The extraction and classification of craquelure patterns for geographical analysis of fine art painting

Mouhanned El-Youssef
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
El-Youssef, Mouhanned, "The extraction and classification of craquelure patterns for geographical analysis of fine art painting" (2013). Electronic Theses and Dissertations. 4973.
https://scholar.uwindsor.ca/etd/4973
The extraction and classification of craquelure patterns for geographical analysis of fine art painting

By

Mouhanned El-Youssef

A Thesis
Submitted to the Faculty of Graduate Studies through the Department of Electrical and Computer Engineering in Partial Fulfillment of the Requirements for the Degree of Master of Applied Science at the University of Windsor

Windsor, Ontario, Canada

2013

© 2013 Mouhanned El-Youssef
The extraction and classification of craquelure patterns for geographical analysis of fine art painting

by

Mouhanned El-Youssef

APPROVED BY:

Dr. Wladyslaw Kedzierski
Department of Physics

Dr. Sazzadur Chowdhury
Department of Electrical and Computer Engineering

Dr. Maev, Principal Advisor
Department of Electrical and Computer Engineering / Physics

September 3, 2013
DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.
ABSTRACT

Craquelure represents the unique crack formations that form on a material with age. This thesis is specifically concerned with the craquelure patterns found on historical art paintings. Although the significance of craquelure has been noted for over 300 years, recent research has shown that these patterns are related to the materials and methods employed by the artist and that these clues may assist in the task of attribution. It has been shown that different art paintings constructed from the same geographical location exhibit similar craquelure formation patterns. With the intention of alleviating expertise on the subject of craquelure, the development of a framework for the geographical analysis of craquelure patterns has been attempted in literature. This thesis seeks to expand on these results with the intention of increasing the accuracy rate in the classification of craquelure to their corresponding geographical origins. Through the use of mathematical morphology and various image processing techniques, craquelure images were converted to binary images. Specific features were then extracted from the binary image and used in the classification process. Several different classifiers were tested and compared in this thesis.
DEDICATION

This thesis is dedicated to my loving parents for all their support and encouragement.
ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to all those who provided me with the help and support I needed along the way. First and foremost, I would like to express my sincere gratitude to my supervisor Dr. Roman Maev for giving me this amazing opportunity to work in his lab and for his continuous support, motivation, knowledge and encouragement during my studies.

I would also like to thank the people at the Hamilton Kerr Institute and notably Dr. Spike Bucklow for all the help and knowledge they have passed on to me. Without the support and advice of Dr. Bucklow, this project wouldn’t have been possible. Thank you for your contribution towards this project as well as your all your support and guidance you have provided for me.

I would also like to thank my committee members for their participation and agreeing to serve on the defence committee. Thank you for your time and efforts.

The people in my lab have also contributed to the completion of my thesis. I would like to thank Dmitry Gavrilov, Dr. Anthony Karloff, Dr. Andrew Chertov, Anthony Lui and Andrew Oullette for all their help, feedback and support during my masters. I would also like to thank my brother Ahmed El-Youssef for his tremendous contribution to this project during the summer he worked alongside me.

Last but not least, I would like to thank my Parents and my youngest brother Jamil for always being there for me unconditionally. They gave me the support and confidence I needed to overcome the many challenges that I have face thus far in life. I feel blessed for the all love and support I have had around me.
# TABLE OF CONTENTS

DECLARATION OF ORIGINALITY .............................................................................. iii

ABSTRACT ....................................................................................................................... iv

DEDICATION ..................................................................................................................... v

ACKNOWLEDGEMENTS ............................................................................................... vi

LIST OF TABLES ............................................................................................................. ix

LIST OF FIGURES ............................................................................................................. x

INTRODUCTION ............................................................................................................... 1

1.1 Structure of Paintings .............................................................................................. 2

1.1.1 Support ............................................................................................................. 2

1.1.2 Glue Sizing ....................................................................................................... 3

1.1.3 Ground ............................................................................................................. 3

1.1.4 Paint Film ....................................................................................................... 4

1.2 Research Motivation .............................................................................................. 5

1.2.1 Objectives ....................................................................................................... 5

CRAQUELURE EXTRACTION ........................................................................................ 6

2.1 Mathematical Morphology ...................................................................................... 7

2.2 Thresholding Techniques ...................................................................................... 11

2.2.2 Global Thresholding ..................................................................................... 15

2.2.3 Segmented Thresholding ............................................................................... 16

2.2.4 Offset Thresholding ...................................................................................... 18

2.2.5 Comparison of Thresholding Methods .......................................................... 24

2.3 Enhancing and Cleaning Crack Network .............................................................. 26

HIERARCHICAL REPRESENTATION OF CRAQUELURE ........................................ 28

3.1 Hierarchical Structure .......................................................................................... 28

3.1.1 Chain Codes ................................................................................................ 28

3.1.2 Node Determination ..................................................................................... 29

3.1.3 Summary of Structure ................................................................................ 30

3.2 Crack Tracking Algorithm .................................................................................. 31

3.3 Crack Network Pruning ...................................................................................... 32
FEATURE EXTRACTION AND CLASSIFICATION

4.1 Descriptions of Craquelure and Corresponding Features

4.1.1 Histogram Based Features

4.1.2 Statistical Based Features

4.1.3 Global Features

4.1.4 Summary of Features

4.2 Feature Selection

4.2.1 Scatter Degree

4.2.2 Principal Component Analysis (PCA)

CLASSIFICATION

5.1 Nearest Neighbour

5.1.1 Feature Normalization

5.1.2 Weighted Nearest Neighbour

5.1.2.2 Implementation

5.1.3 Staged kNN

5.1.3.1 Implementation

5.2 Discriminant Analysis (DA) Classifier

5.2.1 Linear Discriminant Analysis (LDA) Classifier

5.2.1.2 Results

5.2.2 Quadratic Discriminant Analysis (QDA) Classifier

5.2.2.2 Results

CONCLUSIONS

6.1 Future Work

Personal Publications and Presentations

WORKS CITED

VITA AUCTORIS
# LIST OF TABLES

Table 3.1. Craquelure structure and corresponding description .................................................... 31  
Table 4.1. List of features, their symbols and descriptions ............................................................ 48  
Table 5.1. Classification results of kNN for k = 7 and n = 12 using scatter degree ...................... 54  
Table 5.2. Classification results of kNN for k = 5 and n = 19 using PCA ..................................... 55  
Table 5.3. The k and n values that yield highest accuracy for French and Flemish craquelure.... 57  
Table 5.4. Classification result of staged-kNN classifier using scatter degree ............................. 58  
Table 5.5. Classification result of staged-kNN classifier using PCA ............................................ 59  
Table 5.6. Classification accuracy for LDA classifier for n = 18 using scatter degree ................. 63  
Table 5.7. Classification accuracy for LDA classifier for n = 18 using PCA ............................... 63  
Table 5.8. Classification accuracy for QDA classifier for n = 16 using scatter degree ................. 65  
Table 5.9. Classification accuracy for QDA classifier for n = 16 using PCA ............................... 66
LIST OF FIGURES

Figure 2.1. (a) Original Image; (b) using canny edge detector; (c) mathematical morphology...... 7
Figure 2.2. Various structuring elements; (a) disk shaped SE with radius = 5; (b) vertical SE; (c) horizontal SE......................................................................................................................... 8
Figure 2.3. (a) Original image; (b) Opening operation; (c) Closing operation ......................... 9
Figure 2.4. (a) Original craquelure image; (b) image after performing a closing operation...... 10
Figure 2.5. Black Top-Hat transformed image .............................................................................. 11
Figure 2.6. (a) Original French craquelure image; (b) After using top-hat transformation; (c) Thresholding image using Otsu’s method for single k=0.28 value ......................................................................... 14
Figure 2.7. η values for a range of k for image in Figure 2.6. Maximum occurs at k=0.28 ......... 15
Figure 2.8. (a) Original Flemish craquelure image; (b) Thresholded using global k value ......... 16
Figure 2.9. (a) Original image with segmented regions; (b) k values for each region; (c) final result............................................................................................................................................... 17
Figure 2.10. (a) Original Dutch image; (b) k values for each region; (c) final result ................. 18
Figure 2.11. (a) Original Italian craquelure image; (b) Offset threshold image without minimum k threshold; (c) offset thresholded image with minimum k threshold ................................................................. 20
Figure 2.12. (a) Original Dutch Craquelure Image; (b) Segmented image showing k values. Right image is offseted; (c) images after thresholding (d) final image after multiplying the two previous images together ........................................................................................................................................ 23
Figure 2.13. (a) Thresholded using global thresholding technique; (b) using segmented thresholding technique; (c) using offset thresholding technique; (d) original image ............... 25
Figure 2.14. Isolated pixel .............................................................................................................. 27
Figure 3.1. (a) Numbers and their corresponding directoins; (b) a simulated crack line ........... 28
Figure 3.2. Four different scenarios and their corresponding node determination .................. 29
Figure 3.3. Node determination for 5x5 grid ................................................................................. 30
Figure 3.4. Continuation of node determination for 5x5 grid ....................................................... 30
Figure 4.1. Comparison of different Italian craquelure patterns ................................................... 36
Figure 4.2. Comparison of directionality histograms for (a) Flemish craquelure pattern; (b) French craquelure pattern................................................................. 38
Figure 4.3. Comparison of different rectangular structuring elements and their effects on line angles; (a) Original image; (b) 10x1 SE; (b) 15x1 SE; (c) 20x1 SE................................................................. 43
Figure 4.4. (a) Original French craquelure image; (b) vertically filtered image using mathematical morphology; (c) horizontally filtered image using mathematical morphology; (d) final result after taking the minimum of the last two images ........................................................................................................................................ 46
Figure 4.5. Thresholded and cleaned craquelure image of Figure 4.4........................................ 47
Figure 4.6. Scatter degree scores for each feature ........................................................................ 49
Figure 5.1. Classification accuracy over a range of features and k values ................................. 54
Figure 5.2. Staged kNN ............................................................................................................... 56
Figure 5.3. Classification accuracy of LDA classifier over a range of n features ..................... 62
Figure 5.4. Classification accuracy of QDA classifier over a range of n features .................... 65
Chapter 1

INTRODUCTION

Craquelure represents the crack formations that develop on the surface of a painting with age. Connoisseurs have long regarded craquelure as a significant feature of old paintings and although claims have been made that the patterns can assist in the task of attribution they were faced with a couple problems namely; the description of the phenomenon and its process of formation [1]. Dr. Bucklow, a research scientist at the Hamilton Kerr Institute and a world renowned expert in craquelure, set out to define a set of descriptions that would be sufficient in providing a framework which would enable judgements to be made about the materials and methods of construction of a painting [2]. Based off of his research, the pattern of cracks can be treated as a visible record of the physical tensions within the structure of a painting. The ways in which tensions are generated and dissipated are dependent on the choice of materials and methods of construction employed by the artist. Therefore, craquelure is related to the technical traditional in which a particular artist worked. In his previous works [1,3], it was shown that craquelure has potential as a non-destructive means of identifying the structural components of a painting and may possess possible applications in determining the geographical origins of a painting. In one of his studies, he was able to correctly classify a collection of craquelure images into their corresponding regions with an average algorithmic classification rate of 82.5%. Such information led to the emergence of other researchers trying to automate the classification process using computer image processing techniques in the hopes of alleviating the need of expertise in the area of craquelure. These works can be found in [4,5]. Such automated algorithms can be used by the inexperienced or novice connoisseurs in the task attribution.
While the previous works in developing an algorithm for the analysis and classification of craquelure have established a framework for any future works, the results themselves were not comparable to the ones achieved in Dr. Bucklow’s work [3]. This thesis will extend from the results of the previous works and establish a new standard in classifying craquelure images using a fully automated image processing algorithm. The thesis will also cover the techniques used during the crack extraction, crack network representation, feature extraction, features selection and classification phase.

1.1 Structure of Paintings

The purpose of a painting is to present us, the viewers, an image that can either represent things, ideas, or events that are familiar or wildly distant from our own experiences. Paintings are often regarded as two dimensional images commonly hung on walls. The fact that paintings have a material presence is often overlooked or is not fully appreciated. Physically speaking, a painting is a multi-layered structure that’s consists of a support, a glue layer, one or more layers of ground and one or more layers of paint. Craquelure in paintings is the result of the differences in mechanical behaviour of these various layers [6]. The structural components of a painting that determine the type of cracks that become visible on the surface are the support (wood or canvas), the glue sizing, the ground layers and the paint film [1]. The structures of a painting and their corresponding effects on craquelure, when applicable, will be discussed in the following sections.

1.1.1 Support

The back pane and foundation of a painting is known as the support. The support may be a flexible material like cotton or linen canvas stretched over a wooden frame or rigid panels of wood, metal, glass, or plastic. The stretcher is used in canvas supports for stability [7]. The main type supports we deal in this thesis are canvas and wood supports.
Panels, a name for wooden supports, can come in a wide variety however artists mainly use the woods from their region. For example, artists in Portugal used oak or Spanish chestnut, in Italy poplar is used, in Spain pine or poplar and in France oak, walnut or popular. An important aspect of wood is its orthotropic properties in the response to environmental changes. Due to the orthotropic behaviour, the crack patterns on panel painting usually form a lattice [7]. Dr. Bucklow found in his research that a property of early Italian poplar panels is that the crack run perpendicular to the grain of wood. However, a characteristic for early Flemish oak panels is that the cracks were found to run parallel to the grain of wood [8].

Canvas supports became increasingly more used in the beginning of the 16th century. The diversity of fineness of the fabric used as supports for paintings is quite large. The fineness is measured as the number of threads in the warp direction and the number of threads in the weft direction per square centimeter. There was a slight tendency to use a more coarse fabric in the 17th century [6]. In cases where ground layer is relatively thin, the crack lines usually run perpendicular to the longest dimension of the stretcher assuming that the warp is aligned with the longest dimension (which is often the case) [2].

1.1.2 Glue Sizing

On both panel and canvas paintings a sizing layer of animal glue is applied between the support and the ground layers. On panels the glue is used to create a stable and smooth underground for the ground and paint film and the glue also aids in binding the ground to the support. On canvas the glue is used to stiffen the support and to reduce the absorbency of the oil in the ground [6].

1.1.3 Ground

After the size has dried, the ground layer can then be applied. The ground is essentially paint, made of materials compatible with the support material and the paint to be used over it. A common ground that has traditionally been used over the centuries is gesso. Gesso is a mixture of
animal glue, chalk and at times a white pigment. A thin or brittle ground layer permits cracks to follow the grain of the wood or weave of the canvas. If the ground is thicker or less brittle, the cracks on the surface will essentially be decoupled from the support layer which leads to crack patterns largely determined by the mechanical layers of the ground layer and not the support [8]. For canvas, this means that smooth and curved cracks are associated with thick grounds while jagged cracks with a rectangular pattern are associated with think brittle grounds. For panel supports, it was found that some types of grounds, such as gypsum, lead to the formation of jagged cracks while chalk grounds, found in early Flemish panel paintings, lead to relatively smooth cracks.

1.1.4 Paint Film

Paint is primarily made out of two basic components namely; pigment and binding medium (binder). The component we perceive as colour is known as the pigment which is usually a fine powder or organic or inorganic material. Different kinds of pigments have been shown to have different effects on the mechanical characteristics of paint and on the way in which cracks develop. Paintings with multiple pigments have crack patterns that are radically different in areas that contain different pigments [6]. It is this reason why it was necessary in this thesis to investigate cracks that have developed in a common pigment specifically lead white paint.

The pigment is dispersed in a liquid called the binder which ensures that the coloured material remains in the place where it was originally applied. The binder can be a mixture of oils, eggs, gum or a synthetic polymer (acrylic, alkyd) [7]. Dr. Bucklow concludes in his work that the binding medium is probably one of the less influential factors in the way cracks develop in paintings [8].
1.2 Research Motivation

Dr. Bucklow’s findings suggest that craquelure can be used as a non-invasive which compliments other methods of technical examination of paintings. His studies demonstrated that variations in craquelure provide a reliable insight into the material nature of the painting. In doing so, it indicates the potential of a simple description to clarify the documentation of paintings, facilitate the dissemination of knowledge about paintings and highlight new avenues of research [2]. Previous works have attempted to build an automated system that is able to analyze craquelure from digital images and differentiate each of the craquelure patterns into their corresponding classes namely: Flemish, French, Italian and Dutch [4,5]. Their findings have shown that an automated system can indeed be used in the attribution of craquelure patterns. However, the results lack the precision that is needed for practical use and as such lack of accuracy has motivated the development of a more robust and reliable classification system which will be discussed in this thesis.

1.2.1 Objectives

The task of developing an analytical tool for the analysis of craquelure covers several different challenges. The first objective is to develop an efficient crack extraction tool that is able to solely isolate the craquelure found on the surface of a painting from a digital image. This step is crucial for it prevents the detection of false cracks and therefore providing more accurate results. Any misrepresentation in the results of the extracted cracks will deter the accuracy of the next steps. Furthermore, the second objective is to structure the craquelure patterns and extract the necessary features that will be able to differentiate between the four different classes. The strength of the features extracted will dictate the accuracy of classification. Lastly, a suitable classifier must be picked to perform classification. The objective is to choose a classifier that minimizes classification error while being robust.
Chapter 2

CRAQUELURE EXTRACTION

This chapter is concerned with the extraction of crack lines from greyscale images. The starting point is an 8-bit greyscale image and the final result is expected to be a binary image of cracks. It is crucial that the amount of false-positive crack detections is minimized before proceeding any further. As previously mentioned, the craquelure images this report deals with are on lead white backgrounds. The patterns in which crack patterns develop are highly dependent on the type of pigment as mentioned in Dr. Bucklow’s work [8]. Crack lines on lead white paintings naturally appear dark or black due to the contrast from the background. It is assumed that all the craquelure patterns that are dealt with in this report are of a much darker shade relative to the surrounding foreground colour. This assumption, being a reasonable one, allows us to perform mathematical morphology which will be discussed in the next section of this chapter.

The primary technique used in this section is one that consists of morphological operations to detect and extract crack lines. Common edge detectors, such as Sobel or Canny edge detectors, are not used due to the fact that the edges of crack lines are not of interest but rather the focus is on extracting the entire crack lines themselves. These types of edge detectors locate edges in areas where pixel values undergo a sharp variation [9]. Thus using an edge detector will yield in an image with lines that trace the edge of a crack line; an undesirable technique. Figure 2.1 demonstrates the major differences in using edge detectors instead of mathematical morphology for the extraction of crack lines.
Mathematical Morphology is a powerful tool in the realm of image analysis that interests many scientists from many different areas. In fact, mathematical morphology has been used in the study of arrangements of petrographic phases, of milling of rocks, of histology of the brain, of the dynamics of cloud movements, of computer reading of handwriting, etc. [10]. In the interest of being concise, this thesis will describe the mathematical morphological methods that have possible applications in the analysis of craquelure images. More detailed information on mathematical morphology can be found in [10].
Two fundamental operations, erosion and dilation, are used in mathematical morphology as building blocks for more complex operations. Before going into the details, a few definitions must be made. \( X \) is defined as a two-dimensional (2-D) image who range is \([N_{\text{min}}, N_{\text{max}}]\) and \( B \) is defined as a 2-D structuring element that represents the set of the neighbourhoods about the origin. The erosion operation is defined as:

\[
\varepsilon_B(S) = \bigcap_{b \in B} X_b = X \ominus \bar{B}
\]

where \( \bar{B} = \bigcup_{b \in B} \{ -b \} \) is the transposed set of \( B \). For a dilation operation, the converse is defined to be:

\[
\delta_B(S) = \bigcup_{b \in B} X_b = X \oplus \bar{B}
\]

The Structuring Element (SE) \( B \) can be of any shape or size depending on the application and the intended outcome of the procedure. Figure 2.2 illustrates a few of the structuring elements that are used throughout this thesis. Notice that each structuring element is of binary form.

More complex operations such as a closing or opening operation can be taken advantage of depending on the order Equations (2.1) and (2.2) are used. The opening and closing operations can be analytically represented as:

\[
\gamma_B(S) = \delta_B(\varepsilon_B(S)) = (X \ominus \bar{B}) \oplus B
\]

\[
\phi_B(S) = \varepsilon_B(\delta_B(S)) = (X \oplus \bar{B}) \ominus B
\]
where Equation (2.3) and Equation (2.4) represent the opening and closing operation respectively. An example of an opening and closing operation is shown in Figure 2.3. These operations used a disk SE with radius 1.

![Figure 2.3. (a) Original image; (b) Opening operation; (c) Closing operation](image)

As evident in Figure 2.3, the opening operation is successful in removing high-intensity shapes that have a width and/or length smaller to that of the structuring element. Conversely, the closing operation is effective in moving low-intensity shapes that are smaller than the structuring element in width and/or length. In regards to a craquelure image, the crack lines are almost always shown as think black lines. The crack lines can be removed by choosing a disk structuring element with a diameter of rough one and a half times the width of the average crack line and performing a closing operation. This is shown in Figure 2.4.
The image in Figure 2.4(b) has been morphologically closed in order to eliminate the vast majority of cracks. However, the goal of using mathematical morphology is to arrive at an image displaying only the cracks. This is achieved further processing the closed image using a black Top-Hat (TH) transform. The expressions for a white and black TH transform are:

\[
TH_B^w(S) = S - \gamma_B(S) \tag{2.5}
\]
\[
TH_B^b(S) = S - \phi_B(S) \tag{2.6}
\]

where Equations (2.5) and (2.6) are the white TH transform and black TH transform respectively. The different between the two transforms is that one seeks to isolate shapes that are smaller than SE and brighter than the background colour (white) while the other seeks to isolate shapes that are smaller than the SE but darker than the background (black). In the case of isolation crack lines from an image, the black TH transform is to be used. Using the closed image in Figure 2.4(b) and Equation (2.6), the result is illustrated in Figure 2.5.
The image in Figure 2.5 is a greyscale image that represents the majority of the crack lines found from the original image. However, a grayscale image is difficult to work with when trying to structure the crack networks and furthermore a grayscale image will undoubtedly contain false-positive crack detections; the majority of which can be removed by thresholding. The next section will cover the fundamental principles of thresholding and highlight some of the techniques used in this thesis.

2.2 Thresholding Techniques

After extracting the grayscale image using mathematical morphology, the next step is to convert the grayscale image into a binary image. This can be achieved by thresholding. The reason why obtaining a binary image is of interest is two folds. Firstly, noisy pixels are usually
eliminated while retaining just the crack lines. Although not all of the noise is filtered when thresholding, each pixel in the binary image can be regarded as a true crack line. Secondly, a binary image is easier to deal with when trying to structure the image into a list of crack networks. Using a grayscale-image when structuring the crack network would result in a lot of confusion when trying to decide if the pixels at hand contribute to a larger network of cracks. A binary image makes it a lot simpler to keep track of each crack line and to analyze the final crack network when trying to extract features.

The main principle of thresholding is to choose a threshold value, $k$, such that every pixel with a grayscale value above $k$ is set to ‘1’ and the remaining pixels are set to ‘0’. Choosing an optimal threshold value is an important step when trying to filter unwanted pixels. A threshold value that is too low will yield an image with an excess amount of noise. On the other hand, choosing a threshold value that is too high will result in an image that has lost important information. Dr. Otsu developed an automated method for determining a suitable threshold value that will retain the wanted information while discarding the majority of noise [11]. His method selects an optimal threshold value by using a discriminant criterion to maximize the resultant classes in gray levels. When using just a single threshold value and dealing with an image with $L$ gray levels $[1, 2, \ldots, L]$, the resultant classes are defined as $C_0$ and $C_1$ where $C_0$ denotes pixels with levels $[1, \ldots, k]$ and $C_1$ denotes pixels with levels $[k + 1, \ldots, L]$. If denote the number of pixels at level $i$ by $n_i$ and the total number of pixels by $N = n_1 + n_2 + \cdots + n_L$, the probabilities of class occurrence and the class mean levels, respectively, are given by

$$\omega_0 = \Pr(C_0) = \sum_{i=1}^{k} p_i = \omega(k)$$

$$\omega_1 = \Pr(C_1) = \sum_{i=k+1}^{L} p_i = 1 - \omega(k)$$

and
\[
\mu_0 = \sum_{i=1}^{k} i \frac{Pr(i|C_0)}{Pr(C_0)} = \frac{\sum_{i=1}^{k} ip_i}{\omega_0} = \mu(k)/\omega(k) \\
(2.9)
\]

\[
\mu_1 = \sum_{i=1}^{L} i \frac{Pr(i|C_1)}{Pr(C_1)} = \frac{\sum_{i=k+1}^{L} ip_i}{\omega_1} = \frac{\mu_T - \mu(k)}{1 - \omega(k)} \\
(2.10)
\]

where

\[p_i = n_i/N\]

and

\[\mu_T = \mu(L) = \sum_{i=1}^{L} ip_i\]

The class variances are given by

\[
\sigma_0^2 = \sum_{i=1}^{k} (i - u_0)^2 \frac{Pr(i|C_0)}{\omega_0} = \sum_{i=1}^{k} \frac{(i - \mu_0)^2 p_i}{\omega_0} \\
(2.11)
\]

\[
\sigma_1^2 = \sum_{i=k+1}^{L} (i - u_1)^2 \frac{Pr(i|C_1)}{\omega_1} = \sum_{i=k+1}^{L} \frac{(i - \mu_1)^2 p_i}{\omega_1} \\
(2.12)
\]

To evaluate the effectiveness or ‘goodness’ of threshold value \(k\), the following discriminant criterion was used in the discriminant analysis [11]:

\[\eta = \sigma_B^2 / \sigma_T^2\]

where

\[\sigma_B^2 = \omega_0 (\mu_0 - \mu_T)^2 + \omega_1 (\mu_1 - \mu_T)^2 = \omega_0 \omega_1 (\mu_1 - \mu_0)^2\]

and

\[\sigma_T^2 = \sum_{i=1}^{L} (i - u_T)^2 p_i\]

are the between-class variance and the total variance of levels respectively. The problem is then reduced to an optimization problem to search for a threshold value \(k\) that maximizes \(\eta\). This means that the \(k\) value that best separates the classes in gray levels would be chosen as the best threshold value.

Figure 2.6 will demonstrate a simple thresholding procedure using Otsu’s method.
Figure 2.6. (a) Original French craquelure image; (b) After using top-hat transformation; (c) Thresholding image using Otsu’s method for single k=0.28 value
Figure 2.7 displays the value of $\eta$ over the wide range of $k$ values. Notice how the $k$ value was selected to be the maximum value. A maximum value will always be found and the proof can be found in [11].

![Figure 2.7. $\eta$ values for a range of $k$ for image in Figure 2.6. Maximum occurs at $k=0.28$](image)

The next few subsections will cover the three different thresholding techniques that were tested name; global, segmented and offset thresholding.

### 2.2.2 Global Thresholding

Global thresholding refers to thresholding a whole grayscale image by using just one threshold value. This is an easy and fast way to threshold an image without being computationally expensive relative to any alternative techniques. As previously shown, Figure 2.6 displayed a grayscale craquelure image that has been globally thresholding using a constant $k$ value. For craquelure images that have a uniform background colour and consistent crack line colour, this techniques appeared to be suitable and fast. However, problems in this method became apparent when attempting to threshold an image that contained inconsistent background colour and/or crack lines with varying contrast levels. Figure 2.8 demonstrates this problem by using a particular Dutch craquelure image.
As evident in the Figure 2.8(b), in regions where the crack lines are harder to distinguish or low in contrast relative to the background, a fair amount of noise is present. It appears that using a single $k$ value when thresholding an entire image will yield acceptable results in some areas and inaccurate results in other areas.

2.2.3 Segmented Thresholding

A possible remedy for reducing the inaccuracies when using the global thresholding method is to segment the image into several smaller regions and then threshold each area using a newly calculated $k$ value. For the sake of simplicity, this method will be termed segmented thresholding and such a method has been used in [4,5]. Figure 2.9 will show how the craquelure image is segmented, the $k$ values for each region displayed as a greyscale value, and the final outcome of the method.
Figure 2.9. (a) Original image with segmented regions; (b) k values for each region; (c) final result
The results above suggest that segmented thresholding yields more accurate results compared to the global thresholding method. For the most part this was found to be true. However, several cases have been encountered where the results of the segmented method were less accurate. These cases usually involve images that have a large amount of variance in background colour and density of craquelure network. In areas where the crack density is low, the segmented area will attempt to ‘force’ cracks to appear when calculating the corresponding $k$ value. This is demonstrated in Figure 2.10.

![Figure 2.10. (a) Original Dutch image; (b) k values for each region; (c) final result](image)

As evident in the figure above, specific regions with relatively low crack density exhibit false crack detections due to the fact that a relatively low value of $k$ was chosen. These false crack detections pose a problem when attempting to structure the crack network. An alternative method of thresholding is presented in the next section.

### 2.2.4 Offset Thresholding

The offset thresholding method is intended to minimize the amount of false-positive crack detections. Similar to the segmented thresholding method, the grayscale image is segmented into several different regions however these regions are much smaller. A region’s width/length should be one and a half times larger but no larger than twice the average crack width. This ensures that
that each segment will cover a portion of a crack line while ignoring global variances in background colour by concentrating on relative contrast differences between the crack line and local background. Once the image has been segmented, a duplicate image is then created and segmented but this time each segmented region is offset horizontally and vertically by one half the width/length of the segmented regions. This offset takes care of boundary conditions that occur at the edge of cracks. The offset also ensures that the detected cracks are indeed true cracks by choosing the similar pixels between the original and duplicate are kept while discarding ‘noisy’ pixels. This is based off of the premise that true crack lines will be properly thresholded while noisy regions will inconsistently be thresholded. The similarities between both images are then saved and the differences are discarded.

Due to the relatively small size of the segmented regions in the offset method, undoubtedly there will be a high amount of regions that do not cover any sort of cracks. As previously mentioned earlier, the Otsu method will attempt to determine a value of $k$ that will in the end separate two classes of greyscale values resulting in the appearance of pixels in the absence of cracks. However, due to the absence of cracks, the calculated $k$ value will be lower than those that are calculated in the crack-filled regions thus forcing cracks to appear in the absence of any. This problem is solved by choosing a minimum $k$ value threshold. Any $k$ values that are lower than the minimum threshold value will be regarded as noise and the corresponding regions will not be threshold, or in other words, the $k$ value will be set to 255. Figure X will demonstrate visually the results of the offset method works and the difference between using a minimum $k$ value threshold as opposed to not using one.
Figure 2.11. (a) Original Italian craquelure image; (b) Offset threshold image without minimum k threshold; (c) offset thresholded image with minimum k threshold
As evident in the figure above, choosing a minimum threshold value for $k$ prevents the appearance of excess noise. This is done for both the original and offset image. Figure X illustrates the entire offset thresholding process.
Figure 2.12. (a) Original Dutch Craquelure Image; (b) Segmented image showing k values. Right image is offseted; (c) images after thresholding (d) final image after multiplying the two previous images together
2.2.5 Comparison of Thresholding Methods

The subsections above covered the three different thresholding methods that have been attempted and tested in this thesis. While each one has their own benefits, the offset method was deemed to be the most accurate technique although it is significantly more computationally expensive than the rest. However the offset method on average takes less than a minute to compute the threshold of a grayscale image. Such a time duration is more than acceptable for the goal of this thesis. Figure 2.13 compares the different thresholding methods by using the same Dutch craquelure image used before.
Figure 2.13. (a) Thresholded using global thresholding technique; (b) using segmented thresholding technique; (c) using offset thresholding technique; (d) original image
2.3 Enhancing and Cleaning Crack Network

While the offset thresholding method is effective in reducing the amount of noise in the image, the method itself does not eliminate all the noise. Depending on the image at hand, a fair amount of noise can still be filtered however this is at the expense of accidently removing true crack lines. One aspect of enhancing the thresholded image is to connect any disconnected lines that are separated with a gap of 1-5 pixels. This is achieved by using the Equation (2.2), the dilation operation, to expand all the crack lines in the image. Any gaps with a maximum size of 5 pixels are closed. Figure X depicts this process. The image is then thinned using the algorithm found in [12]. At this stage, each crack line is 1 pixel in width. However, the main purpose behind cleaning the image is to preserve all the crack lines while removing the noise. Scattered throughout the image are isolated and paired pixels. These pixels are assumed to be noise from the background of the painting; usually caused by dark blemishes found on the background. The isolated and paired pixels are removed by repeatedly filtering the image using the following filter:

\[
H = \begin{bmatrix}
-1 & -1 & -1 \\
-1 & 10 & -1 \\
-1 & -1 & -1
\end{bmatrix}
\]  

(2.13)

The filter in Equation (2.13) is convoluted with the thresholded image such as the one found in Figure Xc. The spatial convolution is represented by the following equation:

\[
G(x,y) = I(x, y) \ast H(x, y) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} I(n_1, n_2) \cdot H(x - n_1, y - n_2)
\]  

(2.14)

Using Equation (2.14), the filtered image \(G(x,y)\) will contain a range of numbers specifically \([-8,10]\). An isolated pixel is considered to be a pixel with no surrounding pixels within the 3-by-3 neighbourhood. Figure 2.14 shows an example of an isolated pixel.
It is clear to see that such an isolated pixel when filtered with filter shown in Equation (2.13) will result in a value of 10 at that location on the image. In order to remove all isolated pixels, each pixel with a corresponding $G(x,y)$ value of 10 is deleted. Removing paired pixels however is a slightly more complicated task. The end of a crack line will consist of a pixel connected to just one other pixel. Essentially these end points, later to be called edge points, will have a corresponding $G(x,y)$ value of 9. A pair of pixels could then be detected and deleted using the following algorithm:

*Algorithm:*

1) $G(x,y) = I(x,y) \ast H(x,y)$
2) $I_2(x,y) = \begin{cases} 1 & G(x,y) = 9 \\ 0 & otherwise \end{cases}$
3) $G_2(x,y) = I_2(x,y) \ast H(x,y)$
4) $I(x,y) = \begin{cases} 0 & G_2(x,y) = 9 \\ I(x,y) & otherwise \end{cases}$

The algorithm above first filters the thresholded image with filter $H(x,y)$ (Step 1). The ends of each crack line (edge points) are then detected and saved into $I_2(x,y)$ (Step 2). $I_2(x,y)$ is then filtered with $H(x,y)$ in order to detect any paired edge points (Step 3). Paired pixels are considered to be paired edge points. An edge point is paired with another edge point if the value of $G_2(x,y)$ is 9. The pixels that have a $G_2(x,y)$ value of 9 are removed (Step 4).
Chapter 3

HIERARCHICAL REPRESENTATION OF CRAQUELURE

3.1 Hierarchical Structure

After the cracks have been extracted and displayed as binary images, creating a model to represent the craquelure patterns is the next step. From there, features of the craquelure can easily be extracted.

The cracks are organized and stored as edges, nodes, and lines that make up multiple crack networks. Information about the crack’s length, location, and exact movement is also stored.

3.1.1 Chain Codes

Every crack line is stored and represented as a chain code. A chain code is a series of numbers that keep track of a crack line’s movement. Each number represents the direction in which the next line pixel is located. The figure below shows an example of a crack line and its chain code.

![Chain Code Diagram](image)

**Figure 3.1.** (a) Numbers and their corresponding directions; (b) a simulated crack line

In Figure 3.1(b), the chain code is represented by the following string starting from the bottom left corner: 1558225578. Many different features can be extracted from the chain code.
such as the lines general direction and length. Each edge or node will store a chain code along with. This will be discussed in Chapter 4.

3.1.2 Node Determination

A is node is a point where multiple lines intersect. When the tracking algorithm comes across a point with multiple neighbours, either that point or one of its neighbours is a node. Closer inspection is needed to determine exact where. The only time where one of the neighbours may be a node is if three neighbours form a straight line.

Only Figure 3.2(c) has three neighbours forming a line and therefore needs further inspection. Using a 5x5 mask instead of a 3x3, the next step is to count the number of neighbours in the directions of each of the three neighbours from the middle pixel. If the sum in each direction is two, the middle pixel is assigned the node. If the sums are all equal to one instead, the middle pixel of the three original neighbors is assigned the node. This is illustrated in Figure 3.3.
If neither of the two cases were met, a last set of rules is applied. Using the sums from the previous step, if a single maximum exists from one of the directions, the node is the next pixel in that direction from the middle. If no single maximum exists, the node is the next pixel in the direction of its current path. This is demonstrated in Figure 3.4.

3.1.3 Summary of Structure

A craquelure imagine is organized and stored in multiple crack networks. A crack network is a series of interconnected crack lines. The crack lines are made up of edges and node lines.
where a node line is a crack line connecting two nodes. Each edge and node line has stored information about its chain code, starting and ending coordinate, and length.

### Table 3.1. Craquelure structure and corresponding description

<table>
<thead>
<tr>
<th>Objects</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crack Network</td>
<td>• Array of nodes</td>
</tr>
<tr>
<td></td>
<td>• Array of edges</td>
</tr>
<tr>
<td></td>
<td>• Number of Islands</td>
</tr>
<tr>
<td>Node</td>
<td>• One or more chain codes array(s)</td>
</tr>
<tr>
<td></td>
<td>• Location on image ((x, y))</td>
</tr>
<tr>
<td></td>
<td>• Location of neighbouring edges/nodes ((x_1, y_1) \ldots (x_n, y_n))</td>
</tr>
<tr>
<td></td>
<td>• Length of line(s)</td>
</tr>
<tr>
<td>Edge</td>
<td>• One chain code array</td>
</tr>
<tr>
<td></td>
<td>• Location on image ((x, y))</td>
</tr>
<tr>
<td></td>
<td>• Location of neighbouring edges/nodes ((x_1, y_1) \ldots (x_n, y_n))</td>
</tr>
<tr>
<td></td>
<td>• Length of line</td>
</tr>
</tbody>
</table>

### 3.2 Crack Tracking Algorithm

When the cracks have been extracted and represented as a binary image, it is duplicated. The crack tracking algorithm uses the duplicated image because it clears every pixel it tracks over. The algorithm first scans the duplicated image looking for a crack pixel with only one neighbour. This pixel would represent the end of an edge and would be the starting point for the algorithm to trace.

When tracing a crack line, the algorithm first checks if the current pixel has any surrounding neighbours. If only one neighbour exists, the direction of the neighbour is added to the chain code. The algorithm moves over to the neighbour pixel while the other pixel is deleted. If zero neighbours exist, the current crack line the algorithm is tracing is an edge and the current pixel is the end point. An edge structure is created and its chain code, starting and ending coordinate, and length is stored. If the current pixel has multiple neighbours, it may be a node point. Using the node determination method in 3.1.2, the exact position of the node is located.
However, the node may actually be a false node. This can happen if some of the neighbours surrounding the node do not lead to a valid line since it may be too short and does not connect to another node. The algorithm traces each of the neighbour pixels and returns how many lead to a valid line. If the result is zero, the node is actually the end of an edge and all information such as the chain code is stored in an edge structure. If only one of the neighbours lead to a valid line, the node is not actually created since it does not branch to multiple lines. It is instead merged with the other line. If two or more neighbours lead to a valid line, the node is valid and information about the node’s chain code, starting and ending coordinate, and length is stored. Whenever reaching a node, the algorithm continues to branch in each direction until the whole crack network is traced.

After a crack network has been fully organized and stored, the algorithm finds another end of an edge to start tracing a separate crack network. The algorithm deletes every pixel it traces over so there is no possibility of tracing the same crack network twice.

3.3 Crack Network Pruning

When the crack tracking algorithm is running, it may find and store small crack networks. These small crack networks are usually unwanted noise. A threshold is created where all crack networks with a total length of less than 50 pixels are deleted.
When the crack networks have their features extracted in the next step, it is important that the noise has been reduced as much as possible.
Chapter 4

FEATURE EXTRACTION AND CLASSIFICATION

A crack network is a collection of nodes, edges, and lines. Such a database is meaningless when trying to classify a particular craquelure image unless features are extracted from the information collected. A feature represents a scalar property of an object which in this case is a crack network. A feature may indicate the amount of orderings or the amount of crack line density that is found in a craquelure image. These metrics are then used in the classification process. Features are created with the intention of using them to discriminate between the different classes of craquelure patterns. A good feature is said to be one that has a high degree of variance between classes and a low variance within classes. This chapter will cover all the features that are extracted in this thesis and the techniques used to test the strength of each feature.

4.1 Descriptions of Craquelure and Corresponding Features

Dr. Bucklow described crack patterns with a list of descriptions in his previous works [2,3]. Such descriptions were used to build a framework in identifying cracks and classifying them to their particular geographical regions. The descriptions were defined by Dr. Bucklow to be:

1. Predominant direction and orientation of cracks
   - No Direction or direction; isotropy or anisotropy?
   - If anisotropic, then parallel or perpendicular to grain?
2. Changes in direction of cracks
   - Locally- smooth or jagged
   - Globally – straight or curved
3. Relationship between crack directions
   - Paint islands – square or not square; is there an orthogonal relationship
4. Distance between cracks
- Spatial frequency – are the paint islands small or large?

5. Thickness of cracks
   - Are all cracks of uniform thickness or are secondary cracks present?

6. Junctions or terminations of cracks
   - Is crack network connected or broken?

7. Organization of cracks
   - Is crack network ordered or random?

Using the descriptions above, Dr. Bucklow was able to summarize each of the 4 traditions (Flemish, French, Italian and Dutch) accordingly as [2]:

**Italian fourteenth/fifteenth-century paintings on panel**
- Direction: Often have a predominant direction
- Orientation: Usually oriented perpendicular to grain
- Islands: Usually small to medium sized islands
- Smoothness: Usually jagged cracks

**Flemish fifteenth/sixteenth-century paintings on panel**
- Direction: Nearly always have a predominant direction
- Orientation: Usually oriented parallel to grain
- Islands: Usually very small islands
- Smoothness: Usually smooth
- Straightness: Usually straight
- Order: Usually ordered

**Dutch seventeenth-century paintings on canvas**
- Direction: Usually have a predominant direction
- Orientation: Usually oriented perpendicular to longest side
- Islands: Usually medium sized islands
- Smoothness: Usually jagged

**French eighteenth-century paintings on canvas**
- Direction: Usually have no predominant direction
- Orientation: Not applicable
- Islands: Usually large islands
- Smoothness: Usually smooth
- Straightness: Usually curved

Note how the descriptions for each of the four different traditions are described with terms such as “usually”, “often”, etc. This means that each Flemish craquelure image won’t necessarily look identical. This is more obvious when looking at some of the Italian craquelure images. Figure 4.1 illustrates how Italian craquelure can potentially have a significant variance between another. [talk more about it]
The descriptions above were proven to be sufficient in describing and identifying the craquelure patterns. In fact, fewer than half of them proved to be necessary for a very high level of discrimination between categories [2]. Such information is crucial when deciding what features are to be extracted due to the fact that some of the descriptions are difficult to implement in software. For instance, determining if the crack network is ordered is relatively easy to implement in software as opposed to determining the shape of an island (square or not square). This thesis attempted to extract all possible features that are related to the descriptions listed by Dr. Bucklow however some descriptions, such as the thickness of cracks, are not considered due to the complexity of software implementation.

4.1.1 Histogram Based Features

The histograms are created by tallying up the chain codes according in terms of directions. As previously mentioned in Chapter 3, a chain code is a sequence of numbers that represent
consecutive changes in directions. The numbers 1 and 7 represent movement in the vertical direction, 3 and 5 represent movement in the horizontal direction, 2 and 6 represent movement in the top-right diagonal direction and the remaining 0 and 8 codes represent movement in the bottom-right diagonal direction. Therefore we can represent features \( d_0, d_1, d_2, d_3 \) as:

\[
\begin{align*}
    d_0 &= \frac{1}{L_c} \sum_{i=1}^{n_c} \sum_{j=1}^{l_c} C_i^{1,7}(j) \\
    d_1 &= \frac{1}{L_c} \sum_{i=1}^{n_c} \sum_{j=1}^{l_c} C_i^{3,5}(j) \\
    d_2 &= \frac{1}{L_c} \sum_{i=1}^{n_c} \sum_{j=1}^{l_c} C_i^{2,6}(j) \\
    d_3 &= \frac{1}{L_c} \sum_{i=1}^{n_c} \sum_{j=1}^{l_c} C_i^{0,8}(j)
\end{align*}
\]  

(4.1)  

(4.2)  

(4.3)  

(4.4)

where

\[
C_{i}^{x,y}(j) = \begin{cases} 
1 & \text{if } C_i(j) = x \text{ or } C_i(j) = y \\ 
0 & \text{otherwise}
\end{cases}
\]

\( L_c = \text{total length of all chain codes} \)   
\( n_c = \text{number of chain codes} \)   
\( l_c = \text{length of chain code} \)

Looking at the equations above, it is clear to see that the features \( d_0, ..., d_3 \) are a normalized set of histogram features that represent the general directionality of the image. Figure 4.2 demonstrates the differences between the histogram of a Flemish craquelure pattern and a French craquelure pattern.
Figure 4.2. Comparison of directionality histograms for (a) Flemish craquelure pattern; (b) French craquelure pattern
Apply filters to the histogram allows for the creation of features that are able to gauge different aspects of a craquelure pattern in terms of directionality. For instance, the following feature is a rectangular histogram filter:

\[ h_0 = d_0 + d_1 \]  

(4.5)

A high \( h_0 \) value represents a rectangular like craquelure pattern. Another feature is a circle histogram filter:

\[ h_1 = \sum_{i=0}^{3} |d_i - 0.25| \]  

(4.6)

A low \( h_1 \) value represents a circular like craquelure pattern. Ideally, a circle would be \( h_1 = 0 \).

A more complicated and reliable feature extracted is one that measures the strength of the predominant direction, if any, in a craquelure image and correlates it with the orientation of the support. As mentioned in Chapter 1, the support’s orientation is based on the direction of the grain for panels or along the longest dimension of the stretcher for canvas. This feature is important for it is able to distinguish between the different traditions based off of the observations made by Dr. Bucklow. The orientation feature is calculated using the histogram features by:

\[
h_2 = \begin{cases} 
\frac{d_0 - d_1}{d_2 + d_3} & \text{if orientation of the support is vertical} \\
-\frac{(d_0 - d_1)}{d_2 + d_3} & \text{if orientation of the support is horizontal}
\end{cases}
\]  

(4.7)

The feature \( h_2 \) indicates whether the predominant direction is parallel or perpendicular to the orientation of the support. A positive value means that the prominent crack direction is parallel to the orientation of the support while a negative value means that it is perpendicular to the orientation of the support. The magnitude of \( h_2 \) indicates the strength of the prominent direction. A value of 0 would mean that the craquelure pattern does not have a prominent direction; a common attribute found in most French craquelure patterns.
4.1.2 Statistical Based Features

The statistical based features take advantage of the information found in chain codes and the number of edge/nodes relative to image scale. Calculating the length of line is done by:

\[
l_i = \sum_{j=1}^{l_c} \left( C^o_i(j) + \sqrt{2} C^e_i(j) \right)
\]

where

\[
C^o_i(j) = \begin{cases} 
1 & \text{if } C_i(j) \text{ is odd} \\
0 & \text{otherwise}
\end{cases}
\]

\[
C^e_i(j) = \begin{cases} 
1 & \text{if } C_i(j) \text{ is even} \\
0 & \text{otherwise}
\end{cases}
\]

\[l_c = \text{length of chain code}\]

Likewise, the total length and the average length of all the cracks in a craquelure image can be estimated by:

\[
l_T = \sum_{i=1}^{n_c} l_i
\]

\[
l_{avg} = \frac{1}{n_c} \sum_{i=1}^{n_c} l_i
\]

where

\[n_c = \text{number of chain codes}\]

The total length of cracks given in Equation (4.9) is in pixels. A key feature that is statistically extracted is the crack density of an image in relation to the image’s scale. The total crack density of a craquelure pattern (on an image) is calculated as:

\[
\rho_T = l_T \cdot \frac{cm}{pixel}
\]

Similarly, the island density, edge density and node density of a craquelure image is calculated respectively by:

\[
\rho_i = n_i \cdot \frac{cm}{pixel}
\]

\[
\rho_e = n_e \cdot \frac{cm}{pixel}
\]

\[
\rho_n = n_n \cdot \frac{cm}{pixel}
\]
where

\[ n_i = \text{number of islands} \]
\[ n_e = \text{number of edges} \]
\[ n_n = \text{number of nodes} \]

Other features extracted include the average length of an edge-to-node line and the average node-to-node length. These features indicate how widely-spaced the image is in respect with node and edge points. The average edge-to-node and node-to-node length is calculated by:

\[ l_{e-n} = \frac{1}{n_e} \sum_{i=1}^{n_e} l_i \cdot \frac{cm}{pixel} \tag{4.15} \]
\[ l_{n-n} = \frac{1}{n_n} \sum_{i=1}^{n_n} l_i \cdot \frac{cm}{pixel} \tag{4.16} \]

Features based on the ratios of lines per node and edges to nodes are calculated respectively by:

\[ r_0 = \frac{n_n^e}{n_n} \tag{4.17} \]
\[ r_1 = \frac{n_n^e}{n_n} \tag{4.18} \]

Finally, two features that are able to gauge the straightness and smoothness of a line are extracted. The feature that measures the straightness of a line is calculated by:

\[ s_0 = \frac{1}{n_e} \sum_{i=1}^{n_e} \frac{\| (x_i^1, y_i^1) - (x_i^2, y_i^2) \|}{l_i} \tag{4.19} \]

where

\[ (x_i^1, y_i^1) = \text{starting coordinates of the line} \]
\[ (x_i^2, y_i^2) = \text{final coordinate of } i\text{th line} \]
\[ \| \| = \text{magnitude function} \]

A straight line will have a \( s_0 \) value of nearly 0 while a jagged or curved line will have a higher value. This feature however is unable to determine whether the line is jagged or smooth. The feature that measures the jaggedness of a line is calculated by counting the number of times abrupt changes of equal to or more than 90 degrees in direction have occurred and diving that by the length of the line. This feature is denoted by the symbol \( s_1 \).
4.1.3 Global Features

Similar to $h_2$, a global feature is extracted that is able to more accurately determine the orderliness of an image using mathematical morphology instead of using directional histograms. In this method, the original craquelure image is filtered using mathematical morphology but instead of a circular structuring element as what is originally done, a rectangular structuring element is used instead. When performing a closing operation using a rectangular structuring element with one of the dimensions being a pixel long/wide, only the cracks that are oriented along the longest dimension of the structuring element are kept and the rest are removed. The longer the rectangle is, the stricter the orientation filtering becomes. For example, using a $10 \times 1$ pixel rectangular structure element will keep the lines that are oriented anywhere between -15 to +15 degrees relative to the horizontal axis. A $20 \times 1$ pixel rectangular however will only keep the lines that are angled between -5 to +5 degrees for example. These numbers are not exact and were only used to illustrate the point that the size of the structuring element plays a big role when filtering for orientation. Figure 4.3 shows the effects of a closing operation using different structuring element size.
Figure 4.3. Comparison of different rectangular structuring elements and their effects on line angles; (a) Original image; (b) 10x1 SE; (b) 15x1 SE; (c) 20x1 SE
In Figure 4.3, the lines are consecutively rotated by 5 degrees meaning that the first line is oriented vertically at 0 degrees, the second line at 5 degrees, and the third line at 10 degrees and so on. The structuring elements are also varied from $10 \times 1$ pixel to $20 \times 1$ pixel. As evident in the figure, a longer rectangular structuring element is able to more effectively filter rotated lines. Theoretically, an infinitely long rectangular structuring element will only keep the lines that are perfectly vertical (or horizontal). One thing to note is that the widths of the lines also affect the outcome. The ‘thicker’ the lines, the less likely they are to be filtered. Such a consequence is of great importance for it inherently prioritizes the more distinct and obvious craquelure lines as opposed to keeping the thin and diminished crack lines. This means that the thicker crack lines are given more slack when filtering while the thinner, and less important, crack lines are filtered more strictly. Figure X will demonstrate the use of mathematical morphology using a rectangular structuring element on a craquelure image. The rectangular structuring elements were of size $15 \times 1$ (vertical) and $1 \times 15$ (horizontal) pixels. The first step is to create two separate filtered images; one filtered vertically and the other horizontally. The next step would be to take the minimum between both images.
After obtaining the final result in Figure 4.4(d), the craquelure is then extracted, thresholded and cleaned. The final result is a binary image shown in Figure 4.5.
The remaining pixels in Figure 4.5 are the ones that belong to the horizontal or vertical crack lines albeit a 5 degrees variance between the crack lines is still permissible. The pixels are then summed up and are compared to the actual length of the original crack network. Such a comparison is the basis of the global feature that is extracted which is given by:

\[
g_0 = \frac{\sum_{x,y \in T} T(x,y)}{l_T} \tag{4.20}
\]

where

\[T(x,y) = \text{thresholded image of only horizontal and vertical lines}\]

\[l_T = \text{the total length of all crack lines in pixels}\]

A high \(g_0\) value indicates a craquelure image that is ordered meaning that the vast majority of craquelure lines are oriented horizontally and or vertically. A low \(g_0\) value indicates not only that the craquelure image doesn’t have a predominant direction but also that the craquelure pattern is not ordered; a typical feature of French craquelure.
4.1.4 Summary of Features

The table below summarizes all of the features that are extracted in this thesis from each of the craquelure images.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_0$</td>
<td>Vertical direction of normalized histogram</td>
</tr>
<tr>
<td>$d_1$</td>
<td>Horizontal direction of normalized histogram</td>
</tr>
<tr>
<td>$d_2$</td>
<td>Top-right direction of normalized histogram</td>
</tr>
<tr>
<td>$d_3$</td>
<td>Bottom-right direction of normalized histogram</td>
</tr>
<tr>
<td>$h_0$</td>
<td>Rectangular based filtering of normalized histogram</td>
</tr>
<tr>
<td>$h_1$</td>
<td>Circular based filtering of normalized histogram</td>
</tr>
<tr>
<td>$h_2$</td>
<td>Orientation based filtering of normalized histogram</td>
</tr>
<tr>
<td>$\rho_T$</td>
<td>Total crack length density</td>
</tr>
<tr>
<td>$\rho_e$</td>
<td>Edge density</td>
</tr>
<tr>
<td>$\rho_n$</td>
<td>Node density</td>
</tr>
<tr>
<td>$\rho_l$</td>
<td>Island density</td>
</tr>
<tr>
<td>$l_{avg}$</td>
<td>Average length of crack lines</td>
</tr>
<tr>
<td>$l_{e-n}$</td>
<td>Average length of edge crack lines</td>
</tr>
<tr>
<td>$l_{n-n}$</td>
<td>Average distance between nodes</td>
</tr>
<tr>
<td>$r_0$</td>
<td>Lines per node ratio</td>
</tr>
<tr>
<td>$r_1$</td>
<td>Edges to nodes ratio</td>
</tr>
<tr>
<td>$s_0$</td>
<td>Straight line to actual line length ratio</td>
</tr>
<tr>
<td>$s_1$</td>
<td>Measure of jaggedness</td>
</tr>
<tr>
<td>$g_0$</td>
<td>Measure of orderliness using mathematical morphology</td>
</tr>
</tbody>
</table>

4.2 Feature Selection

Table 4.1

4.2.1 Scatter Degree

Scatter Degree is a feature selection method that assigns each feature a degree of scattering. The more separation between classes in a feature, the higher degree of scattering that feature is assigned. Features with higher scatter degrees are selected for classification. A scatter degree of a feature is computed by the following formula [13]:

$$SD(t_i) = \frac{S_B(i, i)}{S_W(i, i)}$$  \hspace{1cm} (4.21)

$S_W$ is the within-class scatter matrix, $S_B$ is the between-class scatter matrix, and $t_i$ is the $i$th feature. The matrixes are computed by the formulas:
\[
S_B = \sum_{i=1}^{c} n_i (m_i - M)(m_i - M)^T \tag{4.22}
\]
\[
S_W = \sum_{i=1}^{c} \sum_{j=1}^{n_i} (d_{ij} - m_i)(d_{ij} - m_i)^T \tag{4.23}
\]

where

\[M = \text{mean vector of all features}\]
\[m_i = \text{mean vector of features for class } i\]
\[d_{ij} = \text{document vector } j \text{ of class } i\]
\[n_i = \text{the number of documents in class } i\]

4.2.1.2 Implementation

The scatter degree method was used on the feature set extracted from the set craquelure images. Figure 4.6 displays the results below.

The overall crack density feature had the highest scatter degree while the grid (?) feature and average edge length feature were placed second and third respectively. With these findings, knowing which features to use in the classification process has become clearer.
4.2.2 Principal Component Analysis (PCA)

The idea behind PCA is to represent a given set of features in a new dimensional space by finding a set of orthonormal basis vectors that are orientated along the highest variance. This orthonormal basis is what makes up the new feature space. The purpose of using PCA is to reduce the dimensions of the original feature space. In a sense, the redundant information within the data set is discarded while preserving only the distinguishable features. Once the orthonormal basis of the new feature space is created from the training set, each test sample is then projected onto the new space before classification.

Given a training set \( \Gamma_1, \Gamma_2, \ldots, \Gamma_M \) where \( \Gamma_i \) represents the set of features listed in Table 4.1 for the \( i^{th} \) image, the data is first centered about the origin. This is done by calculating the average training sample of the training set and subtracting the average sample from each sample in the set. The average training sample is defined by \( \Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i \). Each of the samples in the training set are then subtracted from the average sample and is the result is represented as \( \Phi_i = \Gamma_i - \Psi \). The goal of PCA is to find an orthonormal set of vectors, \( u_n \), which best describes the distribution of the data. As it turns out, the vectors, \( u_k \), are eigenvectors of the covariance matrix [14]:

\[
C = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T
\]

where

\[
A = [\Phi_1, \Phi_2, \ldots, \Phi_M]
\]

The next step is to find the eigenvectors of matrix \( C \) which are denoted by \( u_k \). The eigenvectors are then sorted according to their eigenvalues such that the first eigenvector is the one with the highest eigenvalue. After obtaining the new orthogonal basis (eigenvectors), each sample in the training and testing set can be projected onto the new feature space by using this simple operation:
\[ \omega_k = u_k^T (\Gamma - \Psi) \quad k = 1, ..., M' \] (4.25)

where \( M' \) is the number of eigenvectors such that \( M' < \) total number of eigenvectors. In practice, a smaller \( M' \) is sufficient when attempting to classify the sample. The weights calculated in Equation (4.25) form a vector \( \Omega^T = [\omega_1, \omega_2, ..., \omega_{M'}] \) which represents the weighting factor of each of the eigenvectors respectively. These weights are now considered the new features.
Chapter 5

CLASSIFICATION

Classification is the last step in determining the origin of a painting. There are many different types of classifiers. Selecting the appropriate classifier is crucial for accurate and consistent results. Nearest Neighbour (kNN), Linear Discriminant Analysis (LDA), and Support Vector Machine (SVM) will be discussed in this chapter.

The images used for training and testing in this thesis were acquired from Dr. Bucklow under his permission. The images were taken using a standard 35mm lens and SLR camera.

5.1 Nearest Neighbour

The Nearest Neighbour method is perhaps the simplest of all classifiers. Simply store all of the training set with their label (origin) and calculate the distance of the testing example to each of the training. Keep \( k \) of the closest training examples, where \( k \geq 1 \) is a fixed integer, and the most common label is the classification for the testing example [15]. A tie between multiple classes may occur and there are a number of ways of handling it. A simple method is to keep reducing \( k \) by a value of one until a tie no longer exists [4]. You may also use a small odd integer for \( k \) such as \( k = 3 \) to reduce the chances of a tie [15].

5.1.1 Feature Normalization

When using kNN, the distances are calculated by using the values of the sample’s features. Some features could naturally have higher values which would give make it a bigger influence when calculating distances. To solve this problem, you would need to normalize the feature values. An effective method used was dividing each value by its feature’s average. By doing so, the new average of every feature would be the same, a value of one.
5.1.2 Weighted Nearest Neighbour

Weighted kNN is an extension of the nearest neighbour method where the actual distances between the training example and testing examples is taken into account. Closer distances are given a higher weight factor and have more influence. If you let \( d_1 \) equal the distance of the closest sample and let \( d_k \) equal the distance the furthest, the weight factor \( w_j \) of a sample with a distance \( d_j \) is

\[
w_j = \frac{d_k - d_j}{d_k - d_1}
\]

As you can see, the weight factor of the closest distance is one and zero for the furthest distance. An advantage of weighted kNN is that ties are much less likely.

5.1.2.2 Implementation

By using the scatter degree, the features were ranked in order of effectiveness. When a number \( n \) is selected as the number of features, the weighted kNN classifier would use the top \( n \) features to calculate the distances. To identify the best value for \( k \) and \( n \), all the different combinations of \( k \) and \( n \) were tested where the integers \( k \) and \( n \) were greater or equal to one, and less than or equal to 20. The overall accuracy of classification was recorded for each trail. Figure 5.1 displays the graphed results.
Figure 5.1. Classification accuracy over a range of features and k values

$k = 7$ and $n = 12$ had the highest overall accuracy of classification. The actual classification for each origin using those $k$ and $n$ values is shown below.

Table 5.1. Classification results of kNN for $k = 7$ and $n = 12$ using scatter degree

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flemish</td>
<td>$86.5%$</td>
<td>0%</td>
<td>13.6%</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>French</td>
<td>0%</td>
<td>$71.3%$</td>
<td>16.5%</td>
<td>12.2%</td>
<td></td>
</tr>
<tr>
<td>Italian</td>
<td>1.9%</td>
<td>7.1%</td>
<td>$85.5%$</td>
<td>5.6%</td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td>0%</td>
<td>0.4%</td>
<td>8.5%</td>
<td>$91.2%$</td>
<td></td>
</tr>
</tbody>
</table>

The overall accuracy is calculated to be

$$\frac{86.5 + 71.3 + 85.5 + 91.2}{4} = 83.6\%$$

When using PCA instead of scatter degree, $k = 5$ and $n = 19$ was found to give the best results.
Table 5.2. Classification results of kNN for $k = 5$ and $n = 19$ using PCA

<table>
<thead>
<tr>
<th>Actual</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flemish</td>
<td>87.3%</td>
<td>0.1%</td>
<td>12.6%</td>
<td>0%</td>
</tr>
<tr>
<td>French</td>
<td>0.2%</td>
<td>65.7%</td>
<td>16.7%</td>
<td>17.5%</td>
</tr>
<tr>
<td>Italian</td>
<td>2.9%</td>
<td>2.8%</td>
<td>86.3%</td>
<td>8.1%</td>
</tr>
<tr>
<td>Dutch</td>
<td>0%</td>
<td>2.9%</td>
<td>10.4%</td>
<td>86.7%</td>
</tr>
</tbody>
</table>

The overall accuracy is calculated to be

$$\frac{87.3 + 65.7 + 86.3 + 86.7}{4} = 81.5\%$$

The results of PCA fall slightly short compared to scatter degree when using kNN.

5.1.3 Staged kNN

A staged kNN is a method where a testing example is run through a few different kNN classifiers. Each kNN classifier or stage uses specific features and $k$ values that distinguish one class from the rest. The final stage uses specific features and $k$ values that distinguish the remaining classes or all the classes from each other. Figure 5.2 is an example of a stage kNN flowchart.
As you can see, the final stage still checks if the painting may be French or Flemish even though the previous two stages already checked for them. The reason for this is that the previous stages might not be able to classify all French or Flemish paintings. The French or Flemish paintings that do not get classified in the first two stages can still be accurately classified in the final classifier since it uses a different set of features and a different value for $k$. When creating a staged classifier, it is crucial that there are no or very few misclassifications in the stages prior to the final classifier.

### 5.1.3.1 Implementation

The features were first ranked by using a scatter degree. The best value of $k$ and the number of features $n$ are determined through testing identical to the previous weighted kNN implementation. The results were the same as Table 5.1 with $k = 7$ and $n = 12$ since the same feature set was used. The classification rate of each origin is recorded and will be referred to as
the base classification rate. As an example, the base classification rate for Flemish is 86.5%. The overall accuracy is also recorded and using the results Table 5.1, the overall accuracy is

$$\frac{86.5 + 71.3 + 85.5 + 91.2}{4} = 83.6\%$$

These values are important when creating a stage classifier. To create a stage in the classifier, you select a class such as Flemish and run all the different combinations of $k$ and $n$ in the weighted kNN classifier like before. This time, the classification rate and accuracy for that class is recorded. The accuracy is calculated by dividing the classification rate of a class by the total rate a sample gets classified as that class. As an example, the accuracy of French in Table 5.1 is

$$\frac{71.3}{0 + 71.3 + 7.1 + 0.4} = 90.5\%$$

If a value for $k$ and $n$ exists where the classification rate is higher than its base classification rate and the accuracy of that class is higher than the overall accuracy recorded before (83.6%), a stage classifier for that origin would be effective. Ideally, you want to select the value for $k$ and $n$ that generate the highest classification rate and accuracy. Those new values are used to create a weighted kNN classifier, or staged classifier, that only checks whether the testing example is of that origin or not. In our findings, two stage classifiers were possible for Flemish and French. Table 5.3 displays the details.

<table>
<thead>
<tr>
<th>Origin</th>
<th>k value</th>
<th>n value</th>
<th>Classification rate</th>
<th>Accuracy</th>
<th>Base classification rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>French</td>
<td>16</td>
<td>3</td>
<td>75.8%</td>
<td>96.8%</td>
<td>71.3%</td>
</tr>
<tr>
<td>Flemish</td>
<td>1</td>
<td>16</td>
<td>92.5%</td>
<td>100%</td>
<td>86.5%</td>
</tr>
</tbody>
</table>

The other class’s accuracies were lower than the overall and would result in a higher misclassification rate if a stage classifier were to be made for those classes. The overall staged classifier was constructed by placing the more accurate stage classifier first and the other second.
Then a final weighted kNN classifier was created to classify all the test samples that did not get classified in the previous two stages. Since the classification rates of the two stage classifiers were not 100%, the final weighted kNN may classify the testing example as any of the four origins. The values of $k$ and $n$ for the final classifiers are determined through testing. The two stage classifiers are used in those tests. The best value for $k$ and $n$ for the final classifier was discovered to be 7 and 12 respectively.

Here is an overview on how the staged classifier operates. Given a test sample, the classifier checks if the sample belongs to the French class using the first classifier. 75.8% of French testing examples are accurately classified and only 3.2% of non-French testing examples are misclassified as French in this step. If the example is classified as French, the classification process is complete. If not, the testing example is run through the second staged classifier checking if it belongs to a Flemish origin. 93.7% of Flemish testing examples are accurately classified and 0% of non-Flemish testing examples are misclassified as Flemish in this step. Again, if the testing example is classified as a Flemish in this step, the classification process is complete. If the test sample has not classified yet, the final weighted kNN classifier will classify it as any of the 4 origins. The results of using this staged classifier are shown in Table 5.4 below.

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flemish</td>
<td>92.2%</td>
<td>0%</td>
<td>7.8%</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>French</td>
<td>0%</td>
<td>83.5%</td>
<td>3.7%</td>
<td>12.9%</td>
<td></td>
</tr>
<tr>
<td>Italian</td>
<td>0.2%</td>
<td>7.4%</td>
<td>83.0%</td>
<td>9.6%</td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td>0%</td>
<td>0.7%</td>
<td>7.7%</td>
<td>91.7%</td>
<td></td>
</tr>
</tbody>
</table>

The overall accuracy is calculated to be

$$\frac{92.2 + 83.5 + 83.0 + 91.7}{4} = 87.6\%$$
As you can see, the staged classifier yields much better results than a single weighted kNN classifier. Note that most of the improvements are found in the Flemish and French origins since staged classifiers were constructed specifically for them.

The same method was applied again but PCA was used in place of the scatter degree. The results are displayed below.

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flemish</td>
<td></td>
<td>87.5%</td>
<td>0%</td>
<td>12.5%</td>
<td>0%</td>
</tr>
<tr>
<td>French</td>
<td></td>
<td>0.1%</td>
<td>70.1%</td>
<td>14.9%</td>
<td>14.9%</td>
</tr>
<tr>
<td>Italian</td>
<td></td>
<td>6.5%</td>
<td>7.1%</td>
<td>77.9%</td>
<td>8.6%</td>
</tr>
<tr>
<td>Dutch</td>
<td></td>
<td>0%</td>
<td>0.2%</td>
<td>3.4%</td>
<td>96.4%</td>
</tr>
</tbody>
</table>

The overall accuracy is calculated to be

\[
\frac{87.5 + 70.1 + 77.9 + 96.4}{4} = 83.0\%
\]

PCA’s results fall short again compared to scatter degree when using a staged kNN classifier.

5.2 Discriminant Analysis (DA) Classifier

5.2.1 Linear Discriminant Analysis (LDA) Classifier

Suppose the samples have multivariate normal densities with different means but the same covariance, the Probability Density Functions (PDF) of each sample can be described as [16]

\[
f_k(x|\mu_k, \Sigma) = \frac{1}{|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)\Sigma^{-1}(x-\mu_k)}
\]

where

\[
x = \text{samples} \\
u_k = \text{means for class } k \\
\Sigma = \text{covariance matrix}
\]

The class means, \(\mu_k\), can be calculated using

\[
\hat{\mu}_k = \frac{1}{N_k} \sum_{i \in k} x_i
\]

59
where

\[ N_k = \text{number of samples in class } k \]

Assuming the covariance’s are equal between all classes, the pooled covariance can be calculated as

\[ \hat{\Sigma} = \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in k} (x_i - \bar{\mu}_k)'(x_i - \bar{\mu}_k) \]  \hspace{1cm} (5.4)

where

\[ n = \text{number of total samples} \]

Equation (5.2) represents the likelihood of a sample belong to class \( k \). Obviously a sample \( x \) with a value of \( \mu_k \) would result in a maximum class probability. Essentially, a sample belongs to the class \( k \) which is the one with the highest \( f_k \). Thus for a given \( x \), we choose the \( k \) to maximize

\[ \pi_k e^{-\frac{1}{2}(x-\mu_k)\Sigma^{-1}(x-\mu_k)'} \]  \hspace{1cm} (5.5)

or by taking the logs, the \( k \) that maximizes

\[ -\frac{1}{2}(x - \mu_k)\Sigma^{-1}(x-\mu_k)' + \log(\pi_k) \]  \hspace{1cm} (5.6)

where

\[ \pi_k = \frac{N_k}{n} \]

If Equation (5.6) is expanded and only the terms that are dependent on \( k \) are kept, the result would be as follows

\[ d_k^*(x) = x\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k\Sigma^{-1}\mu_k' + \log(\pi_k) \]  \hspace{1cm} (5.7)

Equation (5.7) is referred to as the discriminant function [16]. Note that the function is linear in \( x \), hence the linear discriminant function. A sample is said to belong to the class that results in the highest \( d_k^* \) score for corresponding \( k \) class. Therefore we can define the classifier based upon Fisher’s linear discriminant function to be

\[ C_{LDA}(x) = k \text{ if } d_k(x) > d_l(x) \text{ for } l \neq k \]  \hspace{1cm} (5.8)
Looking at Equation (5.6) and Equation (5.7), the inverse of the covariance matrix $\Sigma$ is required for the calculations. However attempting to invert a potentially large matrix is not a good idea. One reason is that the inverse computation for a large matrix is expensive and often times difficult. Another reason is that the covariance matrix might be singular (non-invertible) and attempting to invert such a matrix can potentially lead to problems. An alternative and faster way of computing Equation (5.6) is to use QR decomposition [16] which is a decomposition of a matrix $A$ such that:

$$A = QR$$  \hspace{1cm} (5.9)

where

- $Q =$ an orthogonal matrix
- $R =$ triangular matrix

The QR decomposition is usually computed using the Gram-Schmidt process [17]. This decomposition is able to simply the calculation cost of Equation (5.6) by making use of a few matrix properties for orthogonal and invertible matrixes. Assuming that the covariance matrix is non-singular and can be represented in the form of Equation (5.4), the following representation can be made:

Let matrix $Q$ and $R$ be the decomposition of the following matrix such that:

$$\begin{bmatrix}
(x_i - \mu_k) \\
\vdots \\
(x_i - \mu_k)
\end{bmatrix} = QR$$  \hspace{1cm} (5.10)

where

$$i \in k$$

Substituting Equation (5.10) into Equation (5.4)

$$\Sigma = (QR)'(QR) = (R'Q')(QR) = R'R$$  \hspace{1cm} (5.11)

This is due to the fact that $Q'Q$ results in the identity matrix because $Q$ is said to be an orthogonal matrix. Substituting Equation (5.11) into Equation (5.6):

$$-\frac{1}{2}(x - \mu_k)(R'R)^{-1}(x - \mu_k)' + \log(\pi_k)$$  \hspace{1cm} (5.12)
\[ -\frac{1}{2} (x - \mu_k) R^{-1} (R^{-1})' (x - \mu_k) + \log(\pi_k) \]  
\[ -\frac{1}{2} (x - \mu_k) R^{-1} * ( (x - \mu_k) R^{-1} )' + \log(\pi_k) \]  

(5.13)  
(5.14)

As evident in Equation (5.14), the calculations in Equation (5.6) is greatly simplified by using QR decomposition. Computing the inverse of the upper triangle matrix, \( R \), is significantly easier than calculating the inverse of the covariance matrix \( \Sigma \). An efficient implementation of Equation (5.14) is to let \( B = (x - \mu_k) R^{-1} \) and then square every element in \( B \) before adding up the terms. Essentially for each testing sample, Equation (5.14) is used for each class \( k \) and the sample is said to belong to the class \( k \) that yields the highest score.

5.2.1.2 Results

The features were ranked by using a scatter degree. Like the nearest neighbour classifier, when a number \( n \) is selected as the number of features, the weighted LDA classifier would use the top \( n \) features. Through testing, letting \( n = 18 \) yielded the best results.

![Figure 5.3. Classification accuracy of LDA classifier over a range of n features](image-url)
Table 5.6. Classification accuracy for LDA classifier for n = 18 using scatter degree

<table>
<thead>
<tr>
<th>Class</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flemish</td>
<td>95.0%</td>
<td>0%</td>
<td>5.0%</td>
<td>0%</td>
</tr>
<tr>
<td>French</td>
<td>0%</td>
<td>75.0%</td>
<td>4.6%</td>
<td>20.4%</td>
</tr>
<tr>
<td>Italian</td>
<td>15.7%</td>
<td>4.7%</td>
<td>70.1%</td>
<td>9.5%</td>
</tr>
<tr>
<td>Dutch</td>
<td>0%</td>
<td>2.0%</td>
<td>6.9%</td>
<td>90.4%</td>
</tr>
</tbody>
</table>

The overall accuracy is calculated to be

\[
\frac{95.0 + 75.0 + 70.1 + 90.4}{4} = 82.6\%
\]

When using PCA instead of scatter degree, n = 18 also gave the highest overall accuracy.

The results are displayed below.

Table 5.7. Classification accuracy for LDA classifier for n = 18 using PCA

<table>
<thead>
<tr>
<th>Class</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flemish</td>
<td>93.6%</td>
<td>0%</td>
<td>6.4%</td>
<td>0%</td>
</tr>
<tr>
<td>French</td>
<td>0%</td>
<td>75.1%</td>
<td>4.9%</td>
<td>20.4%</td>
</tr>
<tr>
<td>Italian</td>
<td>16.7%</td>
<td>4.5%</td>
<td>69.1%</td>
<td>9.7%</td>
</tr>
<tr>
<td>Dutch</td>
<td>0%</td>
<td>1.9%</td>
<td>6.7%</td>
<td>91.4%</td>
</tr>
</tbody>
</table>

with an overall accuracy of

\[
\frac{93.6 + 75.1 + 69.1 + 91.4}{4} = 82.3\%
\]

PCA and scatter degree seem to delivery very similar results when using LDA.

5.2.2 Quadratic Discriminant Analysis (QDA) Classifier

The linear discriminant analysis (LDA) classifier assumes that the different classes all have the same covariance and hence a pooled covariance matrix is calculated. However if this is not the case, a stratified estimate of the covariance can be made for each \( k^{th} \) group. \( \Sigma_k \) is denoted as the sample covariance matrix for the \( k^{th} \) group. Similar the LDA classifier, the pdf for the samples is assumed to be
\[
f_k(x|\mu_k, \Sigma_k) = \frac{1}{|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k)'}
\]  
(5.15)

where

\[
\Sigma_k^{-1} = \frac{1}{N_k} \sum_{i \in k} x_i
\]

The discriminant functions can be taken as the terms of the exponent thus yielding

\[
\hat{d}_k^Q = -\frac{1}{2} (x - \mu_k)\Sigma_k^{-1}(x - \mu_k)'
\]
(5.16)

Expanding Equation (5.16) and keeping the terms that are dependent on \(k\) yields

\[
-\frac{1}{2} x\Sigma_k x' + x\Sigma_k \mu_k' - \frac{1}{2} \mu_k \Sigma_k \mu_k'
\]
(5.17)

As evident in Equation (5.17), the discriminant function depends not only linearly on \(x\) but also quadratically and hence the name Quadratic Discriminant Analysis (QDA) classifier.

Similar to the LDA classifier, the QR decomposition is utilized in Equation (5.16) to yield an efficient implementation of the discriminant function. For each testing sample \(x\), the discriminant score is calculated for each class \(k\) and the sample is said to belong to the \(k^{th}\) class that pertains to the highest score.

5.2.2.2 Results

Just like in the LDA implementation, a scatter degree is used to rank the features and every value of \(n\) is tested. Through testing, letting \(n = 16\) yielded the best results.
Figure 5.4. Classification accuracy of QDA classifier over a range of n features

Table 5.8. Classification accuracy for QDA classifier for n = 16 using scatter degree

<table>
<thead>
<tr>
<th>Actual</th>
<th>Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Flemish</td>
<td>Flemish 96.7%</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>French 0%</td>
</tr>
<tr>
<td></td>
<td>Italian</td>
<td>Italian 15.0%</td>
</tr>
<tr>
<td></td>
<td>Dutch</td>
<td>Dutch 0%</td>
</tr>
</tbody>
</table>

The overall accuracy is calculated to be

\[
\frac{96.7 + 87.0 + 68.6 + 85.7}{4} = 84.5\%
\]

As you can see, the Quadratic Discriminant Analysis (QDA) Classifier yields slightly higher overall accuracy than the Linear Discriminant Analysis (LDA) Classifier.

When using PCA instead of scatter degree, \( n = 18 \) gave the highest overall accuracy. The results are displayed below.
Table 5.9. Classification accuracy for QDA classifier for n = 16 using PCA

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted Class</th>
<th>Flemish</th>
<th>French</th>
<th>Italian</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flemish</td>
<td>92.7%</td>
<td>0%</td>
<td>7.3%</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>French</td>
<td>0%</td>
<td>84.1%</td>
<td>10.7%</td>
<td>5.2%</td>
<td></td>
</tr>
<tr>
<td>Italian</td>
<td>15.7%</td>
<td>11.2%</td>
<td>66.8%</td>
<td>6.3%</td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td>0%</td>
<td>8.1%</td>
<td>7.9%</td>
<td>84.0%</td>
<td></td>
</tr>
</tbody>
</table>

with an overall accuracy of

\[
\frac{92.7 + 84.1 + 66.8 + 84.0}{4} = 81.9\%
\]

It appears that PCA is not as effective as scatter degree when using a QDA classifier.
Chapter 6

CONCLUSIONS

Craquelure can be used as a non-destructive means of identifying the structural components of a painting as well as potentially indicating the geographical origin of an art painting. With the advances of computer imaging and technology and based off the results of this thesis, it is evident that an automated computer algorithm can accurately classify a particular art piece to its corresponding origin. Such an algorithm alleviates the need for expertise in the subject of craquelure and how craquelure pertains to the materials and methods employed by the artist.

Mathematical morphology has been proved to be a powerful tool when extracting craquelure from an image. The novel offset thresholding method proposed in this thesis provides a very efficient, robust and reliable way of thresholding a greyscale image by reducing the amount of false-positive crack detections and retaining true crack lines. Compared to the other thresholding techniques found in literature on the subject of craquelure extraction, the offset thresholding method has proven to be far superior and such a method can be used for various other applications.

The features extracted in this thesis were capable of effectively discriminating between the four different classes. The majority of the features relied on the information stored in the chain codes however the global feature that measured the amount of orderliness not only scored really high in the scatter degree calculations but also provided a novel way of detecting the grid like structure of a craquelure pattern. The feature extraction phase is arguably the most important phase in this thesis. The features extracted dictate how accurate the classification of our samples will be. Judging from the results obtained, the features extracted in this thesis were sufficient in
being able to provide a relatively high degree of accuracy especially when comparing with similar works.

The classifiers compared in this thesis were capable of classifying the craquelure images with an acceptable degree of accuracy. The kNN classifier however was shown to be the most accurate in terms of classification. Specifically, the stage kNN classifier yielded the best results. While not being superior to the staged kNN classifier, the LDA and QDA classifier are still acceptable in terms of their results. Using the staged kNN classifier, a classification rate of 87% was achieved. On the other hand, the LDA and classifier at best yielded a classification rate of 82% and 84% respectively. The scatter degree method was proven to be a suitable feature selection (dimension reduction) technique prior to classification. Although close in results, the scatter degree method was shown to achieve higher classification accuracy compared to using PCA.

This thesis was successful in developing an automated computer algorithm for the analysis and classification of craquelure for art paintings. Using some of the novel methods proposed in this research, we were able to achieve great results that suggest a potential in using such an algorithm in practice.

6.1 Future Work

The work of this thesis can be further continued in two different directions. Firstly, the classification accuracy can be increased by finding stronger features to extract and by experimenting with different classifiers. The results also suggested that there is room for improvement in terms of classification accuracy especially when looking at classification accuracy versus the number of features used figures. Secondly, a crack monitoring system can be established by using the framework developed in this thesis. Such a monitoring system can be used to periodically track and monitor the growth of cracks on the surface of a painting. By
locating areas of crack growth, an art connoisseur for example, can focus his or her effort on that area of the painting to prevent further damage. Such a monitoring system is in the making here at the University of Windsor.
Personal Publications and Presentations

Pending Publications:


International Presentations:


WORKS CITED


VITA AUCTORIS

NAME: 
Mouhanned El-Youssef

PLACE OF BIRTH: 
Riyadh, Saudi Arabia

YEAR OF BIRTH: 
1989

EDUCATION: 
Vincent Massey Secondary School, Windsor, ON, 2007
University of Windsor, B.Sc., Windsor, ON, 2011
University of Windsor, M.Sc., Windsor, ON, 2013