers to the pixels representing the leaf mines or tar spots, and the "leaf area" indicates the pixels representing the leaves. All the samples of leaves in this chapter were digitised using the acquisition methods discussed in section 5.1; and that the preprocessing and segmentation steps described in section 5.2 have taken place.

6.1 Segmentation results

Figures 6.1 through 6.4 reiterate the segmentation results for three species of leaf mines – *C. ohridella*, *L. clerkella,* and *P. ilicis*; as well as the fungal *R. acerinum*. As mentioned in section 1.3, *C. ohridella* was originally chosen as a primary focus, with *L. clerkella,* and *P. ilicis* being introduced at a later stage to introduce variation and to avoid overfitting the approach for a single species of leaf mines.

Likewise, samples of *R. acerinum* were collected to explore the plausibility of adapting the current method to detect damaged caused by a plant disease. The leaf mines or tar spots in figures 6.1 to 6.4 are indicated by the red lines.

Figures 6.1 (a) and (b) reiterate the segmentation results of *C. ohridella* leaf mines on horse chestnut leaves. Figure 6.1 (a) shows the result of the detection of a single leaf mine. Figure 6.1 (b) shows the edges of multiple leaf mines, which involved an extra step that required user interaction, as mentioned in section 5.2.4. The user input acted as a confirmation as to where the leaf mines might be located.

Figure 6.2 (a) and (b) likewise display two examples of the edge detection results of *R. acerinum* on maple leaves. Figure 6.2 (a) shows the outcome with a single tar spot, and figure 6.2 (b) shows the result with two tar spots. For all the samples in figures 6.1 and 6.2, a flatbed scanner was used as the image acquisition method.

The segmentation of *L. clerkella* and *P. ilicis* leaf mines proved to be more complicated due to the shape of the leaf mine or leaf, and issues caused by the image acquisition method. Unlike the samples used in figures 6.1 and 6.2, the original images for figures 6.3 and 6.4 were digital photographs.

Figure 6.3 shows the segmentation result of three *L. clerkella* leaf mines on an apple leaf. The current samples of apple leaves were photographed outdoor under natural lighting, with a non-uniformed background. Apple leaf mines present an additional challenge for segmentation due to the shape (gallery mines instead of blotches mines), and the insufficiently distinct in intensity between the leaf mines and the leaf.

Figure 6.4 displays the edge detection result of a *P. ilicis* leaf mine on a holly leaf. The colours within a holly leaf mine can differ depending on the developmental stage, which complicates the segmentation process. Additionally, holly leaves are not suitable for scanning due to their three-dimensional shape. However, they are also not suited for photographing due to their glossy appearance, which reflect light. These reasons lead to inaccuracy of the current edge detection method.

(a) *(b)*

Figure 6.1: Segmentation results of C. ohridella leaf mines

(a) *(b)*

Figure 6.2: Segmentation results of R. acerinum tar spots

In all figures the red lines indicate the edges of the leaf mines or tar spots.

Figure 6.3: Segmentation results of L. clerkella leaf mines

Figure 6.4: Segmentation results of a P. ilicis leaf mine

In all figures the red lines indicate the edges of the leaf mines or tar spots.

6.2 Daily growth of *C. ohridella* leaf mines

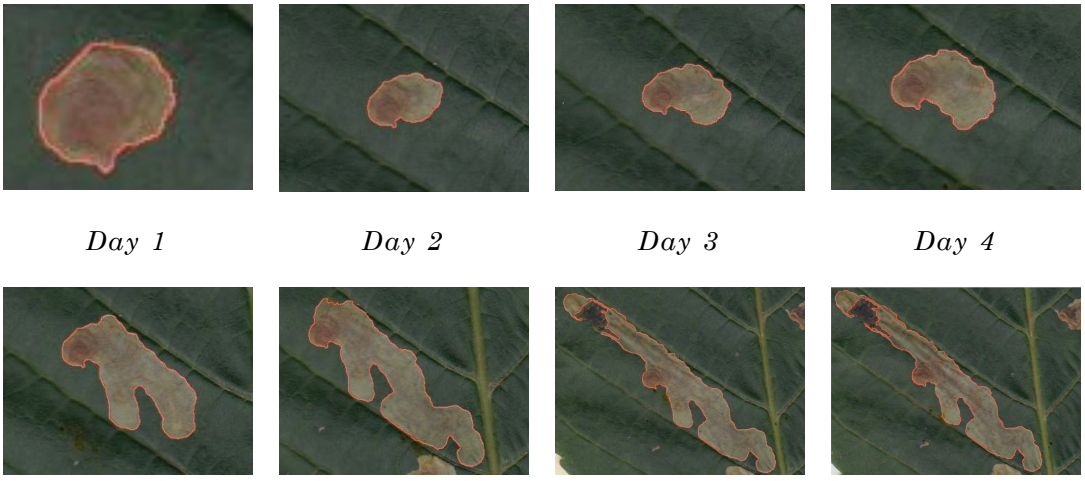
One of the objectives of this project is to extract features from digital images of leaves, and to explore the use of these features as parameters for an automated classifier for leaf mines. Feature extraction is a systematic way to compare the samples and to provide a quantitative approach for the classification stage.

Specifically, for a system aimed at detecting leaf mines from digital images, morphological features are used as examples to explore some of the tools and methodologies in this section. Although additional features were briefly mentioned in section 5.3, only the perimeter and area are further discussed in this section. The chosen method determines the perimeter based on the continuity of pixels surrounding an object. From this, the area of the same object is the total number of pixels within that border.

By repeating the processes described in chapter 5 daily over a number of days, it is possible to find out the area of a leaf mine on a succession of days. As mentioned in section 5.1, samples of horse chestnut leaves were kept in airtight plastic containers at room temperature. The method for image acquisition for these samples was scanning because of the consistent background and lighting. The leaves were put through a scanner every day to observe how much the leaf mines had changed overnight. The increase in area of a leaf mine can be presumed to be the ongoing damage caused by a larva. When the area of a leaf mine remains approximately constant, it could be due to one of the following reasons: (1) the larva is dead; or (2) it is in one of the non-feeding stages; or (3) it has reached maturity and left the leaf as an adult moth.

Figure 6.5 shows the daily growth of a *C. ohridella* leaf mine (leaf mine #1 in the sample set). Noted that the leaf mine for day 3 appears to be smaller than the one on the previous day, but the area had in fact increased by 4.5 pixels. This is due to the manual cropping of the scanned image. However, this does not affect the area of the object as the resolution setting of the flatbed scanner is the same for all 8 days. As the area of the leaf mine grew by 0.0126% from day 2 to 3, this was possibly due to the larva being in a non-feeding stage. It is also noted that the edge detection on day 8 was slightly affected by the uneven colour of the mine.

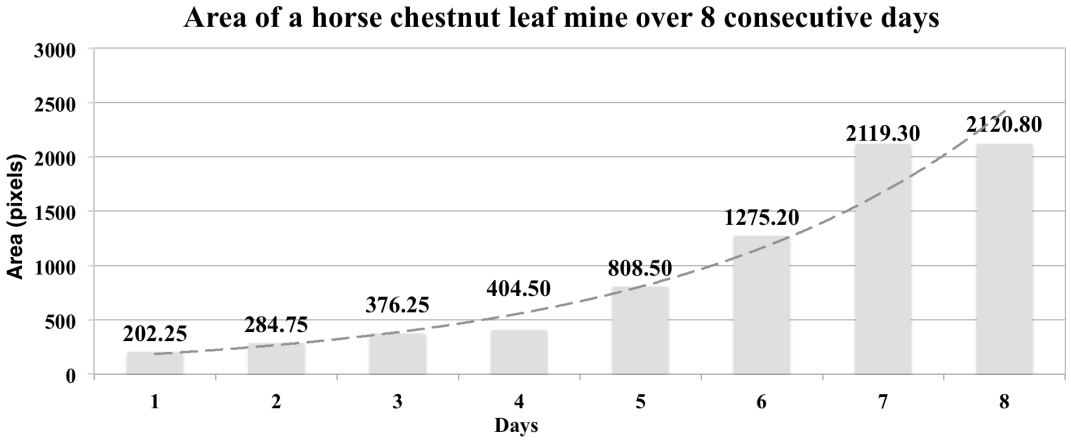
Figure 6.6 shows another sample of *C. ohridella* leaf mine ((leaf mine #2). The area of leaf mine changes little from day 2 to day 4, supported by the evidence where the damaged area continued to grow from day 4 onwards; while from day 7 to day 8 it could be the case where the larva had grown into an adult moth.



Day 1 *Day 2* *Day 3* *Day 4*

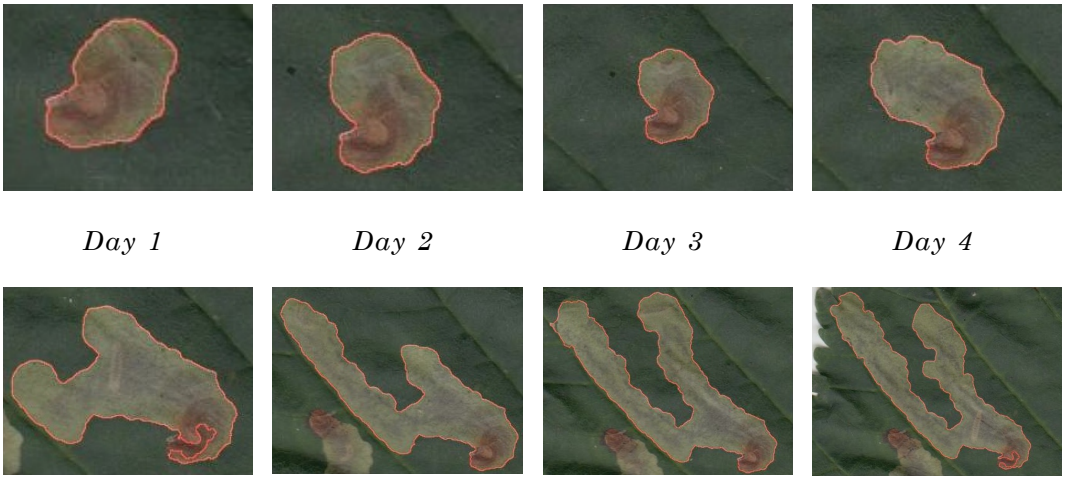
Day 5 *Day 6* *Day 7* *Day 8*

(a) Daily development of a leaf C. ohridella leaf mine, with edges drawn in red



(b) A graph representing the growth in area of the same leaf mine along with and dashed exponential trend line

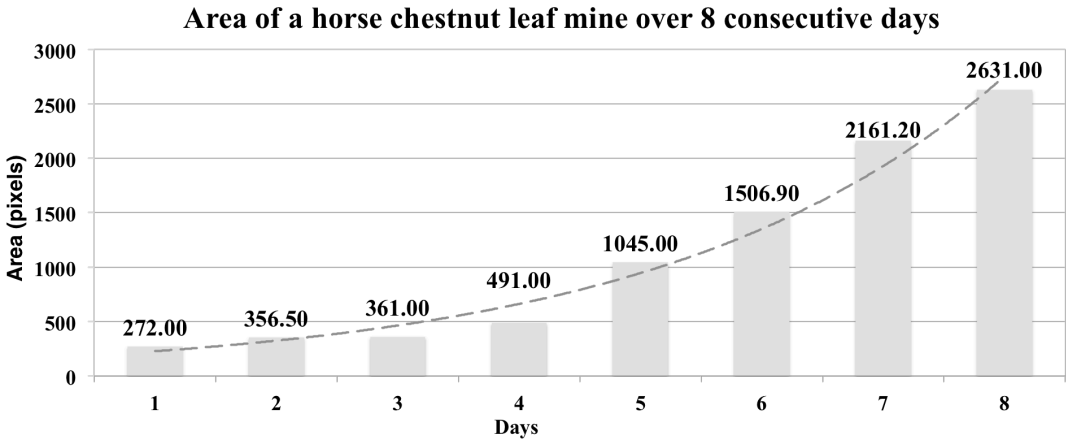
Figure 6.5: An example of a C. ohridella leaf mine



Day 1 *Day 2* *Day 3* *Day 4*

Day 5 *Day 6* *Day 7* *Day 8*

(a) Daily development of a leaf C. ohridella leaf mine, with edges drawn in red



(b) A graph representing the growth in area of the same leaf mine

Figure 6.6: An example of a C. ohridella leaf mine

The estimate area of the leaf mine on day 8 was hindered by the image segmentation method used and could possibly be improved by procedures described in subsection 6.3.3.

Date	Area (pixels)	Growth rate (%)
06/06/14 (Day 1)	202.25	n/a
07/06/14 (Day 2)	284.75	40.79%
08/06/14 (Day 3)	376.25	32.13%
09/06/14 (Day 4)	404.50	7.51%
10/06/14 (Day 5)	808.50	99.88%
11/06/14 (Day 6)	1275.20	57.72%
12/06/14 (Day 7)	2119.30	66.19%
13/06/14 (Day 8)	2120.80	0.07%

Table 6.1: The area of the C. ohridella leaf mine #1 shown in figure 6.5 (a)

Date	Area (pixels)	Growth rate (%)
06/06/14 (Day 1)	272.00	n/a
07/06/14 (Day 2)	356.50	31.07%
08/06/14 (Day 3)	361.00	1.26%
09/06/14 (Day 4)	491.00	36.01%
10/06/14 (Day 5)	1045.00	112.83%
11/06/14 (Day 6)	1506.90	44.20%
12/06/14 (Day 7)	2161.20	43.42%
13/06/14 (Day 8)	2631.00	21.71%

Table 6.2: The area of the C. ohridella leaf mine #2 shown in figure 6.6 (a)

Another example (leaf mine #3) is shown in figure 6.7 – in this instance, the area of the leaf mine had barely increased beyond day 4. As the mine appeared to be quite developed, it is possible to assume the larva had fully developed by that point and left the leaf as a result. The examples in figures 6.8 (leaf mine #4) and 6.9 (leaf mine #5) show similar results. The area of the leaf mines on day 5 had slightly decreased from day 4, but this is more likely to be an error due to the image acquisition or segmentation process. More examples of *C. ohridella* leaf mines are shown in figures 6.10 through 6.18 (leaf mines #6 – 14).

Day 1

Day 2

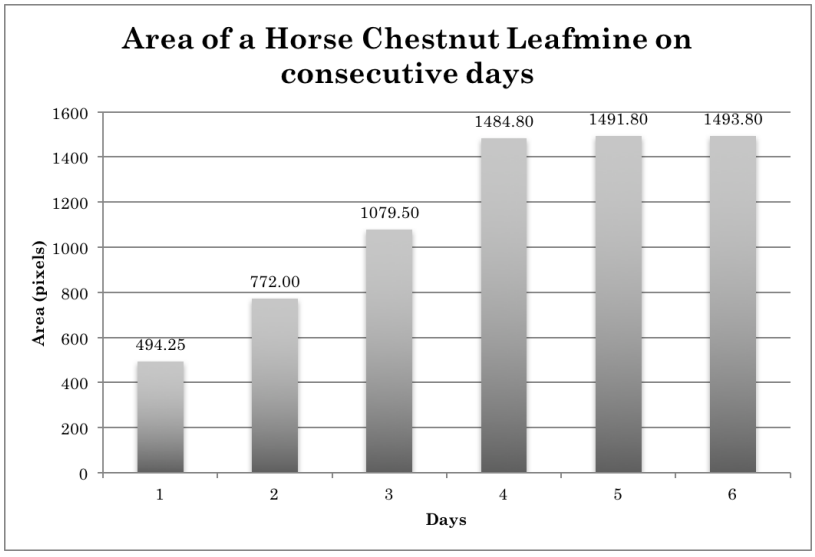
Day 3

Day 4

Day 5

Day 6

(a) Leaf mine example #3

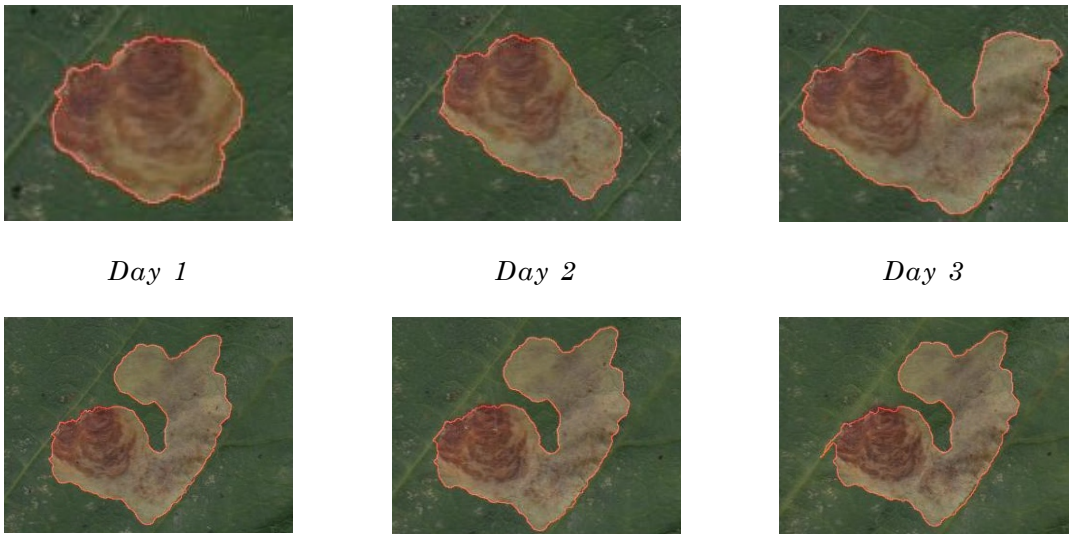


(b) Area of horse chestnut leaf mine #3 on 6 consecutive days

Figure 6.7: An example of a C. ohridella leaf mine

Date	Area (pixels)	Growth rate (%)
14/06/14 (Day 1)	494.25	n/a
15/06/14 (Day 2)	772.00	56.20%
16/06/14 (Day 3)	1079.50	39.83%
17/06/14 (Day 4)	1484.80	37.55%
18/06/14 (Day 5)	1491.80	0.47%
19/06/14 (Day 6)	1493.80	0.13%

Table 6.3: Area of horse chestnut leaf mine #3 on 6 consecutive days



Day 1 *Day 2* *Day 3*

Day 4 *Day 5* *Day 6*

Figure 6.8: Leaf mine example #4

Date	Area (pixels)	Growth rate (%)
14/06/14 (Day 1)	415.63	n/a
15/06/14 (Day 2)	575.13	38.38%
16/06/14 (Day 3)	858.50	49.27%
17/06/14 (Day 4)	1227.80	43.02%
18/06/14 (Day 5)	1223.80	-0.33%
19/06/14 (Day 6)	1269.80	3.76%

Table 6.4: Area of horse chestnut leaf mine #5 on 6 consecutive days

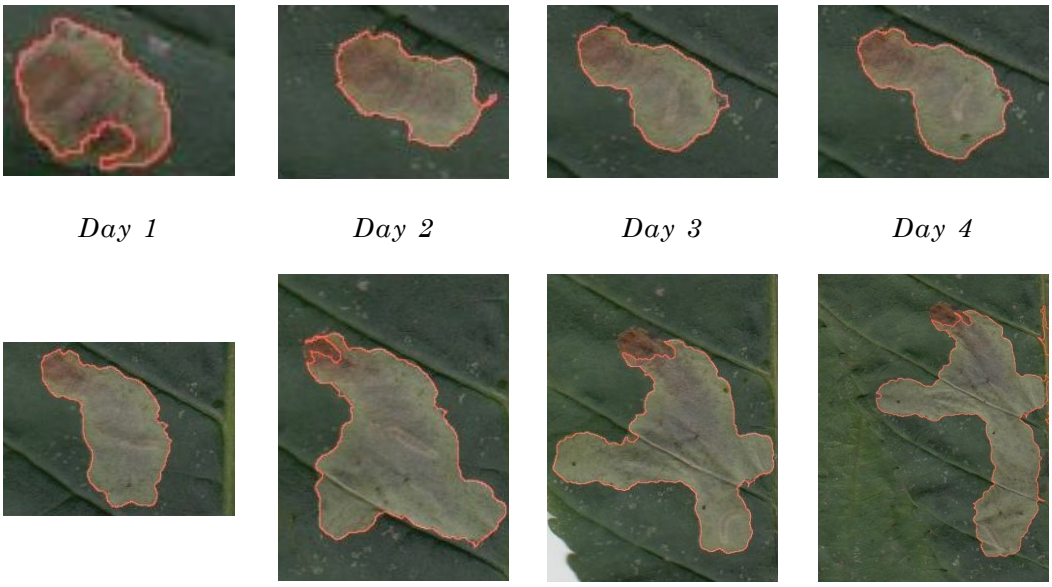
Day 1 *Day 2* *Day 3*

Day 5 *Day 6*

Figure 6.9: Leaf mine example #5

Date	Area (pixels)	Growth rate (%)
14/06/14 (Day 1)	428.88	n/a
15/06/14 (Day 2)	693.75	61.76%
16/06/14 (Day 3)	1041.90	50.18%
17/06/14 (Day 4)	1018.50	-2.24%
18/06/14 (Day 5)	1045.80	2.67%

Table 6.5: Area of horse chestnut leaf mine #6 on 5 consecutive days



Day 1 *Day 2* *Day 3* *Day 4*

Day 5 *Day 6* *Day 7* *Day 8*

Figure 6.10: Leaf mine example #6

Date	Area (pixels)	Growth rate (%)
06/06/14 (Day 1)	243.88	n/a
07/06/14 (Day 2)	316.88	40.79%
08/06/14 (Day 3)	404.88	32.13%
09/06/14 (Day 4)	428.50	7.51%
10/06/14 (Day 5)	586.13	99.88%
11/06/14 (Day 6)	921.50	57.72%
12/06/14 (Day 7)	1257.50	66.19%
13/06/14 (Day 8)	2066.90	0.07%

Table 6.6: Area of horse chestnut leaf mine #4 on 8 consecutive days

Day 1 *Day 2* *Day 3*

Day 4 *Day 5* *Day 6*

Figure 6.11: Leaf mine example #7

Day 1 *Day 2* *Day 3*

Figure 6.12: Leaf mine example #8

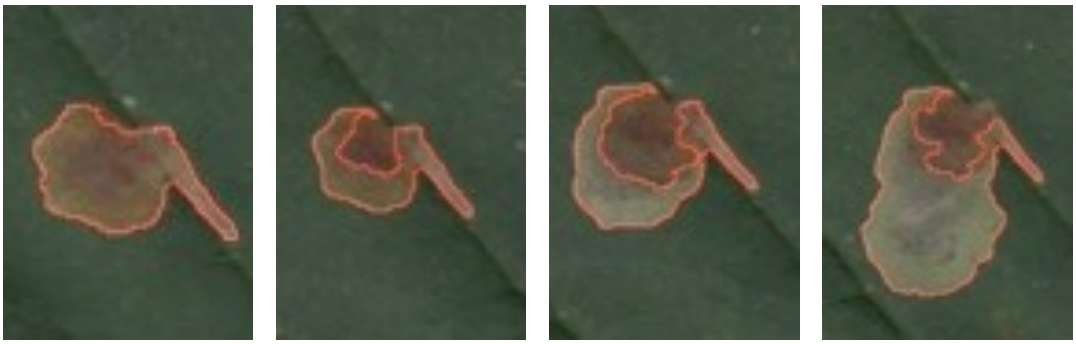
Day 1 *Day 2* *Day 3*

Day 4 *Day 5* *Day 6*

Figure 6.13: Leaf mine example #9

Day 1 *Day 2* *Day 3* *Day 4*

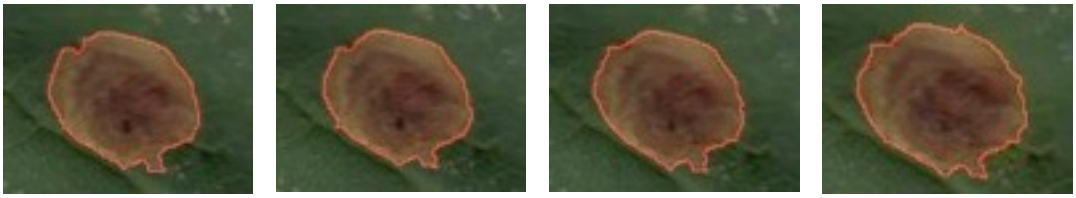
Figure 6.14: Leaf mine example #10



Day 1 *Day 2* *Day 3* *Day 4*

Day 5 *Day 6* *Day 7* *Day 8*

Figure 6.15: Leaf mine example #11



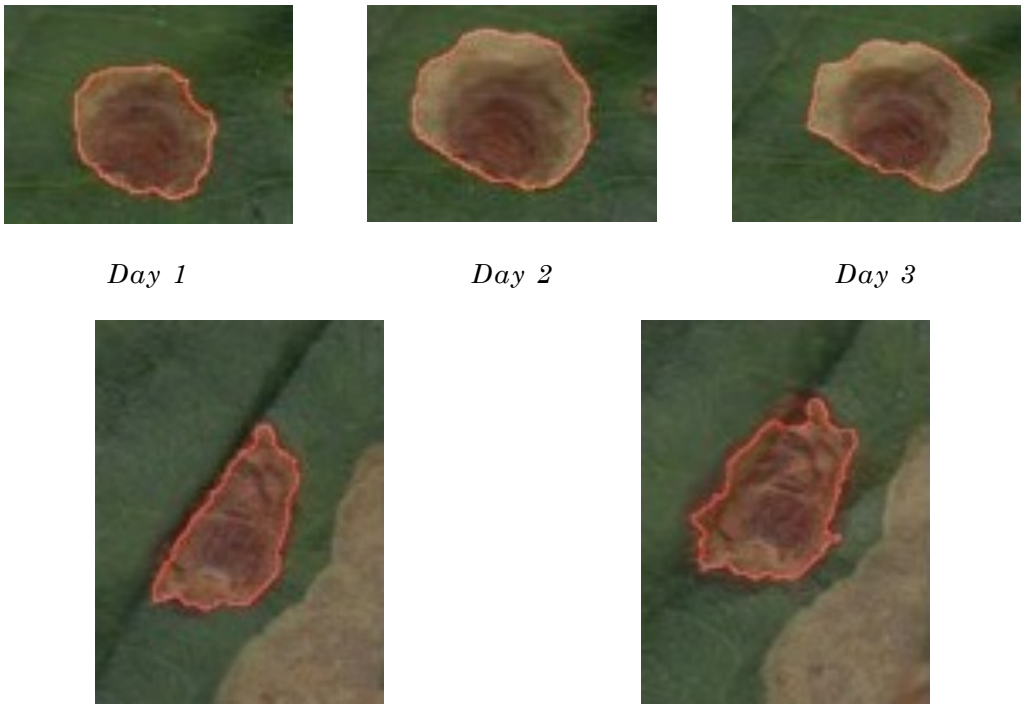
Day 1 *Day 2* *Day 3* *Day 4*

Figure 6.16: Leaf mine example #12

Day 1 *Day 2* *Day 3*

Day 4 *Day 5* *Day 6*

Figure 6.17: Leaf mine example #13



Day 1 *Day 2* *Day 3*

Day 4 *Day 5*

Figure 6.18: Leaf mine example #14

Overall, the proposed approach detects leaf mines of *C.* ohridella successfully from digital images. Further examples are included in appendix B. The accuracy of segmentation depends on the distribution of intensity levels in the images, which in turn relies on the image acquisition methods.

6.3 Limitations

This project proposed developed a method of detecting leaf mines using digital images of leaves from affected host plants. The results show that it is possible to locate leaf mines and fungal spots on scanned images and digital photographs, using a variety of digital image processing techniques. Similarly, it is a viable system to measure a few necessary morphological characteristics of leaf mines and tar spots.

There are several factors that affect the process of the identification of leaf mines. For instance: (1) larvae tend to be inconspicuous during the early stages as they develop within leaf tissues; (2) the complicated task to identify the developmental stage of leaf mines at a particular time, or when the larvae evacuate the mines; (3) few easily recognisable morphological features in the mines; and (4) the lack of association between adult leaf-mining insects and their host plants until oviposition [30].

6.3.1 Challenges in data collection

As the samples were collected from a natural environment, the precise developmental stage of an existing leaf mine at the time of collection is unknown. The exact stage of larvae growth could not be judged by its size or shape alone, nor when the larvae had vacated the mines. These factors are largely influenced by factors such as environmental conditions and behaviour of the adult moths.

For the *C. ohridella* samples, storage conditions between collection and image acquisition also influence accuracy. The leaves were kept inside plastic boxes indoors overnight, at room temperate, hence the development of larvae could potentially be affected, as well as the conditions of the leaves themselves. It was noted that the condition of the samples deteriorates over time with this storage

method, with the leaves drying out, curling up or becoming too brittle to handle after approximately a week. Moreover, the daily handling of the samples also contributes to some damage of the leaves. In general, the usable timeframe for most leaves collection with the current method is around 6 to 10 days. Whether the samples should be protected from light, kept at a different temperature, or preserved in other ways, can be further investigated in the future. These considerations add to the challenge in their identification.

As mentioned in section 2.2.1, the length of the larval stage for *C. ohridella* leaf miners lasts around 20 to 45 days, depending on the season and climate [76] [77] [78]. As the current collection and storage methods limit the use of a particular sample to 6 to 10 days, the data obtained cannot accurately represent the full larval stage of *C. ohridella* leaf mines. Additionally, it is impractical to determine the exact moment when the eggs were laid with the ovipositors. For experimental purposes, the date of sample collection was considered "day 1" throughout the experimental stage. Nevertheless, it was not possible to determine whether "day 1" could be during the final instar, the day before the larva emerges from the mines, or even after the larva has vacated the mine.

Moreover, the data collection process is generally restricted to the spring and summer months due to phenology. Different generations of leaf mines may behave differently; for instance, the growth of later generations *C. ohridella* larva may be restricted by existing leaf mines, which could potentially lead to different feeding patterns. Girardoz *et al.* argued that the rate of infestation of *C. ohridella* does not increase with successive generations in the same summer [73].

6.3.2 Image acquisition

Some of the differences between using scanners and digital photography have been briefly discussed in section 5.1, such as lighting, image quality and file sizes. For digital scanners, it is relatively easy to control the background and keep the illumination source constant; however, there are several issues with this method. The maximum size of the scanned object depends on the glass window of the particular scanner. When focusing on a limited area within the leaf (i.e. a leaf mine or tar spot), this does not generally cause any issue as its size depends on

the dots per inch (dpi) setting, the number of pixels, and not the size of the whole leaf. Nonetheless, if the aim is to estimate the amount of damage on a leaf, the entire leaf must fit within the scanners' range.

As for image acquisition using digital photography, the image quality is determined by the equipment, source of lighting, time of day, background, the distance between the camera and objects, et cetera. For the identification of leaf mines and fungi, samples can either be photographed in their natural environment or collected from the plants for photographing in a more controlled environment. The latter conventionally involves placing a sample against a plain background, and a camera is set up at a fixed distance, perpendicular to the object in question, with a constant and controllable source of illumination. Ideally, the distance between the object and camera lens is uniform, as is the focus and lighting. An alternative is to produce samples under a laboratory environment. Still, it is problematic logistically to prepare different species of leaf-mining insects in order to obtain an adequate sample size, alongside the host plants for each species. It would be feasible to monitor and control the storage condition in which the plants and insects. Nonetheless, the developmental habit of the larvae may behave differently in a laboratory setting compared to their natural environment. It would also be challenging to portray and mimic the unpredictability of nature.

The file size of images is another issue. As mentioned in section 5.1.3, image resolution generally correlates proportionally with the storage space and transmission bandwidth required. Scanned images can have a resolution from 72 dpi, up to a range of thousands. Photos that are taken with smartphone cameras are likely to be around 8 to 12 megapixels, while digital single-lens reflex cameras create pictures can be up to 60 megapixels. The higher number in dpi or megapixels, the larger the file size and the longer it would take to be processed. It is essential to find a compromise between image quality and processing time.

6.3.3 Errors in segmentation

When the pixels of a binary image are divided into "object" area (image foreground pixels) and background pixels, it is inevitable there would be errors. For example, some stems or more prominent veins of the leaf can sometimes

"blended into" lesions areas if the variance of colour is not significant. Small objects are required to be eliminated as a form of noise reduction to improve accuracy, as are objects on the border.

Some of the limitations of the current image segmentation methods were briefly discussed in sections 5.2.2 and 5.2.3. Most common edge detectors are independent of directions, and are highly susceptible to any change in the intensity of neighbouring objects as a result. Additional preprocessing techniques are necessary to filter noise. By incorporating pattern recognition techniques such as clustering and genetic algorithms, it is possible to make improvements in the current image segmentation method, as suggested in sections 4.2.3 (clustering) and 4.4.3 (GAs).

An alternative method for image segmentation is convolutional neural networks, which were suggested in chapter 4. The use of CNNs was dismissed originally as it is not essential for the detection of leaf mines or tar spots from cropped images. However, further refinement to the segmentation techniques discussed in sections 5.2.2 to 5.2.4 is required to improve performance, as the effectiveness of thresholding deteriorates with the increase of number of classes. Compared to traditional image segmentation techniques, CNNs are suited to segment images into multiple classes. For instance, a leaf image can be sorted into "object", "leaf", and "background" pixels using CNNs. This could possibly counter the issues caused by photographing samples at the natural habitat of leaf-mining insects and their host plants.

6.4 Growth rate analysis

As reviewed in chapters 1 and 2, management strategies for leaf-mining insects require an efficient way of monitoring their growth. The success of traditional chemical management is ambiguous due to the leaf mines being protected by the epidermis of the leaf. This section proposes a way to monitor and predict the growth of leaf mines based on the daily increase in size in digital images. As mentioned in the previous subsection, the current data collection method limits the usable period of each sample to a shorter portion of the larval stage of *C. ohridella*. It is also logistically difficult to collect enough samples from a large

variety of horse chestnut trees. Potentially, a substantial database can be created by placing horse chestnut plants in a greenhouse or a laboratory environment for constant monitoring. It is not known, however, whether their behaviour would differ in captivity. Regardless, the possibility of monitoring and predict the growth of leaf-mining insects will be explored in this section, using *C. ohridella* as an example.

Some of the computer-aided taxonomy systems discussed in section 2.1 have used machine learning for solving related problems, with clustering and neural networks being some of the more popular classifiers of choice. Artificial neural networks, due to their adaptability, have been chosen as a viable method to predict the growth rate of C. ohridella leaf mines. The background of neural networks was discussed in chapter 4. ANNs are capable of learning and identifying correlated patterns, even when the relationship between input data and target values is unknown. The possibility of using a multilayer perceptron (MLP) neural network to optimise the data will be addressed in this section. The system was trained using back-propagation algorithms. MATLAB was once again used as the platform for implementation.

6.4.1 Machine learning in MATLAB

The proposed approach was implemented with MATLAB (version R2018b). The Deep Learning Toolbox (known as the Neural Network Toolbox in previous versions), and the Statistics and Machines Learning Toolbox (the successor of the Statistics Toolbox) was used to develop the proposed system.

The Deep Learning Toolbox provides a framework to suitable for designing and implementing neural networks with various algorithms, models and applications [222]. It is compatible with shallow neural networks, deep neural networks, convolutional neural networks (CNNs) and long-short term memory networks (LTSMs). MATLAB can be used for creating a wide range of neural networks, from simple two-layer feed-forward networks to pattern recognition networks, as well as fitting and training ANNs. The toolbox also includes a graphical user interface (GUI) for curve-fitting, pattern recognition, clustering and time series tools. An example neural network created in MATLAB is displayed in figure 6.19.

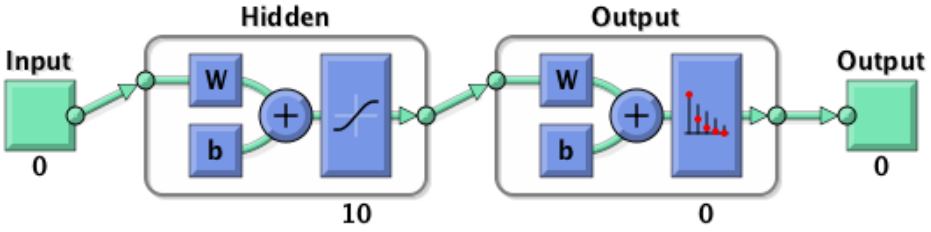


Figure 6.19: Sample neural network in MATLAB [238]

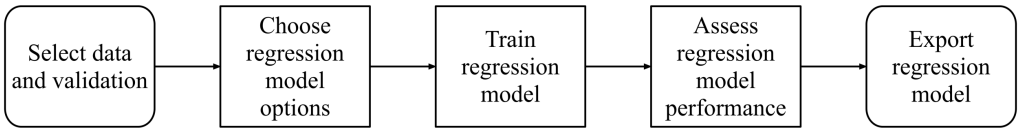


Figure 6.20: Workflow for training regression models [240]

The Statistics Toolbox (precursor to the Statistics and Machine Learning Toolbox) of MATLAB provides another set of tools, functions and applications to analyse or model data, using statistics and machine learning [223]. The toolbox incorporates supervised and unsupervised machine learning algorithms, from support vector machines (SVMs), decision trees, *k*-nearest and *k*-means, hierarchical clustering, Gaussian mixture models, and hidden Markov models.

The Regression Learner Application, which was first introduced with MATLAB R2017b), is a tool from the Statistics Toolbox for training and validate regression models. It supports various regression models and algorithms, including linear regression models, regression trees, Gaussian process regression models, support vector machines, and an assortment of regression trees [239]. It is suitable for data exploration, feature selection, validation, and result evaluation. The flow chart in figure 6.20 describes the working for training regression models using the Regression Learner Application [240].

6.4.2 Curve fitting

The performance of any machine learning algorithm was based on the relevance of input and output variables used in the training process. The purpose for the

proposed ANN in this instance being the monitoring for leaf mines in terms of area, the input and output variables were chosen based on these practical considerations: (1) inputs should be relevant to the development of *C. ohridella* larvae; (2) the sensitivity of the output variables to the input should have the same order of magnitude for all input variables, and should be as high as possible to improve training effectiveness [241].

After the samples were digitalised and segmented using the processes described in chapter 5, 15 leaf mines from 7 leaves were selected randomly. The area extracted from the images of leaves was regarded as the inputs of the growth rate predictor for training and predicting purposes. The area of the leaf mines ranges from approximately 200 to over 1000 pixels. Due to the nature of some of the input variables and the sensitivity to input data distribution, transformations were made to some of the data. For example, instead of the dates of image acquisition, these were substituted with numbers, with 1 being the day on which the sample was collected.

Since the area of a leaf mine corresponds to the amount of damage caused by a larva, one can conclude that it can only increase (indicating further consumption of leaf tissue by the larva), or remain constant (for the reasons addressed in section 6.1). Based on this assumption, the area of a leaf mine cannot shrink overnight, therefore on days where the area of a leaf mine seemed to have be shrunk overnight, this should be attributed to an error in the segmentation process, and treated as an erroneous data point. For instance, based on the example shown table 6.4, the area for day 5 can be considered a miscalculation.

It is necessary to adjust these erroneous data points (values outside a healthy growth trend), and to substitute missing data. This can be approached in the following ways, assuming the damaged caused by the leaf mine cannot be reversed overnight: (1) substitute with the same data as the previous day; (2) use the same data as the next day; (3) calculate the mean value of the data from the days before and after; and (4) estimate the data based on growth behaviour. Methods (1) to (3) can easily be implemented, but (4) is potentially more accurate if it is possible to simulate the growth rate of *C. ohridella* leaf mines. Regression models can be used estimate the erroneous or missing data.

Figures 6.21 and 6.22 shows two example using generalised linear regression to help model the growth in area of leaf mines. The data sets used in figures 6.21 and 6.22 were shown in tables 6.1 (leaf mine #1) and 6.2 (leaf mine #2) respectively. In both instances, the higher-polynomial curve fits are more satisfactory than using a linear fit.

Figure 6.23 shows the area plots of the 14 leaf mines in section 6.2. As discussed in subsection 6.3.1, the current data collection method limit the samples' usable timeframe to around 6 to 10 days, which cannot accurately represent the full larval stage of *C. ohridella* leaf mines. As a result, the current data is inadequate to determine whether it is possible to predict the damage caused by *C. ohridella* leaf mines. The life cycle of *C. ohridella* was described in further details in section 2.2.1. However, the method discussed in this section can be used an as example for fitting curves to the ongoing damage caused by *C. ohridella* leaf mines based on area.

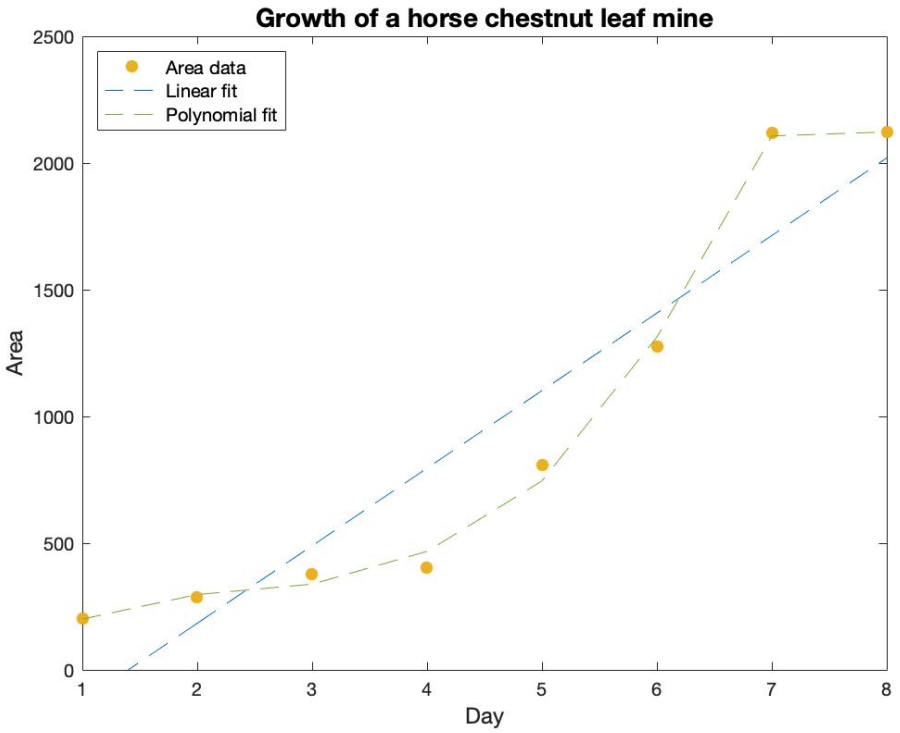


Figure 6.21: Fitting data with generalised linear regression (leaf mine #1)

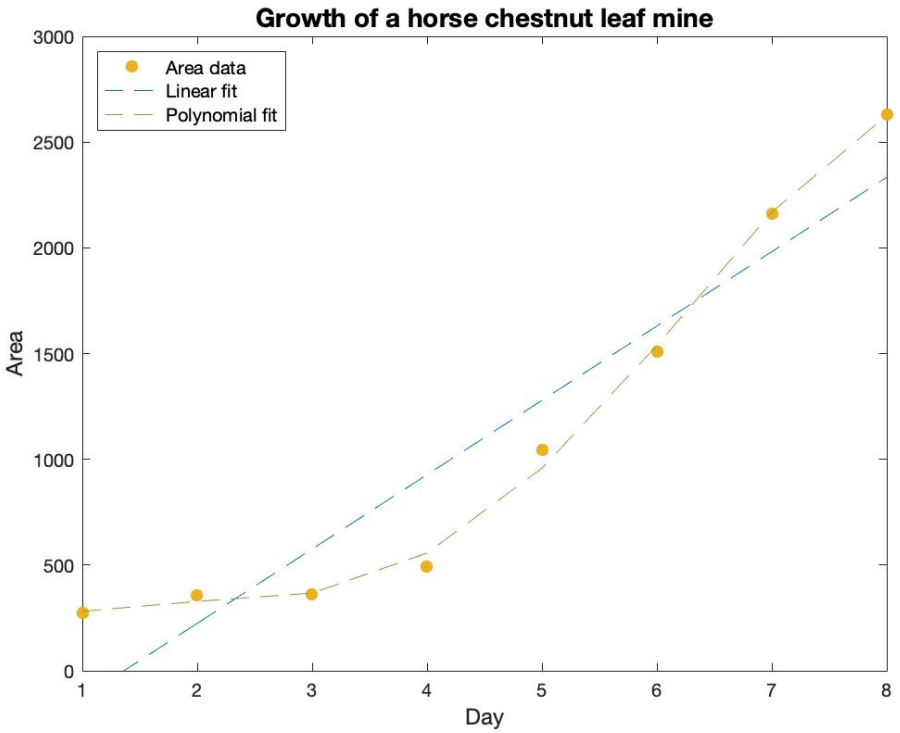


Figure 6.22: Fitting data with generalised linear regression (leaf mine #2)

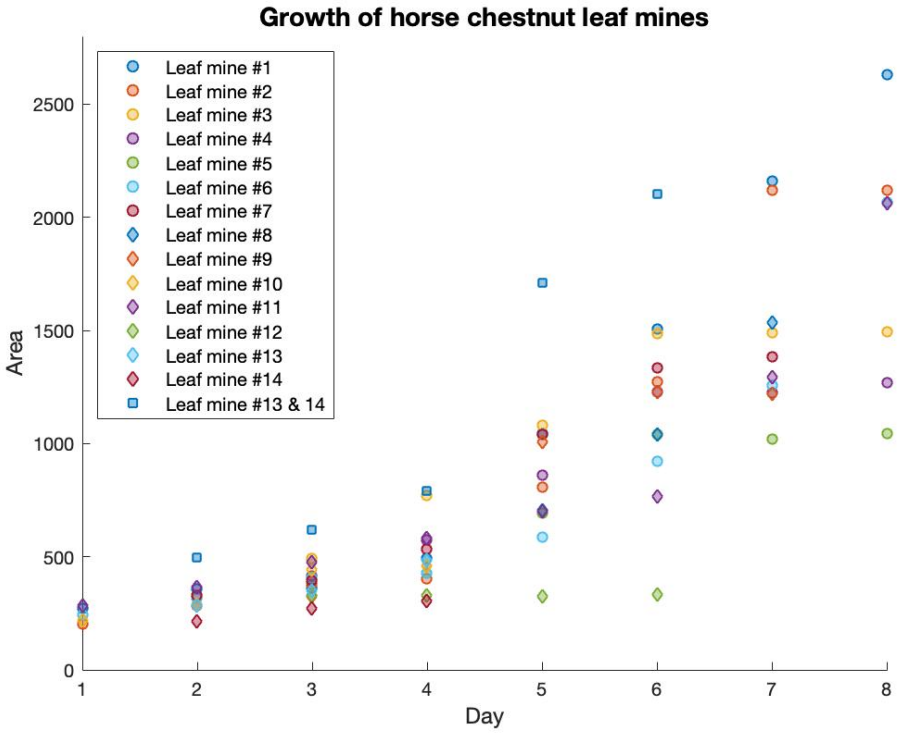


Figure 6.23: The growth of 14 horse chestnut leaf mines

6.4.3 Performance analysis

The neural network in figure 6.24 was chosen as the MLP for mapping between a set of 116 area data to a set of targets (days 1 to 8). The two-layer feed-forward neural network has 10 hidden neurons and a single hidden layer. 70% of samples (82 samples) were presented to the NN for training, 15% (17 samples) were used for measuring network generalisation, and the remaining 15% (17 samples) were used for testing. The testing data has no effect on training; their purpose is to independently measure the performance of the network during and after training. Figure 6.25 shows the regression result. The error histogram in figure 6.26 shows the differences between target values and predicted values after training the perceptron [242]. The regression plots and error histogram after retraining are shown in figures 6.27 and 6.28. Additional regression plots and error histograms after further retraining are included in appendix C.

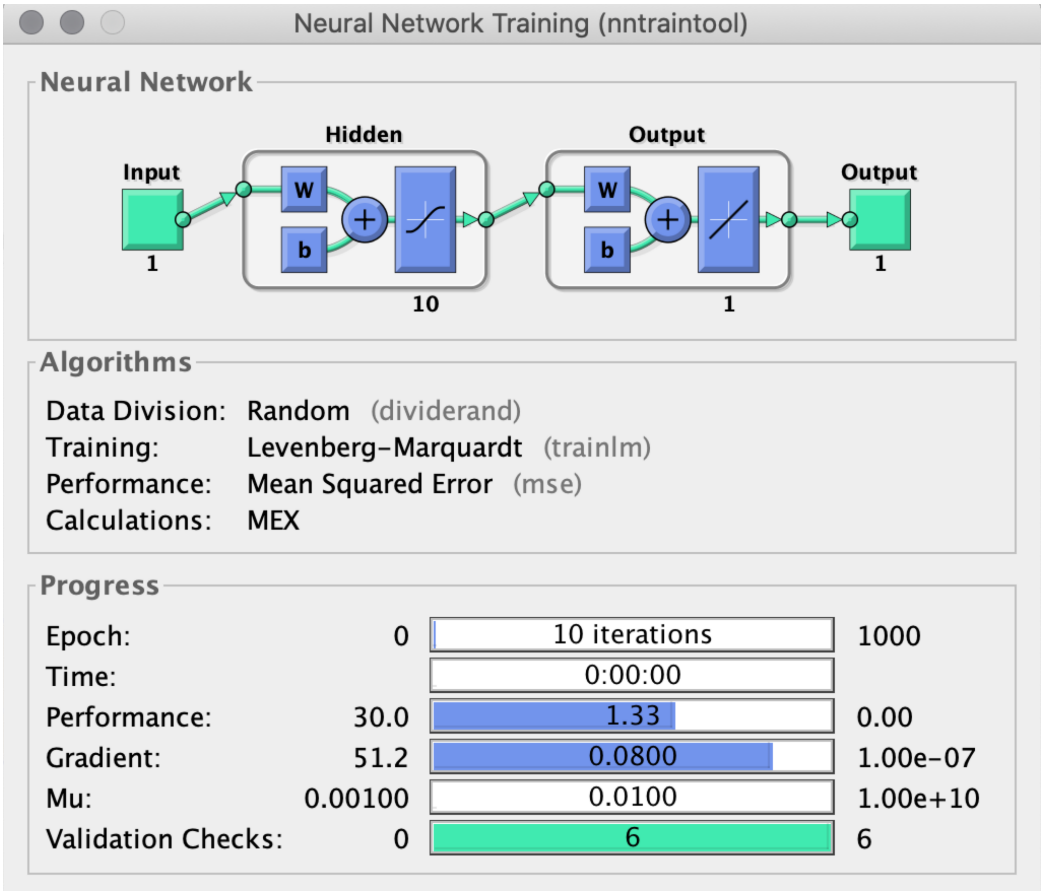


Figure 6.24: Two-layer feed-forward neural network

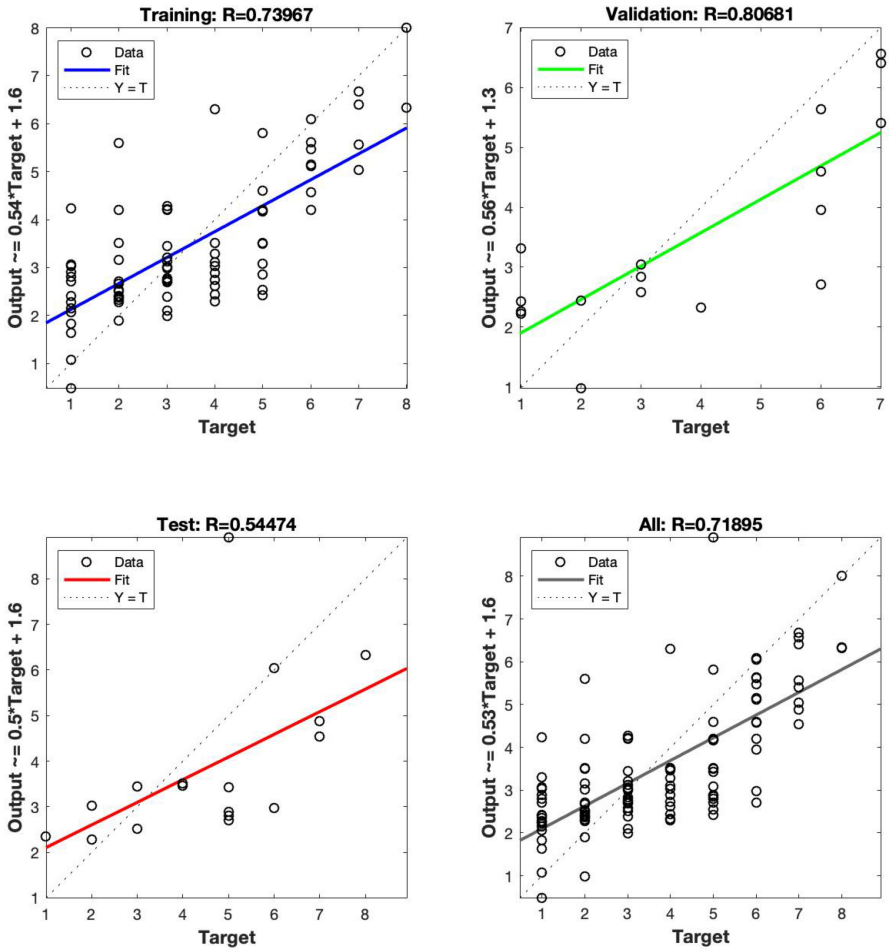


Figure 6.25: Regression plots

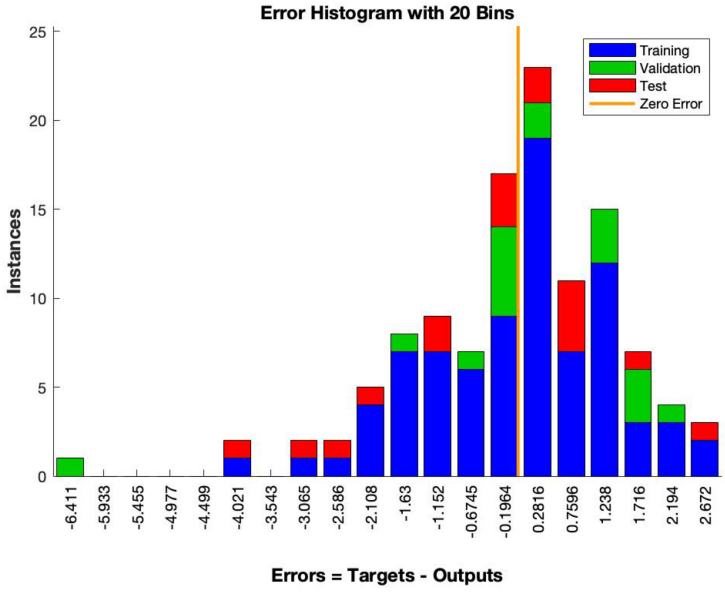


Figure 6.26: Error histogram

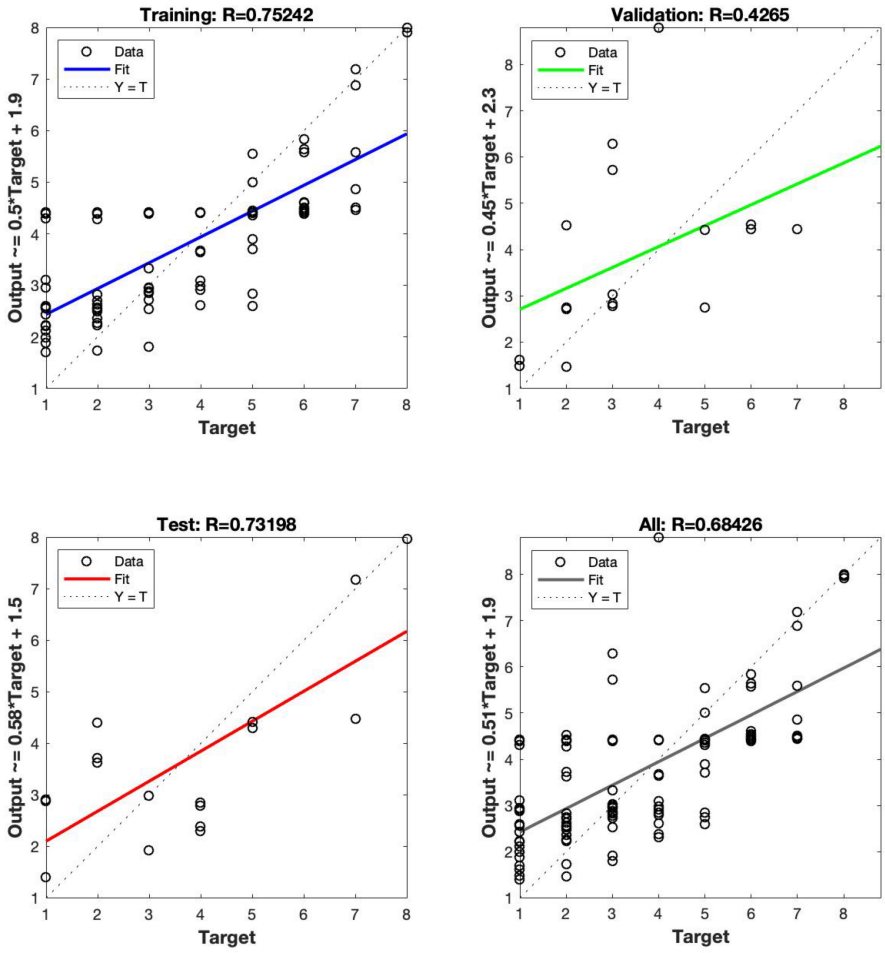


Figure 6.27: Regression plots after retraining

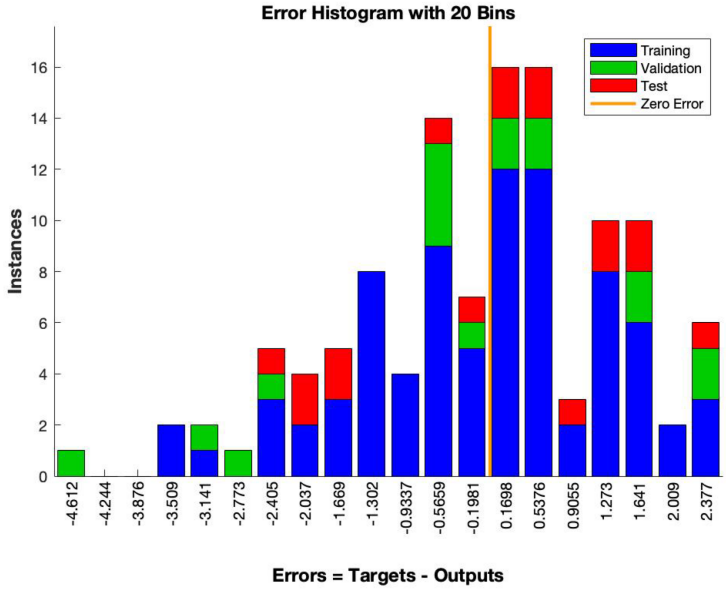


Figure 6.28: Error histogram after retraining

Unsurprisingly, based on the considerations highlighted earlier in this chapter, it is not possible to effectively predict the growth rate of *C. ohridella* larvae using a multilayer perceptron neural network. However, this experiment illustrates the overall features and requirements for an ANN aimed at predicting the increase in damage caused by *C. ohridella* leaf mines. For example, if a predictor was successfully capable of forecasting the growth of *C. ohridella* leaf mines, its prediction performance can be evaluated by these error functions: the coefficient of correlation, the mean square error, and the average absolute error [243]. The prediction result using ANNs can also be compared to those obtained with a standard statistical regression method such as generalised linear regression, as shown in section 6.4.2, or multiple linear regression (MLR).

6.5 Summary

This chapter shows that leaf mines of *C. ohridella* and tar spots of *R.* acerinum can be successfully detected from scanner samples of horse chestnut and maples leaves respectively. The detection system can serve as a semi-automated tool for extracting morphological features commonly used in research projects, such as area and perimeter. Although the main unit of measurement is in pixels, its relationship between another unit, for examples, millimetres was established in chapter 5. This could possibly make the measurement process less labour and time consuming, thus facilitating future projects. Chapter 6 also addressed some of the limitations of the proposed system for detecting leaf mines and fungal spots. Possible improvements to each stage of the process were discussed in this chapter, as were potential solutions to counter some of the limitations.

Additionally, this chapter investigated the use of multilayer perceptron neural networks for monitoring and predicting the growth in damages caused by *C. ohridella* leaf mines. By measuring the area of the leaf mines every day, it is possible to calculate the amount of growth overnight. While the experiment described in this chapter, due to the limitations of samples, did not allow the development of a neural network that is able to successfully carry this task, the methodology used in its development can serve as a guide to the development of feature-based intelligent algorithms for a variety of tasks related to the monitoring and classification of leaf-mining insects.

Chapter 7

Conclusion

Computer-aided taxonomy, in particular automated classification systems, are in high demand due to the discovery of new species. Automated species identification has become more established in entomology and taxonomy in the past few decades, as well as in agricultural engineering to monitor crop growth and enhance quality control. This thesis described previous studies on automated species identification and the use of methods to classify them. Although there were a small number of automated classification systems for plant diseases, research into leaf-mining insects is limited. Leaf miners are highly invasive species which attack a wide range of crops and ornamental plants. The larvae of leaf-mining insects feed on their host plants, and gradually forming blotch or gallery mines on the surface of leaves. The amount of damage caused by leaf miners vary, depending on their species, as well as the environmental conditions and stages of development. The increase in human activities in recent years has led to the wide-spread of infestation, meaning an effective way to detect and monitor their behaviour is becoming necessary.

This project began with developing a semi-automated system to identify the leaf mines of *Cameraria ohridella* (horse chestnut leaf miner) using digital image processing techniques. The detection approach was also tested on two additional species of leaf-mining insects, *Lyonetia clerkella* (apple leaf miners) and *Phytomyza ilicis* (holly leaf miner), which presented specific challenges due to their shape and other characteristics. The plant pathogen *Rhytisma acerinum* was also introduced as a variation.

As part of the investigation into detecting leaf mines, different methods were trialled and tested for: (1) locating a leaf mine on a digital image, (2) finding the edge of the mine, (3) calculating the area of the leaf mine in pixels, and (4) extracting global and local features. Different data collection methods were compared, and a system was designed with MATLAB to locate leaf mines and tar spots in photographs and scanned images. Digital image processing techniques such as the conversion between colour models, image segmentation, and morphological operations were incorporated. The segmentation results and features extracted can then be used as parameters to possibly classifying leaf mines based the growth of infestation.

The increase in area of a leaf mine can be presumed to be the ongoing damaged caused by a larva. Therefore, the image segmentation results serve as an alternative way to measure the size of a leaf mines in terms of area and perimeter. By repeating the leaf mine detection process over several consecutive days, it is possible to observe the growth of the infestation and infer its growth rate, as well as provide information that aids the behavioural study of leaf-mining insects. This thesis then explored the use of linear regression models and multilayer feed-forward neural networks to measure the growth of *C. ohridella* leaf mines. The area and growth data were then used to train a multilayer perceptron neural network, to investigate the possibility of classifying leaf mines based on the growth of infestation. Their growth rate over several days can potentially be used as a variable for the classification of different species of leaf miners.

This thesis addressed some of the challenges and limitations encountered during the process and highlighted the merits and disadvantages of different image acquisition methods. The tools for detecting leaf mines from digital images and for monitoring their growth aid in the automated classification of leaf mines, and act as the fundamental components of image-based automated classification system for leaf mines. The image processing section also serves as a semi-automated tool for extracting morphological features commonly used in research projects, such as area and perimeter. Moreover, the proposed approach uses only digital images, unlike the traditional method, which relies heavily on the expertise of taxonomists, or DNA barcoding, which requires the complex process of scraping evacuated leaf mines to obtain the DNA. Because of this, DNA

barcoding is a way more complex method that is suitable for collecting live data in a natural environment, unlike photographing samples in a natural environment without removing the leaves from the plants.

7.1 Future work

Further data collection is required to create a broader database, and more species of leaf-mining insects or plant diseases can be incorporated in the future. The proposed leaf mine detecting system can potentially be adapted to suit a wider variety of leaf-mining insects. This would benefit a wider audience, as well as prevent overfitting of the proposed method. Alternative storage solution of samples is to be considered, alongside refinement of current image acquisition method. The improvement would include professional setup for digital photography, completing with a technique for colour calibration.

Additionally, if samples of *C. ohridella* leaf mines on secondary host plants were to become available, the current feature extraction methods can be extended to cover a broader range of features, to distinguish among the leaf mines on different host plants. Similarly, a classifier for *C. ohridella* leaf mines and the horse chestnut leaf blotch could be helpful for botanists or taxonomists, provided that samples of the fungal spot are accessible.

Alternative methods for image segmentation can be explored, such as the use of clustering algorithms and convolutional neural network. Improvement in data collection and image acquisition methods are also required. In the future, the system could be extended as a smartphone application, where users can analyse images from their devices and upload pictures onto an online database.

Appendix A

MATLAB code

Reading images

```
% clear workspace and close existing figures
clear;
close ALL;

% read image
imOriginal = imread('filename.jpg');

% obtain the dimension of input image
[height, width, dim] = size(imOriginal);

% define variables
nrows = height;
ncols = width;
```

Colour transformation (pp. 47 – 54, 107 – 109)

```
% RGB component images
imRed = imOriginal(:,:,1);
imGreen = imOriginal(:,:,2);
imBlue = imOriginal(:,:,3);

% RGB to HSI transformation
imHSV = rgb2hsv(imOriginal);

% HSI component images
hue = imHSV(:,:,1);
saturation = imHSV(:,:,2);
intensity = imHSV(:,:,3);
```

HSI binary masks (pp. 110 – 112)

```
% assign low and high thresholds for each component
% restrict hue value to between 80° to 160° for green colours
hueThresholdLow = (80/360);
hueThresholdHigh = (160/360);

saturationThresholdLow = graythresh(saturation);
saturationThresholdHigh = 1.0;

intensityThresholdLow = graythresh(intensity);
intensityThresholdHigh = 1.0;

hueMask = (hue >= hueThresholdLow) & (hue <=
hueThresholdHigh);

saturationMask = (saturation >= saturationThresholdLow) &
(saturation <= saturationThresholdHigh);

intensityMask = (intensity >= intensityThresholdLow) &
(intensity <= intensityThresholdHigh);
objectMask = uint8(hueMask & saturationMask & intensityMask);
```

Intensity thresholding (pp. 112 – 125)

```
% for N threshold values
level = multithresh(intensity,N);
valuesMax = [level max(intensity(:))];
[quant_max, index] = imquantize(intensity,level,valuesMax);
valuesMin = [min(intensity(:)) level];
quant_min = valuesMin(index);

% segment image into N+1 levels
seg_I = imquantize(intensity,level);

for c = 1:ncols
    for r = 1:nrows
        if ((intensity(r,c) > 0) && (intensity(r,c) <=
level(:,2)))
            B(r,c) = 1;
        elseif (intensity(r,c) > level(:,2))
            B(r,c) = 0;
        end
    end
end
```

Edge detection & morphological image processing (pp. 125 – 128)

```
% remove connected components/objects with >1000 pixels
imBinary = bwareaopen(B,1000);

% edge detection using the Sobel operator
[~,threshold] = edge(imBinary,'sobel');
fudgeFactor = 0.5;
imBWs = edge(imBinary,'sobel',threshold*fudgeFactor);

% create flat strel [1 1 1]
se90 = strel('line',3,90);
% create flat strel [1;1;1]
se0 = strel('line',3,0);
% dilate using both structuring elements in sequence
imBWsdil = imdilate(imBWs,[se90 se0]);

% user interaction
% define regions to fill by selecting points using the mouse
imBWsdil = imfill(imBWsdil);
 % fill holes
imBWsdil = imfill(imBWsdil,'holes');

% remove any objects connected to the border of the image
imBWnobord = imclearborder(imBWsdil,4);

% create diamond strelseD = strel('diamond',1);
% smooth object by eroding the image twice (with diamond
shaped strel)
imBWfinal = imerode(imBWnobord,seD);
imBWfinal = imerode(imBWfinal,seD);
```

Feature extraction (pp. 129 – 131)

```
% method 1: calculate the perimeter of an object
imBWoutline8 = bwperim(imBWfinal,8);

% method 2: calculate the area of an object
imStat.Area = regionprops(imBWfinal,'area');
% concatenate structure array into a single matrix
imArea = cat(1,imStat.Area.Area);

% calculate centroids of objects in the image
imStat.Centroid = regionprops(imBWfinal,'centroid');
% concatenate structure array into a single matrix
imCentroid = cat(1,imStat.Centroid.Centroid);
```

```
% calculate orientation of objects in the image
imStat.Orientation = regionprops(imBWfinal,'orientation');
% concatenate structure array into a single matrix
imOrientation = cat(1,imStat.Orientation.Orientation);

% find the number of objects
mCircle = size(imCentroid,Y);

for s = 1:mCircle
   imCircle(s).area = imArea(s);
   imCircle(s).centroid = imCentroid(s,:);
   imCircle(s).orientation = imOrientation(s);

   % calculate radius of circle using area information
   imCircle(s).radius = sqrt(imCircle(s).area/pi);
   imCircle(s).xunit = imCircle(s).radius * cos(th) +
imCircle(s).centroid(:,1);
   imCircle(s).yunit = imCircle(s).radius * sin(th) +
imCircle(s).centroid(:,2);
end
```

Curve fitting (pp. 153 – 156)

```
% for days 1 to 8, day 1 = day of sample collection
% day = [1 2 3 4 5 6 7 8]';
% area = [area data obtained in the previous section]';

% linear fit
linearCoef = polyfit(day,area,1);
linearFit = polyval(linearCoef,day);
plot(day,linearFit,'LineStyle','--','Color',[0 0.4470
0.7410])

% polynomial fit
% P = degree of the polynomial
[cubicCoef,stats,ctr] = polyfit(day,area,P);
cubicFit = polyval(cubicCoef,day,[],ctr);
plot(day,cubicFit,'LineStyle','--','Color',[0.4660 0.6740
0.1880])

% logical regression
[logitCoef,dev] = glmfit(day,[area test],'binomial','logit');
logitFit = glmval(logitCoef,day,'logit');
plot(day,logitFit,'LineStyle','--','Color',[0.8500 0.3250
0.0980]);
```

Appendix B

Additional data of *C. ohridella* leaf mines

The growth data of leaf mine examples #7 to #14 in section 6.2 and 8 additional leaf mines are presented in this appendix.

Leaf mine #7

Day	Area (pixels)	Growth rate (%)
Day 1	327.75	n/a
Day 2	392.63	19.79%
Day 3	534.38	36.10%
Day 4	1040.90	94.79%
Day 5	1334.90	28.24%
Day 6	1383.10	3.61%

Table B1: Area of horse chestnut leaf mine #7 on 6 consecutive days

Leaf mine #8

Day	Area (pixels)	Growth rate (%)
Day 1	705.25	n/a
Day 2	1041.50	47.68%
Day 3	1533.80	47.27%

Table B2: Area of horse chestnut leaf mine #8 on 3 consecutive days

Leaf mine #9

Day	Area (pixels)	Growth rate (%)
Day 1	284.75	n/a
Day 2	323.75	13.70%
Day 3	461.75	42.63%
Day 4	1008.10	118.32%
Day 5	1227.80	21.79%
Day 6	1218.60	-0.75%

Table B3: Area of horse chestnut leaf mine #9 on 6 consecutive days

Leaf mine #10

Day	Area (pixels)	Growth rate (%)
Day 1	217.50	n/a
Day 2	289.00	32.87%
Day 3	445.38	54.11%
Day 4	460.13	0.31%

Table B4: Area of horse chestnut leaf mine #10 on 4 consecutive days

Leaf mine #11

Day	Area (pixels)	Growth rate (%)
Day 1	283.00	n/a
Day 2	364.75	28.89%
Day 3	475.50	30.36%
Day 4	572.38	22.48%
Day 5	702.88	20.69%
Day 6	766.00	8.98%
Day 7	1293.80	68.90%
Day 8	2063.50	59.49%

Table B5: Area of horse chestnut leaf mine #11 on 8 consecutive days

Leaf mine #12

Day	Area (pixels)	Growth rate (%)
Day 1	328.13	n/a
Day 2	329.13	0.30%
Day 3	323.88	-1.60%
Day 4	334.88	3.40%

Table B6: Area of horse chestnut leaf mine #10 on 4 consecutive days

Leaf mines #13 and #14

Day	Area (pixels)	Growth rate (%)
Day 1	282.25	n/a
Day 2	349.75	23.91%
Day 3	484.25	38.46%

Table B7: Area of horse chestnut leaf mine #13 on 3 consecutive days

Day	Area (pixels)	Growth rate (%)
Day 1	215.25	n/a
Day 2	270.25	25.55%
Day 3	305.00	12.86%

Table B8: Area of horse chestnut leaf mine #14 on 3 consecutive days

Day	Area (pixels)	Growth rate (%)
Day 4	1709.60	116.61%
Day 5	2100.00	22.92%
Day 6	2880.00	36.91%

Table B9: Total area of leaf mines #13 and #14 for the next 3 days

The data of 8 additional leaf mines (examples #15 to #22) over 5 consecutive days are given in tables B10 through B17. The 8 mines came from two leaves, with leaf mines #15 to #19 on one leaf, and leaf mines #20 to #22 on the other. Figure B1 shows the area plots of leaf mines #15 to #22.

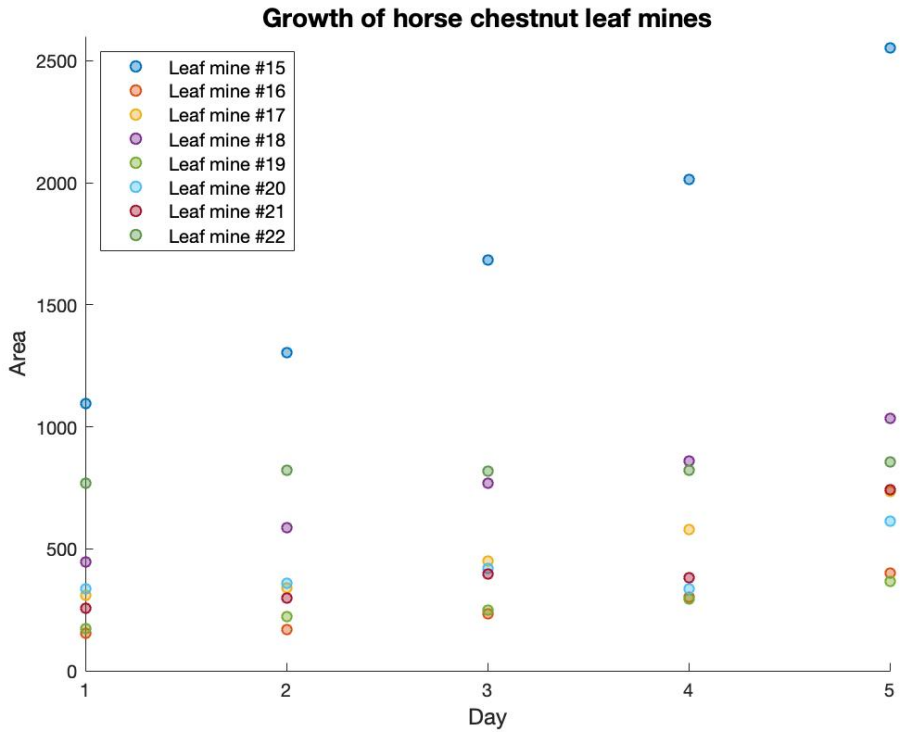


Figure B1: The growth of 8 horse chestnut leaf mines

Leaf mine #15

Day	Area (pixels)	Growth rate (%)
Day 1	1095.10	n/a
Day 2	1305.30	19.19%
Day 3	1682.10	28.87%
Day 4	2014.40	19.76%
Day 5	2551.20	26.65%

Table B10: Area of horse chestnut leaf mine #15 on 5 consecutive days

Leaf mine #16

Day	Area (pixels)	Growth rate (%)
Day 1	153.30	n/a
Day 2	169.75	10.73%
Day 3	232.50	36.97%
Day 4	303.25	30.43%
Day 5	398.63	31.45%

Table B11: Area of horse chestnut leaf mine #16 on 5 consecutive days

Leaf mine #17

Day	Area (pixels)	Growth rate (%)
Day 1	309.25	n/a
Day 2	330.00	9.62%
Day 3	450.38	32.85%
Day 4	577.75	28.28%
Day 5	732.88	26.85%

Table B12: Area of horse chestnut leaf mine #17 on 5 consecutive days

Leaf mine #18

Day	Area (pixels)	Growth rate (%)
Day 1	447.38	n/a
Day 2	587.50	31.32%
Day 3	770.00	31.06%
Day 4	860.88	11.80%
Day 5	1032.90	19.98%

Table B13: Area of horse chestnut leaf mine #18 on 5 consecutive days

Leaf mine #19

Day	Area (pixels)	Growth rate (%)
Day 1	173.38	n/a
Day 2	223.00	28.62%
Day 3	248.25	11.32%
Day 4	293.75	18.33%
Day 5	366.88	24.89%

Table B14: Area of horse chestnut leaf mine #19 on 5 consecutive days

Leaf mine #20

Day	Area (pixels)	Growth rate (%)
Day 1	334.38	n/a
Day 2	358.38	7.18%
Day 3	419.63	17.09%
Day 4	377.63	-19.54%
Day 5	612.00	81.27%

Table B15: Area of horse chestnut leaf mine #20 on 5 consecutive days

Leaf mine #21

Day	Area (pixels)	Growth rate (%)
Day 1	257.25	n/a
Day 2	299.75	16.52%
Day 3	397.13	32.49%
Day 4	383.25	-3.49%
Day 5	741.50	93.48%

Table B16: Area of horse chestnut leaf mine #21 on 5 consecutive days

Leaf mine #22

Day	Area (pixels)	Growth rate (%)
Day 1	767.63	n/a
Day 2	821.13	6.97%
Day 3	818.00	-0.38%
Day 4	822.00	0.49%
Day 5	856.88	4.24%

Table B17: Area of horse chestnut leaf mine #22 on 5 consecutive days

Appendix C

Regression plots and error histograms

The two-layer feed-forward network for the input-output fitting problem is shown in figure C1. Additional regression plots and error histograms after retraining are included in figures C2 to C7. The input is a set of 1 × 116 matrix, representing static data: 116 samples of 1 element. The input data came from 20 leaf mines located on 10 leaves. The target is a set of 1 × 116 matrix, representing the day data (days 1 to 8). As mentioned previously in section 6.3.1, the date of sample collection was considered "day 1", the following day after was considered "day 2", and so on. The MATLAB codes are also included in this appendix.

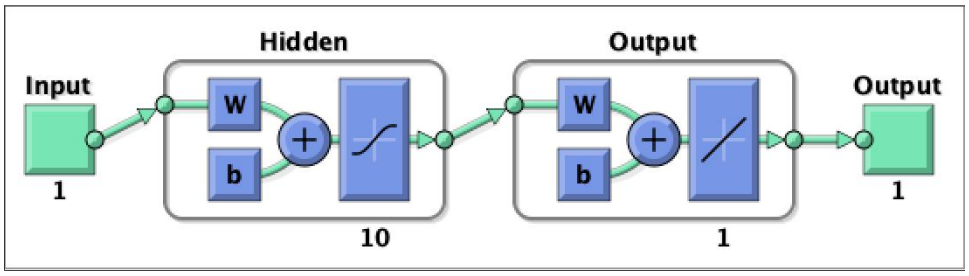


Figure C1: The two-layer feed-forward network

Import spreadsheet data into the MATLAB workspace

```
% clear workspace and close existing figures
clear;
close all;

% import Excel spreadsheet data using readtable
T = readtable('areatable.xlsx','Range','A1:B117');
T1 = T.Area;
T2 = T.Day;

% start neural network GUI
nnstart;
```

Solving an Input-Output Fitting problem (pp, 158 – 161)

```
% input-output fitting problem with a neural network
% script generated by Neural Fitting app

% assuming these variables are defined:
%
%   T1 - input data.
%   T2 - target data.

x = T1';
t = T2';

% choose training function
% Levenberg-Marquardt backpropagation is chosen for training
trainFcn = 'trainlm';

% other training functions:
% 'trainbr' - for challenging problems
% 'trainscg' - for low memory situations


% create a fitting Network
% number of hidden neurons = 10
hiddenLayerSize = 10;
net = fitnet(hiddenLayerSize,trainFcn);

% setup division of data for training, validation, testing
% the samples are divided randomly
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;

% train the network
[net,tr] = train(net,x,t);

% test the network
y = net(x);
e = gsubtract(t,y);
performance = perform(net,t,y)

% view neural network
view(net)

% plots
%
% figure, plotperform(tr)
% figure, plottrainstate(tr)
% figure, ploterrhist(e)
% figure, plotregression(t,y)
% figure, plotfit(net,x,t)
```

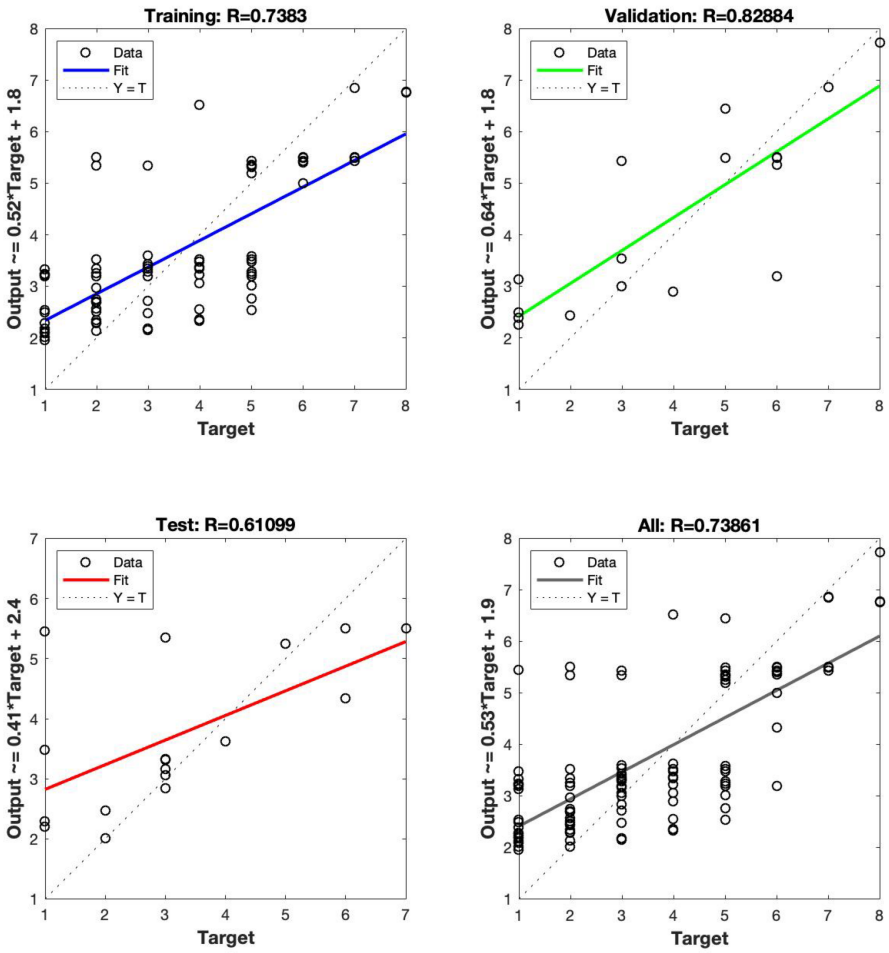


Figure C2: Additional regression plots after retraining

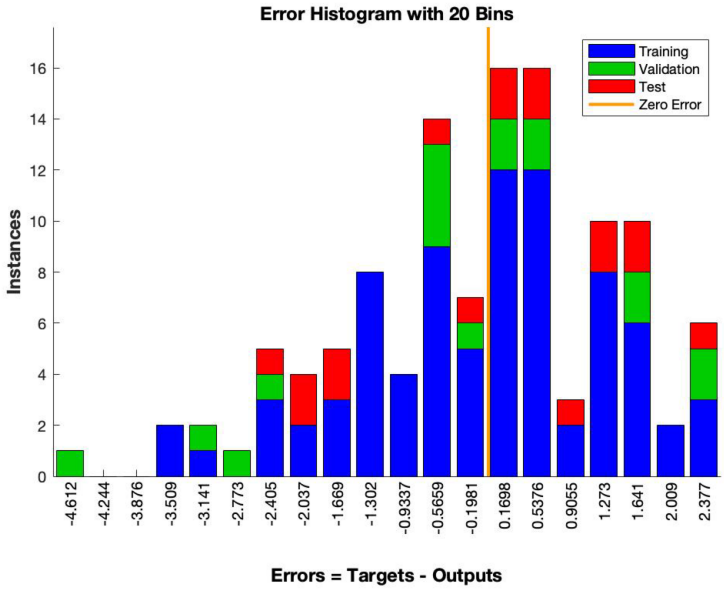


Figure C3: Additional error histogram after retraining

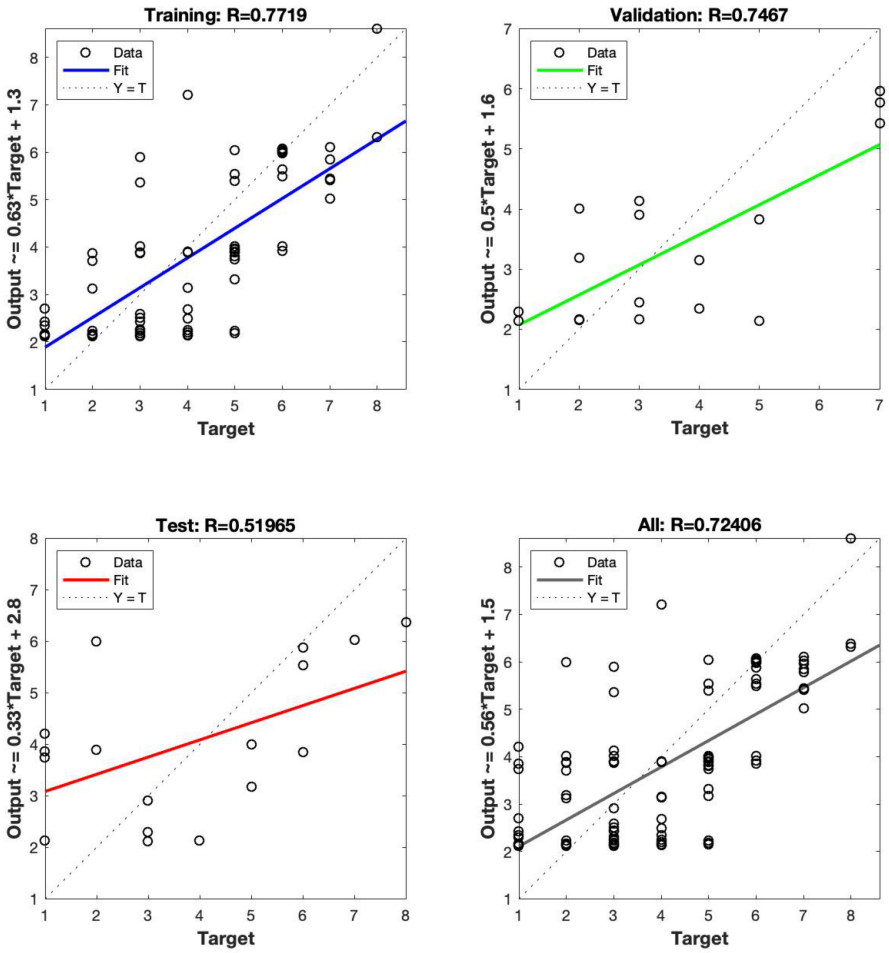


Figure C4: Additional regression plots after retraining

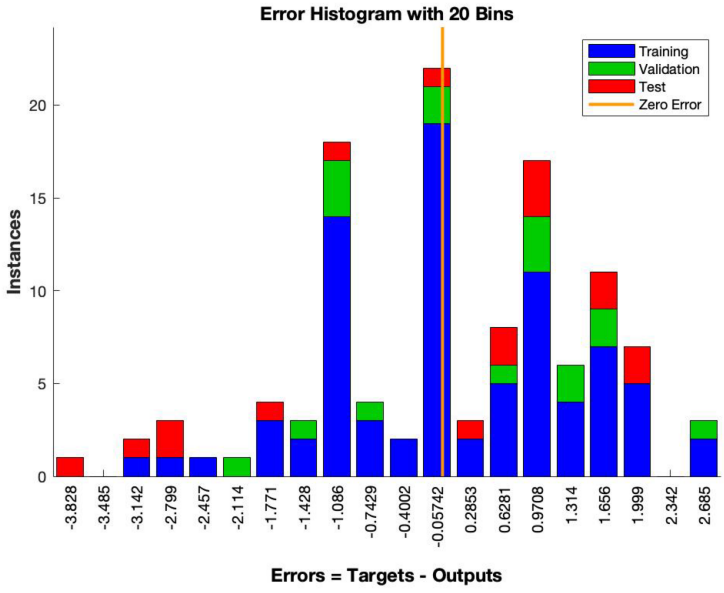


Figure C5: Additional error histogram after retraining

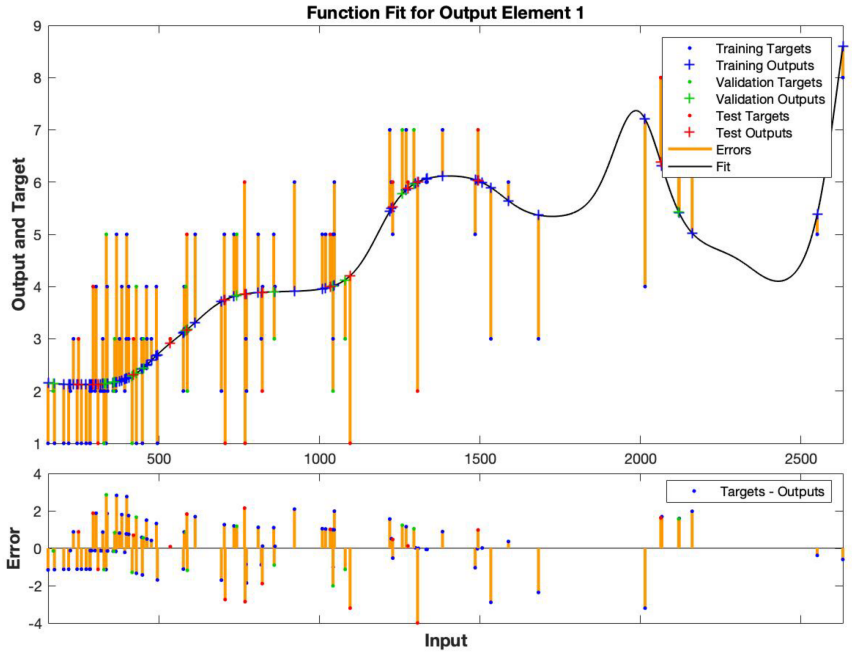


Figure C6: Input-output fitting using a two-layer feed-forward nerual network

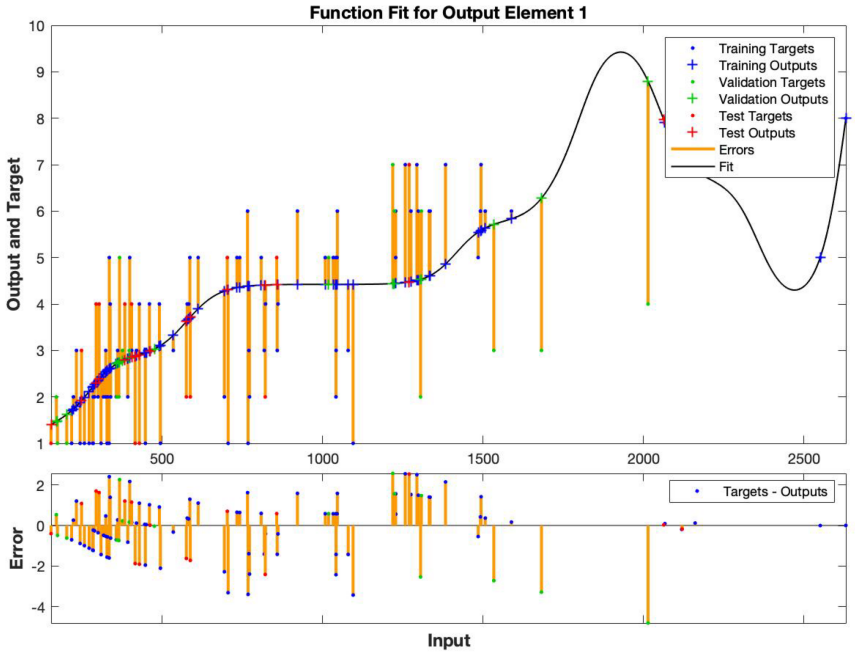


Figure C7: Input-output fitting using a two-layer feed-forward nerual network (after retraining)

Appendix D

Taxa of leaf miners, tar spots and host plants

Cameraria ohridella (Deschka & Dimić, 1986) [70]

Kingdom:	Animalia
Phylum:	Arthopoda
Class:	Insecta
Order:	Lepidoptera
Family:	Gracillariidae
Genus:	*Cameraria*
Species:	*Cameraria ohridella*

Lyonetia clerkella (Linné, 1758) [5]

Kingdom:	Animalia
Phylum:	Arthopoda
Class:	Insecta
Order:	Lepidoptera
Family:	Lyonetiidae
Genus:	*Lyonetia*
Species:	*Lyonetia clerkella*

Phytomyza ilicis (Curtis, 1846) [117]

Kingdom:	Animalia
Phylum:	Arthopoda
Class:	Insecta
Order:	Diptera
Family:	Agromyzidae
Genus:	*Phytomyza*
Species:	*Phytomyza ilicis*

***Rhytisma acerinum* (Schweinitz, 1832) [244]**

Kingdom:	Fungi
Phylum:	Ascomycota
Class:	Leotiomycetes
Subclass	Leotiomycetidae
Order:	Rhytismatales
Family:	Rhytismataceae
Genus:	*Rhytisma*
Species:	*Rhytisma acerinum*

***Aesculus hippocastanum* (Linné, 1758) [5]**

Kingdom:	Plantae
Clade:	Angiosperms
Clade:	Eudicots
Clade:	Rosids
Order:	Sapindales
Family:	Sapindaceae
Genus:	*Aesculus*
Species:	*Aesculus hippocastanum*

***Malus pumila* (Miller, 1768) [245]**

Kingdom:	Plantae
Clade:	Angiosperms
Clade:	Eudicots
Clade:	Rosids
Order:	Rosales
Family:	Rosaceae
Genus:	*Malus*
Species:	*Malus pumila*

***Ilex aquifolium* (Linné, 1758) [5]**

Kingdom:	Plantae
Clade:	Angiosperms
Clade:	Eudicots
Clade:	Asterids
Order:	Aquifoliales

Family:	Aquifoliaceae
Genus:	*Ilex*
Species:	*Ilex aquifolium*

Acer (Linné, 1758) [5]

Kingdom:	Plantae
Clade:	Angiosperms
Clade:	Eudicots
Clade:	Rosids
Order:	Sapindales
Family:	Sapindaceae
Subfamily:	Hippocastanoideae
Genus:	*Acer*

References

1 M. J. Everett *et al.*, "Methods for mapping tissue with optical coherence tomography data," U.S. Patent Application no. 11/717, 263, Sept 20, 2007.

2 M. A. O'Neill *et al.*, "DAISY: An automated invertebrate identification system using holistic vision techniques," in *Proceedings of the Inaugural Meeting of the BioNET-INTERNATIONAL Group for Computer-Aided Taxonomy*, Cardiff, 1997, pp. 13 – 22.

3 R. C. Gonzalez and L. E. Woods, *Digital Image Processing*, 3rd ed. Upper Saddle River, NJ: Pearson Edu., 2008.

4 M. Adanson, Familles naturelles des plantes, Paris, Vincent, 1763.

5 C. Linné, Species Plantarum, Impensis GC Nauk, vol. 1, 1753.

6 P. H. Sneath and R. R. Sokal, Numerical taxonomy. The principles and practice of numerical classification, San Francisco, W. H. Freeman and Company, 1973.

7 L. Boddy *et al.*, "Development of artificial neural networks for identification" in *Information Technology, Plant Pathology and Biodiverisity*, Wallingford, CAB International, 1998, p. 21.

8 M. A. O'Neill, "DAISY: A Practical Computer-Based Tool for Semi-Automated Species Identification," in *Automated Taxon Identification in Systematics, Theory Approaches and Application*, N. MacLeod (Ed.), 1st ed., Boca Raton, FL: CRC Press, 2007, ch. 7, pp. 101 – 114.

9 Tumbling Dice. "Daisy," tumblingdice.co.uk. [Online] Available: http://www.tumblingdice.co.uk/daisy. [Accessed: Apr. 25, 2013].

10 K. J. Gaston and M. A. O'Neill, "Automated species identification: why not?" in *Taxonomy for the twenty-first century*, vol. 359, no. 1444, London: Phil. Trans. R. Soc. Lond. B. 2004, pp. 655 – 667.

11 K. W. Will and D. Rubinoff, “Myth of the molecule: DNA barcodes for species cannot replace morphology for identification and classification” in *Cladistics*, vol. 20, pp. 47 – 55, 2004.

12 P. D. N. Hebert *et al.*, “Barcoding animal life: cytochrome c oxidase subunit 1 divergences among closely related species,” in *Proc. Biol. Sci.*, vol. 270 (Suppl 1), pp. 96 – 99, 2003.

13 R. DeSalle, “Species Discovery versus Species Identification in DNA Barcoding Efforts: Response to Rubinoff,” in *Conservation Biologyi*, vol. 20, no. 5, pp. 1545 – 1547, 2006.

14 Convention on Biological Diversity. “The Darwin Declaration” in Conference of the parties to the Convention on Biological Diversity, fourth meeting, Bratislava, Slovakia, May 4 – 15, 1998.

15 Q. Y. Xiang *et al.*, “Origin and biogeography of *Aesculus* L. (Hippocastanaceae): A molecular phylogenetic perspective,” in *Evolution*, vol. 52, no.4, pp. 988 – 997, 1998.

16 G. Grabenweger and R. Grill, “On the Place of Origin of *Cameraria ohridella* DESCHKA & DIMIC (Lepidoptera: Gracillariidae),” in *Beiträge zur Entomofaunistik*, vol. 1, pp. 9 – 17, 2000.

17 J. F. Freise *et al.*, “Host plant range of the horse-chestnut leaf miner, *Cameraria ohridella* (Lepidoptera: Gracillariidae), a pest of the white flowering horse-chestnut, *Aesculus hippocastanum*,” in *Mitteilungen der Deutschen Gesellschaft für Allgemeine und Angewandte Entomologie,* vol. 14, pp. 351 – 354, 2004.

18 S. B. Dhaygude and N. P. Kumbhar, “Agricultural plant Leaf Disease Detection Using Image Processing,” in *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 2, no. 1, pp. 599 – 602, Jan. 2013.

19 Arivazhagan *et al.*, “Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features,” in *Agricultural Engineering International: CIGR Journal*, vol. 15, no. 1, pp. 211 – 217, Mar. 2013.

20 A. H. Kilkarni and R. K. Patil, “Applying image processing technique to detect plant diseases” in *International Journal of Modern Engineering Research (IJMER)*, vol. 2, issue 5, Sept. – Oct. 2012.

21 K. S. Deshmukh, "Disease detection of crops using hybrid algorithm" in *International Journal of Engineering Research & Technology (IJERT)*, vol. 1, issue 10, Dec. 2012.

22 C. Wen *et al.*, "Local feature-based identification and classification for orchard insects," in *Biosystems Engineering*, vol. 104, Amsterdam, Netherlands: Elsevier B. V. 2009, pp. 299 – 307.

23 P. S. Iandge *et al.*, "Automatic Detection and Classification of Plant Disease through Image Processing," in *Internal Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 7, pp. 798 – 801, July 2013.

24 B. Zhou *et al.*, "Research on Cucumber Downy Mildew Detection System based on SVM Classification Algorithm" in *Proceedings of the 3rd International Conference on Material, Mechanical and Manufacturing Engineering*, Aug. 2015.

25 Z. B. Husin *et al.*, "Feasibility study on plant chilli disease detection using image processing techniques" in *2012 Third International Conference on intelligent systems modelling and simulation*, 2012.

26 H. Wang *et al.*, "Application of neural networks to image recognition of plant diseases," in *2012 International Conference on Systems and Informatics (ICSAI)*, Yantai, China, May 19 – 20, 2012, IEEE, June 2012. Pp. 2159 – 2164.

27 W. Shen *et al.*, "Grading Method of Leaf Spot Disease Based on Image Processing," in *2008 International Conference on Computer Science and Software Engineering*, IEEE, vol. 6, pp. 491 – 494, 2008.

28 Hainruddin *et al.*, "Overview of Image Processing Approach for Nutrient Deficiencies Detection in Elaeis Guineensis," in *2011 International Conference on System Engineering and Technology*, IEEE, pp. 116 – 120, 2011.

29 S. A. P. Derocles *et al.*, "Determining Plant – Leaf Miner – Parasitoid Interactions: A DNA Barcoding Approach," in *PLoS ONE*, vol. 10, no. 2, e0117872, 2015.

30 J. J. Mlynarek *et al.*, "Identification of leaf-mining-insects via DNA recovered from empty mines," in *FACETS*, vol. 1, pp. 217 – 224, 2016.

31 S. B. Patil and S. K. Bodhe, "Leaf disease severity measurement using image processing," in *International Journal of Engineering and Technology*, vol. 3, issue 5, 2011.

32 Z. B. Husin *et al*., "Feasibility Study on Plant Chili Disease Detection Using Image Processing Techniques," in *2012 Third International Conference on Intelligent Systems Modelling and Simulation*, Kota Kinabalu, 2012, pp. 291 – 296.

33 H. Wang *et al*., "Image recognition of plant diseases based on principal component analysis and neural networks," in *2012 8th International Conference on Natural Computation (ICNC)*, Chongqing, China, May 29 – 31, 2012, IEEE, July 2012. Pp. 246 – 251.

34 Phadikar and Sil, "Rice Disease Identification using Pattern Recognition Techniques," in *11th International Conference on Computer and Information Technology*, Khulna, Bangladesh, Dec. 24 – 27, 2008, IEEE, Mar. 2009. pp. 420 – 423.

35 H. Al-Hiary *et al*., "Fast and accurate detection and classification of plant diseases," in *Machine learning*, vol. 14, p. 5, 2011.

36 D. Al Bashish *et al*., "A Framework for Detection and Classification of Plant Leaf and Stem Diseases," in *2010 International Conference on Signal and Image Processing*, Dec. 15 – 17, 2010, Chennai, India, IEEE, Jan. 2011, pp. 113 – 118.

37 M. Gilbert and J. Grégoire, "Visual, semi-quantitative assessments allow accurate estimates of leafminer population densities: An example comparing image processing and visual evaluation of damage by the horse chestnut leafminer *Cameraria ohridella* (Lep., Gracillariidae)," in *Journal of Applied Entomology*, vol. 127, pp. 354 – 359, 2003.

38 J. Schofield and E. D. Chesmore, "Acoustic detection and monitoring of wood boring beetles," *The Journal of the Acoustical Society of America*, vol. 123, no. 5, p. 3778, June 2008.

39 M. Graciarena *et al*., "Bird species recognition combining acoustic and sequence behaviour," in 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 22 – 27, 2011, Prague, Czech Republic. IEEE, July 2011.

40 E. Vilches *et al*., "Data Mining Applied to Acoustic Bird Species Recognition," *18th International Conference on Pattern Recognition (ICPR'06)*, Hong Kong, Aug. 20 – 24, 2006.

41 P. Matyjasiak, “Birds associate species – specific acoustic and visual cues: recognition of heterospecific rivals by male blackcaps,” *Behavioral* Ecology, vol. 16, issue 2, pp. 467 – 471, Mar. 1, 2004.

42 H. Rakotonirina *et al.*, “The role of acoustic signals for species recognition in redfronted lemurs (*Eulemur* rufifrons),” *BMC Evolutionary* Biology, May 2016.

43 D. V. Helversen and O. V. Helversen, “Species recognition and acoustic localization in acridid grasshoppers: a behavioral approach,” in *Neuroethology and behavioral physiology*, pp. 95 – 107, Berlin, Heidelberg: Springer, 1983.

44 M. Mayo and A. T. Watson, “Automated species identification of live moths,” in *Knowledge-Based Systems*, vol. 20, Amsterdam, Netherlands: Elsevier B. V. pp. 195 – 202, 2006.

45 G. E. A. P. A. Batista *et al.*, “Classification of Live Moths Combining Texture, Colour and Shape Primitives,” in *Machine Learning and Application (ICMLA), 2012 Ninth International Conference*, Washington D. C., 2010, pp. 903 – 906.

46 X. Yu *et al.*, “Measuring Geometrical Features of Insect Specimens using Image Analysis,” in *Proceedings of the Third Asian Conference for Information Technology in Agriculture*, Beijing, China, 2002, pp. 591 – 595.

47 S. Flandin, “Colour image analysis on Lepidoptera,” Final year project report, Elec. Eng. Dept., Univ. Hull, Hull, 1996.

48 S. Kang *et al.*, “Identification of butterfly species with a single neural network system,” in *Journal of Asia-Pacific Entomology*, vol. 15, Amsterdam, Netherlands: Elsevier B. V. 2012, pp. 431 – 435.

49 N. Larios *et al.*, “Automated insect identification through concatenated histograms of local appearance features: feature vector generation and region detection for deformable objects,” in *Machine Vision and Applications*, vol. 19, issue 2, New York, NY: Springer-Verlag. pp. 105 – 123, 2008.

50 PlantSnap. *Indentify Plants with An App*, plantsnap.com. [Online]. Available: https://plantsnap.com. [Accessed: Sept. 27, 2018].

51 PictureThis. *PictureThis – Let’s Indentify Plants – Together.*, picturethisai.com. [Online]. Available: https://www.picturethisai.com. [Accessed: Sept. 27, 2018].

52 My Garden Answers. *Garden Answers – Identify your Plant Instantly,* gardenanswers.com. [Online]. Available: http://www.gardenanswers.com. [Accessed: Sept. 27, 2018].

53 B. Mars. *iPlant App – New Herbal App for iPhone by Brigitte Mars*, brigittemars.com. [Online]. Available: https://brigittemars.com/iplant-app/. [Accessed: Sept. 27, 2018].

54 SmartPlant. *Smart Homes start with Smart Plants*, smartplantapp.com. [Online]. Available: https://www.smartplantapp.com. [Accessed: Sept. 27, 2018].

55 Gardenista. *Plantifier: A Gardening App to ID Plants*, gardenista.com. [Online]. Available: https://www.gardenista.com/posts/plantifier-a-gardening-app-to-id-plants/. [Accessed: Sept. 27, 2018].

56 Phuong Bui. *Identify any plant like a pro with Plant Identification app on iOS*, plantidentificationapp.wordpress.com. [Online]. Available: https://plantidentificationapp.wordpress.com/2018/04/13/the-journey-begins/. [Accessed: Sept. 27, 2018].

57 Apple. *Smart Identifier: Plant+Insect*, itunes.apple.com. [Online]. Available: https://itunes.apple.com/GB/app/id1265061997. [Accessed: Sept 27, 2018].

58 Pl@ntNet. *Identify, explore and share your observations of wild plant*, identify.plantnet-project.org. [Online]. Available: https://identify.plantnet-project.org. [Accessed: Sept. 27, 2018].

59 Apple. *Garden Flower Identification – Plant Identifier Free*, itunes.apple.com. [Online]. Available: https://itunes.apple.com/gb/app/garden-flower-identification-plant-identifier-free/id1128290219?mt=8. [Accessed: Sept. 27, 2018].

60 Apple. *Plant Identification & Info*, itunes.apple.com. [Online]. Available: https://itunes.apple.com/gb/app/plant-identification-info/id1409523353?mt=8. [Accessed: Sept. 27, 2018].

61 Softonic. *WildPlantID – Plant Identifier*, wildplantid-plant-identifier.en.softonic.com. [Online]. Available: https://wildplantid-plant-identifier.en.softonic.com/iphone. [Accessed: Sept. 27, 2018].

62 J. P. Cohn, "*Citizen science: can volunteers do real research?*" in *BioScience*, vol. 58, pp. 192 – 197, 2008.

63 M. J. O. Pocock and D. M. Evans, "The Success of the Horse-Chestnut Leaf-Miner, *Cameraria ohridella*, in the UK Revealed with Hypothesis-Led Citizen Science", in *PloS ONE*, vol. 9, no. 1, Jan. 2014. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0086226. [Accessed: Sept. 28, 2018].

64 Natural Apptitude (Jun. 28, 2011). "*Conker Tree Science: Leaf Watch*," iTunes Preview. [Online]. Available: https://itunes.apple.com/gb/app/conker-tree-science-leaf-watch/id445371129. [Accessed: June 3, 2015].

65 Jisc. "*Leaf Watch*," Leaf Watch | Jisc. [Online]. Available: http://www.jisc.ac.uk/leaf-watch. [Accessed: June 3, 2015].

66 University of Bristol. "*Results*", Conker Tree Science: Leaf Watch – Overview. [Online]. Available: http://leafwatch.naturelocator.org/results. [Accessed: June 3, 2015].

67 University of Bristol. "*Nationwide mission to save conkers from alien invaders gets underway*," Bristol University | News | 2011: Conker Tree Science. [Online]. Available: http://www.bristol.ac.uk/news/2011/7744.html. [Accessed: June 3, 2015].

68 Conker Tree Science. "*Identifying mines and blotches,*" Conker Tree Science. [Online]. Available: http://www.conkertreescience.org.uk/identifying-mines-and-blotches. [Accessed: Sept. 28, 2018].

69 M. Gilbert *et al.*, "Forecasting Cameraria ohridella invasion dynamics in recently invaded countries: from validation to prediction," in *Journal of Applied Ecology*, vol. 42, issue 5, pp. 805 – 813, Oct. 2005.

70 Deschka and Dimić, "*Cameraria ohridella sp. N.* (Lep., Lithocolletidae)," 1986.

71 D. C. Lees *et al.*, "Tracking origins of invasive herbivores through herbaria and archival DNA: the case of the horse-chestnut leaf miner," in *Frontiers in Ecology and the Environment*, vol. 9, no. 6, pp. 322 – 328, 2011.

72 R. Valade *et al.*, "Mitochondrial and microsatellite DNA markers reveal a Balkan origin for the highly invasive horse-chestnut leaf miner *Cameraria ohridella* (Lepidoptera, Gracillariidae)," in *Molecular Ecology*, vol. 18, no. 16, pp. 3458 – 3470, 2009.

73 Girardoz *et al.*, "Recruitment of native parasitoids by an exotic leaf miner, *Cameraria ohridella*: host – parasitoid synchronization and influence of the environment", in *Agricultural and Forest Entomology*, vol. 8, pp. 49 – 56, 2006.

74 Šefrova and Laštůvká, "Dispersal of the horse-chestnut leafminer, *Cameraria ohridella* Deschka & Dimiè, 1986, in Europe: its course, ways and causes (Lepidoptera: Gracillariidae)," in *Entomologische Zeitschrift, Stuttgart*, vol. 111, pp. 194 – 198, 2001.

75 B. Syeryebryennikov *et al.*, "Ecology and control of horsechestnut leaf miner (*Cameraria ohridella*)," in *16th International Environmental Project Olympiad*, 2008, pp. 1 – 67.

76 H. Šefrová, "Control possibility and additional information on the horse-chestnut leafminer *Cameraria ohridella* Deschka and Dimic (Lepidoptera, Gracillariidae)," in *Acta Universitas Argriculturae et Silviculturae Mendelianae* Brunensis, vol. 13, pp. 121 – 127, 2001.

77 H. Pschorn-Walcher, "Freiland-Biologie der eingeschleppten Roßkastanien-Miniermotte *Cameraria ohridella* Deschka et Dimic (Lep., Gracillariidae) im Wienerwald," in *Linzer biologische* Beitrage, vol. 26, pp. 633 – 642, 1994.

78 J. Freise and W. Heitland, "Aspects of the biology and ecology of the horse-chestnut leaf miner, *Cameraria ohridella* Deschka & Dimiè, 1986, a new pest on *Aesculus hippocastanum*," in *Mitteilungen der Deutschen Gesellschaft für Allgemeine und Angewandte Entomologie*, vol. 13, pp. 135 – 139, 2001.

79 D. C. Lees *et al.* (Jun. 22, 2009). *Position at rest.* Encyclopedia of Life [Online]. Available: http://eol.org/data_objects/13144359. [Accessed: June 3, 2015].

80 D. C. Lees *et al.* (Jun. 22, 2009). *Eggs laid on leaf of horse chestnut.* Encyclopedia of Life [Online]. Available: http://eol.org/data_objects/13144358. [Accessed: June 3, 2015].

81 D. C. Lees *et al.* (Jun. 22, 2009). *Cameraria ohridella first larval instar, sap feeding with empty eggshell and start of mine.* Encyclopedia of Life [Online]. Available: http://eol.org/data_objects/13144350. [Accessed: June 3, 2015].

82 D. C. Lees *et al.* (Jun. 22, 2009). *Cameraria ohridella second spinning larval instar.* Encyclopedia of Life [Online]. Available: http://eol.org/data_objects/1314 4351. [Accessed: June 3, 2015].

83 D. C. Lees *et al.* (Jun. 22, 2009). *Incremental distribution of Cameraria ohridella, 1984 – 2007*. Encyclopedia of Life [Online]. Available: http://eol.org/data_objects/13144354. [Accessed: June 3, 2015].

84 C. Thalmann *et al.* "Effects of defoliation by horse chestnut leafminer (*Cameraria* ohridella) on reproduction in *Aesculus hippocastanum*," in *Trees*, vol. 17, no. 5, pp. 383 – 388, 2003.

85 C. Tomiczek and H. Krehan, "The horsechesnut leafmining moth (*Cameraria Ohridella*): A new pest in Central Europe," in *Journal of Arboricult.*, vol. 24, no. 3, pp. 144 – 148, 1998.

86 V. Skuhravy, "Zusammenfassende Betrachtung der Kenntnisse iiber die Roskastanien miniermotte *Cameraria ohridella* Desch & Dem Lep Gracillariidae," in *Pest Science*, vol. 72, pp. 95 – 99, 1999.

87 G. Raimondo *et al.*, "New optical and near-infrared surface brightness fluctuation models. II. young and intermediate-age stella populations," in *The Astronomical Journal*, vol. 130, pp. 2625 – 2646, 2005.

88 M. Nardini *et al.*, "Regulated gene expression of hyaluronan synthases during Xenopus laevis development," in *Gene Expr. Patterns*, vol. 4, no. 3, pp. 303 – 308, 2004.

89 G. C. Percival *et al.*, "The impact of horse chestnut leaf miner (*Cameraria ohridella* Deschka and Dimić; HCLM) on vitality, growth and reproduction of *Aesculus hippocastanum L.*," in *Urban forestry & urban greening*, vol. 10, no. 1, pp. 11 – 17, 2011.

90 J. Vangjeli *et al.*, *Red Book (Threatened Plants, Plants Associations and Animals).* Academy of Science of Albania, Tirana, 1997.

91 D. C. Lees *et al.*, "Tracking origins of invasive herbivores through herbaria and archival DNA: the case of the horse-chestnut leaf miner," in *Frontiers in Ecology and the Environment*, vol. 9, no. 6, pp. 322 – 328, 2011.

92 J. Mertelík *et al.*, "Aesculus hippocastanum with resistant behaviour towards the larvae of *Cameraria ohridella* in the Czech Republic," in *Cameraria ohridella and other invasive leaf-miners in Europe. Proceedings of the 1st, International Cameraria Symposium*, Prague, 2004, pp. 24 – 27.

93 J. F, Freise, *Untersuchungen zur Biologie und Ökologie der Rosskastanien-Miniermotte (Cameraria ohridella Desch. & Dim. 1986 (Lepidoptera: Gracillariidae).* PhD thesis. Munich, Technical Univ. of Munich, 2001.

94 V. Bačovský *et al.*, "Genetic diversity of chestnut tree in relation to susceptibility to leaf miner (*Cameraria ohridella* Deschka & Dimič)," in *Trees*, vol. 31, pp. 753 – 763, 2017.

95 N. Straw and C. Tilbury, "Host plants of the horse-chestnut leaf-miner (*Cameraria ohridella*), and the rapid spread of the moth in the UK 2002 – 2005," in *Arboricultural Journal*, vol. 29, no. 2, pp. 83 – 99, 2006.

96 H. J. Hudson, "*Guignardia* leaf blotch of horsechestnut," in *Trans. Br. Myco. Soc.,* vol. 89, pp. 400 – 401, 1987.

97 K. Pastircakova *et al.*, "*Guignardia aesculi* on species of *Aesculus*: New records from Europe and Asia, in *Mycotaxon.*, vol. 108, pp. 287 – 296, 2009.

98 D. C. Lees *et al.* (Jun. 22, 2009). *Position at rest.* Encyclopedia of Life [Online]. Available: http://eol.org/data_objects/13144349. [Accessed: June 3, 2015].

99 The Royal Horticultural Society. *Horse chestnut leaf blotch*, rhs.orhg.uk. [Online]. Available: https://www.rhs.org.uk/advice/profile?PID=200. [Accessed: Apr. 10, 2020].

100 R. Jagiełło *et al.*, "Ecophysiological aspects of the interaction between *Cameraria ohridella* and *Guignardia aesculi* on *Aesculus hippocastanum*," in *Dendrobiology*, vol. 78, pp. 146 – 156, 2017.

101 U. Gerson and S. Applebaum. (Dec. 23, 2014). *Lyonetia clerkella (Linnaeus).* Plant Pests of the Middle East. [Online]. Available: http://www.agri.huji.ac.il/mepests/pest/Lyonetia_clerkella/. [Accessed: Sept 27, 2018].

102 National Biodiversity Network Atlas. *Lyonetia clerkella (Linnaeus, 1758) Apple Leaf Miner*, species.nbnatlas.org. [Online]. Available: https://species.nbnatlas.org/species/NHMSYS0000503134. [Accessed: Apr. 10, 2020].

103 Encyclopedia of Life. *Lyonetia clerkella*, eol.org. [Online]. Available: http://eol.org/data_objects/30158219. [Accessed: Sept 27, 2018].

104 The Royal Horticultural Society. *Apple leaf mining moth*, rhs.org.uk. [Online]. Available: https://www.rhs.org.uk/advice/profile?PID=893. [Accessed: Sept 27, 2018].

105 B. Pitkin *et al.* (May 10, 2018). *Lyonetia clerkella (Linnaeus, 1758).* The leaf and stem mines of British flies and other insects. [Online]. Available: www.ukflymines.co.uk/Moths/Lyonetia_clerkella.php. [Accessed: Sept 27, 2018].

106 Flemish Entomological Society. *Lyonetia clerkella (Linnaeus, 1758).* Catalogue of the Lepidoptera of Belgium. [Online]. Available: https://projects.biodiversity.be/lepidoptera/species/5186/. [Accessed: Apr. 10, 2020].

107 Plant parasites of Europe. *Lyonetia clerkella (Linnaeus, 1758) apple leaf-miner.* bladmineerders.nl. [Online]. Available: https://bladmineerders.nl/parasites/animalia/arthropoda/insecta/lepidoptera/ditrysia/yponomeutoidea/lyonetiidae/lyonetia/lyonetia-clerkella/. [Accessed: Apr. 10, 2020].

108 British Leafminers. *21.001 Lyonetia clerkella (Linnaeus, 1758).* leafmines.co.uk. [Online]. Available: http://www.leafmines.co.uk/html/Lepidoptera/L.clerkella1.htm. [Accessed: Sept 27, 2018].

109 Ken Muir Ltd. (Aug. 05, 2016). *Apple Leaf Miner.* Ken Muir. [Online]. Available: https://www.kenmuir.co.uk/image/data/pdf/Fact%20Sheets/APPLE%20LEAF%20MINER%20Aug%202016.pdf. [Accessed: Apr. 10, 2020].

110 I. Kimber (2018). *Apple Leaf Miner Lyonetia clerkella.* UK Moths. [Online]. Available: ukmoths.org.uk/species/191ehaviou-clerkella. [Accessed: Sept 27, 2018].

111 The National Biodiversity Network. *Lyonetia clerkella.* Easymap.nbnatlas.org. [Online]. Available: https://easymap.nbnatlas.org/EasyMap?tvk=NHMSYS0000503134&w=400&b0fill=39B0D5. [Accessed: Sept 27, 2018].

112 B. Pitkin *et al.* (Dec. 6, 2017). *Phytomyza ilicis Curtis, 1846.* The leaf and stem mines of British flies and other insects. [Online]. Available: www.ukflymines.co.uk/Flies/Phytomyza_ilicis.php. [Accessed: Sept 27, 2018].

113 W. N. Ellis, "Biological notes on the holly leaf miner, *Phytomyza ilicis* (Diptera: Agromyzidae)," in *Entomologische Berichten*, vol. 60, no. 9, ref. 25, pp. 165 – 170, 2000.

114 Field Studies Council. *Holly leaf miner*. biology-fieldwork.org. [Online]. Available: https://www.biology-fieldwork.org/gcse/food-chains/holly-leaf-miner/. [Accessed: Apr. 10, 2020].

115 M. Glackin *et al*. (Mar. 2006). *What happened to the holly leaf miner? Studying real food chains*. School Science Review 87 (320). [Online]. Available: www.field-studies-council.org/documents/Pubs/SSR%202006%20HLM.pdf. [Accessed: Sept 27, 2018].

116 C. Saunders. (Jan. 18, 2012). *The holly leaf miner: Phytomyza ilicis*. Woodlands.co.uk – Flora & Fauna. [Online]. Available: https://www.woodlands.co.uk/blog/flora-and-fauna/the-holly-leaf-miner-phytomyza-ilicis/ [Accessed: Apr. 10, 2020].

117 Diptera.info. *Phytomyza ilicis (female)*, diptera.info [Online]. Available: https://diptera.info/photogallery.php?photo_id=9021. [Accessed: Sept 27, 2018].

118 British Leafminers. Phytomyza ilicis Curtis, 1846, leafmines.co.uk. [Online]. Available: http://www.leafmines.co.uk/html/Diptera/P.ilicis2.htm. [Accessed: Sept 27, 2018].

119 J. J. Dombroskie (Sep. 1, 2013). *Ilex miner – Phytomyza ilicis*. BugGuide.Net [Online]. Available: https://bugguide.net/node/view/885399. [Accessed: Sept 27, 2018].

120 R. J. Bevan and G. N. Greenhalgh, "*Rhytisma acerinum* as a biological indicator of pollution," in *Environ. Pollut.*, vol. 10, pp. 271 – 285, 1976.

121 T. Hsiang and X. L. Tian, "Sporulation and Identity of Tar Spot of Maple in Canada," in *Acta Silv. Lign. Hung., Spec. Edition*, pp. 71 – 74, 2007.

122 S. M. Karami *et al*., "Biotechnical control of tar spot (*Rhytisma acerinum*) disease on velvet maple *(Acer velutinum* Boiss) *in vitro*, in *Journal of Forest Sci.*, vol. 60, no. 8, pp. 330 – 335, 2014.

123 M. Lapointe and J. Brisson, "Tar spot disease on Norway maple in North America: Quantifying the impacts of a reunion between an invasive tree species and its adventive natural enemy in an urban forest," in *Écoscience*, vol. 18, no. 1, pp. 63 – 66, 2011.

124 C. Solomon and T. Breckon, *Fundamental of Digital Image Processing – A Practical Approach with Examples in Matlab*, 1st ed. Chichester: J. Wiley & Sons, 2010.

125 S. Jayaraman *et al.*, *Digital Image Processing*, 3rd ed. New Delhi: Tata McGraw Hill, 2010.

126 T. Berk *et al.*, “A human factors study of color notation systems for computer graphics” in *Communications of the ACM*, vol. 25, no. 8, pp. 547 – 550, Aug. 1982.

127 W.D. Wright, “A re-determination of the trichromatic coefficients of the spectral colours,” in *Transactions of the Optical Society*, vol. 30, issue 4, pp. 141 – 164, 1929.

128 Intel IPP 2018. “Color Models” in *Intel® Integrated Performance Primitives Developer Reference Volume 2: Image Processing*, Intel Corporation, 2018. [Online]. Available: https://software.intel.com/sites/default/files/managed/db/72/ippi_0.pdf. [Accessed: Sept. 28, 2018].

129 E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*, 2nd ed. London: Academic Press, 1997.

130 A. R. Weeks and G. E. Hague, "Color segmentation in the HSI color space using the k-means algorithm", in *Nonlinear Image Processing VIII*, vol. 3026, pp. 143 – 154, 1997.

131 K. Lin *et al.*, “Color image segmentation method Summary,” in *Journal of Image and Graphics*, vol. 10, no. 1, pp. 1 – 10, 2005.

132 P. K. Sahoo *et al.*, “A survey of thresholding techniques,” in Computer Vision, Graphics, and Image Processing, vol. 41, no. 2, pp. 233 – 260, 1988.

133 N. Otsu, “A threshold selection method from gray level histogram,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, pp. 62 – 66, 1979.

134 The MathWorks, Inc. “*Convert image to binary image*,” mathworks.com. [Online]. Available: http://www.mathworks.co.uk/help/images/ref/im2bw.html [Accessed: Apr. 19, 2013]

135 J. S. Weska, “A survey of threshold selection techniques,” in *Comput. Graph. Image Process*, vol. 7, pp. 259 – 265, 1978.

136 P. S. Liao *et al.*, “A fast algorithm for multilevel thresholding,” in J. Inf. Sci. Eng., vol. 15, no. 5, pp. 713 – 727, 2001.

137 A. K. Jain, “Edge detection,” in *Fundamentals of Digital Image Processing*, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, ch. 4, sec. 1, pp. 347 – 357.

138 L. G. Roberts, “Machine Perception of Three-Dimensional Solids,” in *Optical and Electro-Optical Information Processing*, J. T. Tippet (Ed.), Cambridge, MA: MIT Press, 1965.

139 J. M. S. Prewitt, "Object Enhancement and Extraction," in *Picture Processing and Psychopictorics*, B. S. Lipkin and A. Rosenfeld (Ed.), New York, NY: Academic Press, 1970.

140 I. E. Sobel, "Camera Models and Machine Perception," Ph.D dissertation, Stanford University, Palo Alto, CA., 1970.

141 D. Marr and E. Hildreth, "Theory of Edge Detection," *Proc. R. Soc. Lond.*, vol. B207, pp. 187 – 217, 1980.

142 J. Canny, "A Computational Approach for Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 1293 – 1299, 1986.

143 L. Chen *et al.*, "An Automatic Diagnostic System for CT Liver Image Classification," in *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 5, pp. 783 – 794, June 1998.

144 E. D. Chesmore, "The Automated Identification of Taxa: Concepts and Applications," in *Automated Taxon Identification in Systematics*, CRC Press, pp 85-100, 2007.

145 E. D. Chesmore, "Methodologies for automating the identification of species," in *Proceedings of the Inaugural Meeting of the BioNET-INTERNATIONAL Group for Computer-Aided Taxonomy*, Cardiff, pp. 3 – 12, 1997.

146 A. N. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

147 B. S. Everitt, *Cluster Analysis*. New York: John Wiley & Sons, Inc., 1974.

148 A. K. Jain and R. C. Dubes, "Feature definition in pattern recognition with small sample size," in *Pattern Recognition*, vol. 10, pp. 85 – 97, 1978.

149 A. C. Sanderson and A. K. C. Wong, "Pattern trajectory analysis of nonstationary multivariate data," in *IEEE Transactions on Systems, Man and Cybernetics*, vol. 10, pp. 384 – 392, 1980.

150 R. Zimmerman *et al.*, "A comparison of the accuracy of four methods of clustering job," in *Applied Psychological Measurement*, vol. 6, pp. 353 – 266, 1982.

151 C. R. Rao, "Clustering analysis applied to a study of race mixture in human populations," in *Classification and Clustering*, J. Van Ryzin, Eds. New York: Academic Press, Inc., pp. 175 – 197, 1977.

152 K. Garland, "An experiment in automatic hierarchical document classification," in *Information Processing and Management*, vol. 19, pp. 113 – 120, 1983.

153 G. Salton *et al.*, "A vector space model for automatic indexing," in *Communications of the ACM*, vol. 18, pp. 613 – 620, 1975.

154 L. R. Rabiner and J. G. Wilpon, "Considerations in applying clustering techniques to speaker-independent word recognition," in *Journal of the American Statistical Association*, vol. 66, pp. 663 – 673, 1979.

155 R. Hoffman and A. K. Jain, "Segmentation and classification of range images," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 9, pp. 608 – 620, 1987.

156 G. Stockman, "Object detection via image registration," in *Pattern Recognition in Practice*, E. S. Gelsema and L. N. Kanal, Eds. New York: Elsevier Borth-Holland, Inc., pp. 75 – 86, 1980.

157 M. Tuceryan and A. K. Jain, "Texture Analysis," in *Handbook of Pattern Recognition and Computer Vision*, C. H. Chen *et al.*, Eds. World Scientific Publishing Company, pp. 235 – 276, 1993.

158 S. W. Zucker and K. Kant, "Multiple-level representations for texture discrimination," in *Proceedings of IEEE Conference on Pattern Recognition and Image Processing*, Dallas, TX, pp. 609 – 614, 1981.

159 J. Chen and A. K. Jain, "A structural approach to identify defects in textured images," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Beijing, China, pp. 29 – 32, 1988.

160 P. Dewaele *et al.*, "Texture inspection with self-adaptive convolution filters," in *Proceedings of 9th International Conference on Pattern Recognition*, Rome, Italy, Nov. 1988, pp. 58 – 60.

161 R. W. Conners *et al.*, "Identifying and locating surface defects in wood: Part of an automated lumber processing system," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 92 – 105, 1988.

162 J. M. Coggins, "A framework for texture analysis based on spatial filtering," Ph.D. thesis, Dept. Comput. Sci., Michigan State University, East Lansing, 1982.

163 O. R. Mitchell and S. C. Carlton, "Image segmentation using a local extrema texture measure," in *Pattern Recognition*, vol. 10, p. 205 – 210, 1978.

164 B. J. Schachter *et al.*, "Some experiments in image segmentation by clustering of local feature value," in *Pattern Recognition*, vol. 11, pp. 19 – 28, 1978.

165 G. B. Coleman and H. C. Andrews, "Image segmentation by clustering," in *Proceedings of the IEEE*, vol. 67, pp. 773 – 785, 1979.

166 D. L. Davies and D. W. Bouldin, "A cluster separation measure," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 95 – 110, 1982.

167 N. R. Pal and S. K. Pal, "A review on image segmentation techniques," in *Pattern Recognition*, vol. 26, issue 9, pp. 1277 – 1294, 1993.

168 M. Egmont-Petersen *et al.*, "Image processing with neural networks – a review," in *Pattern Recognition*, vol. 35, issue 10, pp. 2279 – 2301, 2002.

169 J. A. Hertz, *Introduction to the theory of neural computation*. Addison-Wesley/Addison Wesley Longman, 1991.

170 W. S. MuCulloch and W. Pitts, "A Logical calculus of Ideas Immanent in Nervous Activity," in *Bulletin of Mathematical Biophysics*, vol. 5, issue 4, pp. 115 – 133, 1943.

171 F. Rosenblatt, *Principles of Neurodynamics*. New York: Spartan, 1962.

172 P. D. Wasserman, *Neural Computing*. New York: Van Nostrand Reinhold, 1989.

173 M. L. Minsky and S. A. Papert, *Perceptrons*. Cambridge: MIT Press, 1969.

174 T. Kohonen, "Self-organization and associative memory," in *Series in Information Sciences*, vol. 8, Berlin: Springer Verlag, 1984.

175 D. O. Hebb, *Organization of Behaviour*. New York: Science Editions, 1961.

176 A. E. Bryson and Y. C. Ho, *Applied Optimal Control*. New York: Blaisdell, 1969.

177 P. J. Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences," Ph,D. thesis, Harvard University, 1974.

178 D. B. Parker, "Learning Logic", in *Technical Report TR-47*, Center for Computational Research in Economics and Management Science, MIT, Cambridge, MA, 1985.

179 D. E. Rumelhart *et al.*, "Learning Representations by Back-Propagating Errors," in *Nature*, vol. 323, pp. 533 – 536, 1986.

180 D. E. Rumelhart *et al.*, “Learning Internal Representations by Error Propagation,” in *Parallel Distributed Processing*, vol. 1, ch. 8, 1986.

181 Google Developers. *ML Practicum: Image Classification*, Machine Learning. [Online]. Available: https://developers.google.com/machine-learning/practica/image-classification. [Accessed: Apr. 10, 2020].

182 S. Bhatt, *Understanding the basics of CNN with image classification*, becominghuman.ai. [Online]. Available: https://becominghuman.ai/understanding-the-basics-of-cnn-with-image-classification-7f3a9ddea8f9. [Accessed: Apr. 10, 2020].

183 MissingLink, *Convolutional Neural Network Tutorial: From Basic to Advanced*, missinglink.ai. [Online]. Available: https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-network-tutorial-basic-advanced/

184 B. Nickolay *et al.*, “Parameter Optimization of an Image Processing System using Evolutionary Algorithms,” in *Proceedings of the 7th International Conference, Computer Analysis of Images and Patterns ‘97*, Kiel, Germany, Sept. 1997, vol. 1296, pp. 637 – 644.

185 I. Aizenberg, “Solving the XOR and parity N problems using a single universal binary neuron,” in *Soft Computing*, vol. 12, no. 3, pp. 215 – 222, 2008.

186 E. R. Hruschka *et al.*, “A Survey of Evolutionary Algorithms for Clustering”, in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 39, no. 2, pp. 133 – 155, 2009.

187 A. K. Bhandari *et al.*, “Tsallis entropy based multilevel thresholding for colored satellite image segmentation using evolutionary algorithms,” in *Expert systems with applications*, vol. 42, issue 22, pp. 8707 – 8730, 2015.

188 R. F. Martinez *et al.*, “Use of decision tree models based on evolutionary algorithms for the morphological classification of reinforcing nano-particle aggregates”, in *Computational materials science*, vol. 92, pp. 102 – 113, 2014.

189 H. Spencer, *The principles of biology*, vol. 1. 1864, p. 444.

190 D. Dasgupta and Z. Michalewicz, “Evolutionary Algorithms – An Overview,” in *Evolutionary Algorithms in Engineering Applications*, D. Dasgupta and Z. Michalewicz, Eds. Berlin, Heidelberg, New York: Springer-Verlag, 1997, pp. 3 – 28.

191 M. Mitchell, *An Introduction of Genetic Algorithms*. MIT Press, 1998.

192 D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

193 J. Holland, *Adoption in natural and artificial systems*. The University of Michigan Press, 1975.

194 C. B. Pettey *et al.*, "A Parallel genetic algorithm," in *Proceedings of 2nd International Conference on Genetic Algorithms*, pp. 151 – 161, 1987.

195 D. J. Montana and L. D. Davis, "Training feedforward networks using genetic algorithms," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 89, Morgan Hoffman Publishers, Inc., pp. 762 – 767, 1989.

196 J. D. Schaffer *et al.*, "Combinations of genetic algorithms and neural networks: A survey of the state of the art," in *COGANN-92: International Workshop on Combinations of Genetic Algorithms and Neural Networks*, L. D. Whitely and J. D. Schaffer, Eds. IEEE Computer Society Press, 1992.

197 G. F. Miller *et al.*, "Designing neural networks using genetic algorithms," in *Proceedings of the Third International Conference on Genetic Algorithms*, J. D. Schaffer, Eds. Morgan Hoffman Publishers, Inc., 1989.

198 H. Kitano, "Designing neural networks using genetic algorithms with graph generation system," in *Complex Systems*, vol. 4, pp. 461 – 476, 1990.

199 D. J. Chalmers, "The evolution of learning: An experiment in genetic connectionism," in *Proceedings of the 1990 Connectionist Models Summer School*, D. S. Touretzky *et al.*, Eds. Morgan Hoffman Publishers, Inc., 1990.

200 B. Bhanu *et al.*, ""Adaptive image segmentation using a genetic algorithm," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 12, pp. 1543 – 1567, 1995.

201 G. F. Hughes, "On the mean accuracy of statistical pattern recognisers," in *IEEE Trans. Inf. Theory*, vol. 14, pp. 55 – 63, 1968.

202 G. B. Chittineni, "Efficient feature subset selection with probabilistic distance criteria," in *Inf. Sci.*, vol. 22, pp. 19 – 35, 1980.

203 I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. New York: Springer-Verlag, 2002.

204 K. Pearson, “On lines and planes of closest fit to systems of points in space,” in *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, series 6, vol. 2, pp. 559 – 572, 1901.

205 H. Hotelling, “Analysis of a Complex of Statistical Variables into Principal Components,” in *The Journal of Educational Psychology*, vol. 24, no. 6, pp. 417 – 441, 498 – 520, 1933.

206 S. Wold *et al.*, “Principal Component Analysis,” in *Chemometrics and Intelligent Laboratory Syst*ems, vol. 2, no. 1 – 3, pp. 37 – 52, Aug. 1987.

207 B. Flury and H. Riedwyl*, Multivariate Statistics: A Practical Approach*. New York: Chapman and Hall, Mar. 1988, pp. 181 – 183.

208 T. Clemensat, “The application of colour image analysis to Lepidoptera classification,” Second year training period report, Elec. Eng. Dept., Univ. Hull, Hull, vol. 1, 1997.

209 D. K. Mishra, "ADC testing using interpolated fast Fourier transform (IFFT) technique," in *International journal of electronics*, vol. 90, issue 7, pp. 459 – 469, 2003.

210 M. Silveira and A. Monteiro, “Automatic recognition and measurement of butterfly eyespot patterns,” in *BioSystems*, vol. 94, Amsterdam, Netherlands: Elsevier B. V. pp. 130 – 136, 2012.

211 A. Afshar *et al.*, “Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation,” in *Journal of the Franklin Institute*, vol. 344, pp. 452 – 462, Aug. 2007.

212 S. Bonadies *et al.*, “A survey of unmanned ground vehicles with applications to agricultural and environmental sensing,” in *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping*, vol. 9866, International Society for Optics and Photonics, May 2016.

213 C. R. Macadam, “The current status of *Heptagenia Longicauda* (Stephens 1835) (*Ephemeroptera: heptageniidae*),” in *Entomologist’s Monthly Magazine*, vol. 142, p. 163, July 2006.

214 M. Sonka *et al.*, *Image Processing, Analysis, and Machine Vision*, 3rd ed. Ontario: Thomson, 2008.

215 J. Sachs. *Digital Image Basics*, Digital Light & Colours. [Online]. Available: https://www.dl-c.com/Temp/downloads/Whitepapers/Basics.pdf. [Accessed: Apr. 10, 2020].

216 Hewlett-Packard. *HP Photosmart 5520 series*, hp.com. [Online]. Available: http://h10032.www1.hp.com/ctg/Manual/c03353113. [Accessed: Apr. 10, 2020].

217 Canon. *Canon CanoScan LiDE 110 Specifications*, canon.co.uk. [Online]. Available: https://www.canon.co.uk/scanners/flatbed-scanners/canoscan_lide_110 /specification.html. [Accessed: Apr. 10, 2020].

218 A. Skodras *et al.*, "The JPEG 2000 Still Image Compression Standard," in *Signal Processing Magazine*, vol. 18, no. 5, IEEE, pp. 36 – 58, 2001.

219 In Easy Steps Team. *Understanding Camera Raw*, ineasysteps.com. [Online]. Available: https://ineasysteps.com/understanding-camera-raw/. [Accessed: Apr. 10, 2020].

220 M. Reichmann. *Understanding RAW Files Explained*, Luminous Landscape. [Online]. Available: https://luminous-landscape.com/understanding-raw-files-explained/. [Accessed: Apr. 10, 2020].

221 MathWorks. *Image Processing Toolbox*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/ products/image.html. [Accessed: Apr. 10, 2020].

222 MathWorks. *Deep Learning Toolbox*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/deeplearning/index.html. [Accessed: Sept. 25, 2018].

223 MathWorks. *Statistics and Machine Learning Toolbox*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/stats/. [Accessed: Sept. 25, 2018].

224 A. Quinquis, "MATLAB software presentation," in *Digital Signal Processing Using MATLAB*, 1st ed. London: ISTE Ltd. And Hoboken, NJ: John Wiley and Sons, Ltd., 2008, ch. 1, sec. 1.1, pp. 1 – 3.

225 MathWorks. *Expressing image locations*, mathworks.com. [Online]. Available: http://www.mathworks.co.uk/help/images/image-coordinate-systems.html. [Accessed: Apr. 28, 2013].

226 MathWorks. *Working with Images in MATLAB Graphics*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/matlab/creating_plots/wor

king-with-images-in-matlab-graphics.html. [Accessed: Sept. 25, 2018].

227 MathWorks. *Convert Between Image Types*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/images/convert-between-image-types.html. [Accessed: Sept. 25, 2018].

228 J. G. Moik. *Digital Processing of Remotely Sensed Images*. Washington DC: NASA SP-431, 1980.

229 A. Rosenfield and A. C. Kak. *Digital Picture Processing*, 2nd ed. New York: Academic Press, 1982.

230 MathWorks. *Image Segmentation*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/images/image-segmentation.html. [Accessed: Sept. 25, 2018].

231 MathWorks. *Morphological Operations*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/images/morphological-filtering.html. [Accessed: Sept. 25, 2018].

232 H. Murase and S. K. Nayar, "Visual learning and recognition of 3-D objects from appearance," in *International journal of computer vision*, vol. 14, no. 1, Dordrecht, Netherlands: Kluwer Academic. 1995, pp. 5 – 24.

233 M. J. Swain and D. H. Ballard, "Color indexing," in *International journal of computer vision*, vol. 7, no. 1, Dordrecht, Netherlands: Kluwer Academic. pp. 11 – 32, 1991.

234 B. Schiele and J. Crowley, "Object recognition using multidimensional receptive field histograms," in *Computer Vision ECCV '96: 4th European Conference on Computer Vision Cambridge, UK, April 1518, 1996 Proceedings*, vol. 1, Cambridge. 1996, pp. 610 – 619.

235 MathWorks. *Regionprops*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/images/ref/regionprops.html. [Accessed: Sept. 25, 2018].

236 MathWorks. *Bwarea*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/images/ref/bwarea.html. [Accessed: Sept. 25, 2018].

237 MathWorks. *Bwperim*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/images/ref/bwperim.html. [Accessed: Sept. 25, 2018].

238 MathWorks. *fitnet*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/deeplearning/ref/fitnet.html. [Accessed: Sept. 25, 2018].

239 MathWorks. *Train Regression Models in Regression Learner App*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/stats/train-regression-models-in-regression-learner-app.html. [Accessed: Apr. 10, 2020].

240 MathWorks. *Regression Learner App*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/stats/regression-learner-app.html. [Accessed: Apr. 10, 2020].

241 O. Seidou *et al*., "Modeling ice growth on Canadian lakes using artificial neural networks," in *Water Resources Research*, vol. 42, issue 11, 2006.

242 MathWorks. *Plot Error Histogram for a Neural Network*, uk.mathworks.com. [Online]. Available: https://uk.mathworks.com/help/thingspeak/plot-error-histogram.html. [Accessed: Apr. 10, 2020].

243 Y. Chtioui *et al*., "A generalized regression neural network and its application for leaf wetness prediction to forecast plant disease," in *Chemometrics and Intelligent Laboratory Systems*, vol. 48, issue 1, pp. 47 – 58, 1999.

244 L. D. Schweinitz, "Dothidea pomigena," in *Transactions of the American Philosophical Society*, p. 232, 1832.

245 P. Miller, "Malus pumila," in *The Gardeners Dictionary*, ed. 8, Malus no. 3, 1768.