

Image Analysis of Microscope Slides for Palynofacies Studies

A dissertation presented

by

James John Charles

to

The Department of Computer Science
in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in the subject of

Mathematics & Computer Science

Bangor University

Bangor, Gwynedd, UK

June 2009

Declaration

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed :..... (James J. Charles)

Date : 08/06/09

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed :..... (James J. Charles)

Date : 08/06/09

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed :..... (James J. Charles)

Date : 08/06/09

Image Analysis of Microscope Slides for Palynofacies Studies

Abstract

Microfossil analysis is essential when identifying hydrocarbon resources for the petroleum industry. Such analysis is conducted by paleontologists who interpret the depositional environment by examining microscope slides containing samples of organic microfossils known as palynofacies. The main goal of this thesis is to develop the components of an image analysis system for automatic segmentation of palynofacies.

Microscope images of palynofacies contain three types of material: kerogen, palynomorphs and amorphous matter. These types have very different appearances and significance for the image interpretation by the domain experts. Kerogen is a type of organic microfossil that yields oil upon heating. Two kerogen types are usually presented in images of palynofacies: inertinite and vitrinite. The prevalence and the appearance of the two types carry important information about the environment. Kerogen pieces are the darkest objects in the image, highly irregular in shape, overlapping and touching. Distinguishing between the two kerogen types is not straightforward even for the trained paleontologists.

We propose a system for automatic classification of kerogen into vitrinite and inertinite using 5 image processing stages: image acquisition, background removal, microfossil segmentation, feature extraction and classification.

Background removal corrects for uneven lighting using multiple 1D parabolas. A marker-based segmentation method is proposed, called Centre Supported Segmentation (CSS) for identifying touching and overlapping objects in a binary image. Its only parameter expresses the acceptable degree of overlap in segmenting the individual objects. A measure of the segmentation quality is proposed to compare marker-based segmentation results.

An expert palaeontologist labelled kerogen objects as either vitrinite or inertinite, which provided the ground truth labels for training a classifier. A study comparing ten

state-of-the-art classifiers singled the logistic classifier out as the most accurate one for the task. We show that the classifier is stable with respect to the overlap parameter of CSS.

The palynomorph microfossils are deformed or folded around one another. They are presented in the image as semi-transparent, partly or entirely elliptical objects. We propose a scheme for classification of complete elliptic palynomorphs using the logistic classifier. After segmentation, two classes of objects are formed - “complete palynomorphs”, and “other”, containing the kerogen, amorphous matter and other unspecified debris. ROC curve analysis is used to select a certainty threshold for the classification.

The methods and solutions proposed in this study offer a toolbox for developing commercial systems for palynofacies classification.

Contents

Title Page	i
Declaration	ii
Abstract	iii
Table of Contents	v
Acknowledgments	viii
1 Research hypothesis	1
1.1 Chapter map	2
1.2 Contributions	2
2 Image analysis of palynofacies	4
2.1 Palynomorphs	5
2.2 Kerogen	7
2.3 Amorphous material	8
2.4 Image acquisition	9
2.4.1 Microscope Stepping Stage	10
2.5 Digital Images	11
2.5.1 Digital monochrome images	11
2.5.2 Digital colour images	12
2.6 Feature extraction	13
2.6.1 Colour features	13
2.6.2 Size features	15
2.6.3 Texture Features	17
2.6.4 Shape features	25
2.6.5 Summary of features	31
2.7 Stages of the image processing	31
Nomenclature 1	36
3 Background segmentation	37
3.1 Microscopy images	38
3.2 Segmentation methods based on thresholding the original image	40
3.2.1 Global thresholding	40
3.2.2 Adaptive thresholding	45

3.2.3	Clustering	53
3.3	Lighting correction methods	54
3.3.1	Homomorphic filtering	57
3.3.2	Estimating image background	57
3.4	Crossing Stripe Parabolas (CSP)	60
3.4.1	The method	62
3.4.2	Forming a background estimate	64
3.5	Evaluation and results	64
Nomenclature 2		68
4	Microfossil segmentation	69
4.1	Segmenting kerogen from the background	70
4.2	Segmentation of touching objects	70
4.2.1	Recent methods	72
4.2.2	Watershed segmentation	73
4.2.3	Marker controlled watershed	77
4.3	Centre supported segmentation (CSS)	79
4.3.1	Stage 1 of CSS - locating object centres	80
4.3.2	Stage 2 of CSS - removing redundant centres	85
4.3.3	Segmenting the objects using centres	88
4.4	Segmentation evaluation	90
4.4.1	Obtaining centres for a given segmentation	92
4.4.2	Centre-based measure	93
4.4.3	When can we apply this measure?	96
4.5	Experimental assessment	96
4.5.1	Kerogen object segmentation	96
4.5.2	CSS vs Extend h -maxima transform	98
4.6	Summary	100
Nomenclature 3		102
5	Kerogen classification	103
5.1	System overview	104
5.2	Classifiers	105
5.3	Training data	107
5.4	Cross-validation	108
5.5	Comparing classifiers	109
5.6	Logistic classifier	110
5.7	Classification experiment using all features	111
5.8	Feature selection	112
5.9	Further analysis	118
5.10	Classification stability	119
5.10.1	Hierarchical structure of centres	120
5.10.2	Label inheritance trees	123

5.11 Stability experiment	127
5.12 Summary	130
Nomenclature 4	131
6 Complete palynomorph recognition	132
6.1 Microfossil segmentation	135
6.2 Detecting complete palynomorphs	136
6.2.1 Classifiers	138
6.2.2 Feature extraction and selection	138
6.2.3 Training data	139
6.2.4 Class imbalance	139
6.2.5 Receiver-Operator characteristics (ROC) analysis	139
6.2.6 Classification using top 10 classifiers	141
6.3 Classification using the logistic classifier	141
6.4 Logistic classification examples	145
6.5 Summary	152
Nomenclature 5	153
Conclusion	154
7.1 Overview	154
7.2 Future work	156
7.3 Publications related to the thesis	158
Glossary	159
A Logistic Classifier - A Matab implementation	170

Acknowledgments

Although I am the sole author of this thesis, merit for its success should also be attributed to many others who provided me with the help, support and guidance necessary for its completion.

First and foremost I would like to express my gratitude to Dr Ludmila Kuncheva (Lucy) who was always there to provide advice, patiently supervising me and guiding me in the right direction. I will never forget the times we spent discussing segmentation and classification algorithms over a coffee at the local coffee shop. In fact, there was no problem unsolvable by Lucy if a coffee was at hand.

This project would not have taken place had it not been for my co-supervisor Dr Ik Soo Lim who won the EPSRC CASE grant (CASE/CNA/05/18), which I acknowledge with gratitude. I am indebted to Ik Soo for providing me with two 21" widescreen monitors - these greatly increased my productivity and were superb for watching the latest movie!

I would like to thank Dr Barrie Wells for coordinating the project and bringing me up to speed on relevant geological information. Many thanks to Dr Lisa Buckley who provided some useful information for the project and Chris Whitaker who was on hand for statistical advice. I am grateful to Dr Alan Collins and Dr Nick Miles for the pain staking task of manually labelling over 600 images for use in the classification system.

At times when a break from work was necessary, my office colleagues, Rhys Thomas, David Hughes and Catrin Plumpton were always there to provide amusement or put up with my crafting of office games. I will miss our debates and discussions on various mathematical and computing problems.

Last but by no means least, I would like to say a sincere thankyou to my parents and family for all their love, support, direction and encouragement which has kept me on track over the last three years. Special heartfelt thanks go to Layla, the light of my life and best friend for her forbearance and understanding during the past roller-coaster months

leading up to the completion of my thesis.

Chapter 1

Research hypothesis

Industrial sectors are continually developing old and new applications which pertain themselves to image analysis solutions. With industrial backing, image analysis techniques are constantly enhanced and cultivated in areas such as astronomy, medicine, defence, robotics, security, remote sensing and microscopy.

Analysing images of microscope slides containing assemblages of microfossils is one possible application within the area of microscopy. An assemblage of acid-resistant organic microfossils recovered from deep below the ground is collectively known as palynofacies. A domain expert would spend about four hours examining a single microscope slide containing palynofacies. Their task is varied but typically includes categorisation of microfossils into predefined classes along with investigation of microfossil characteristics such as size, colour and shape. Such an arduous job is both time consuming and critical to the development of oil and gas prone sites within the hydrocarbon industry.

Given the extremely subjective nature of palynofacies evaluation, together with the haphazard arrangement of microfossils on a slide, can an image analysis system be built to identify microfossils automatically? The research hypothesis of this work is that such a

system can be developed, and will result in accurate, robust and fast fossil identification. The intention of this study is to propose and implement part of an overall block-structure of such an automatic system, as detailed next. The new image analysis techniques developed in the process will be usable within other domains.

1.1 Chapter map

Presented in this thesis is a system for analysing kerogen and palynomorph microfossils on an image of a slide containing palynofacies. The “chapter map” is used to visualise each chapter of the thesis in accordance with the systems processes. Figure 1.1 displays the chapter map.

1.2 Contributions

This thesis claims the following contributions:

1. Background segmentation and background correction using Crossing Stripe Parabolas (CSP) method.
2. Centre Supported Segmentation (CSS) method and hierarchical tree representation used for stability assessment.
3. Segmentation evaluation measure
4. Application of classification methods to kerogen classification and complete elliptic palynomorph recognition
5. An overall system design for automatic palynofacies analysis

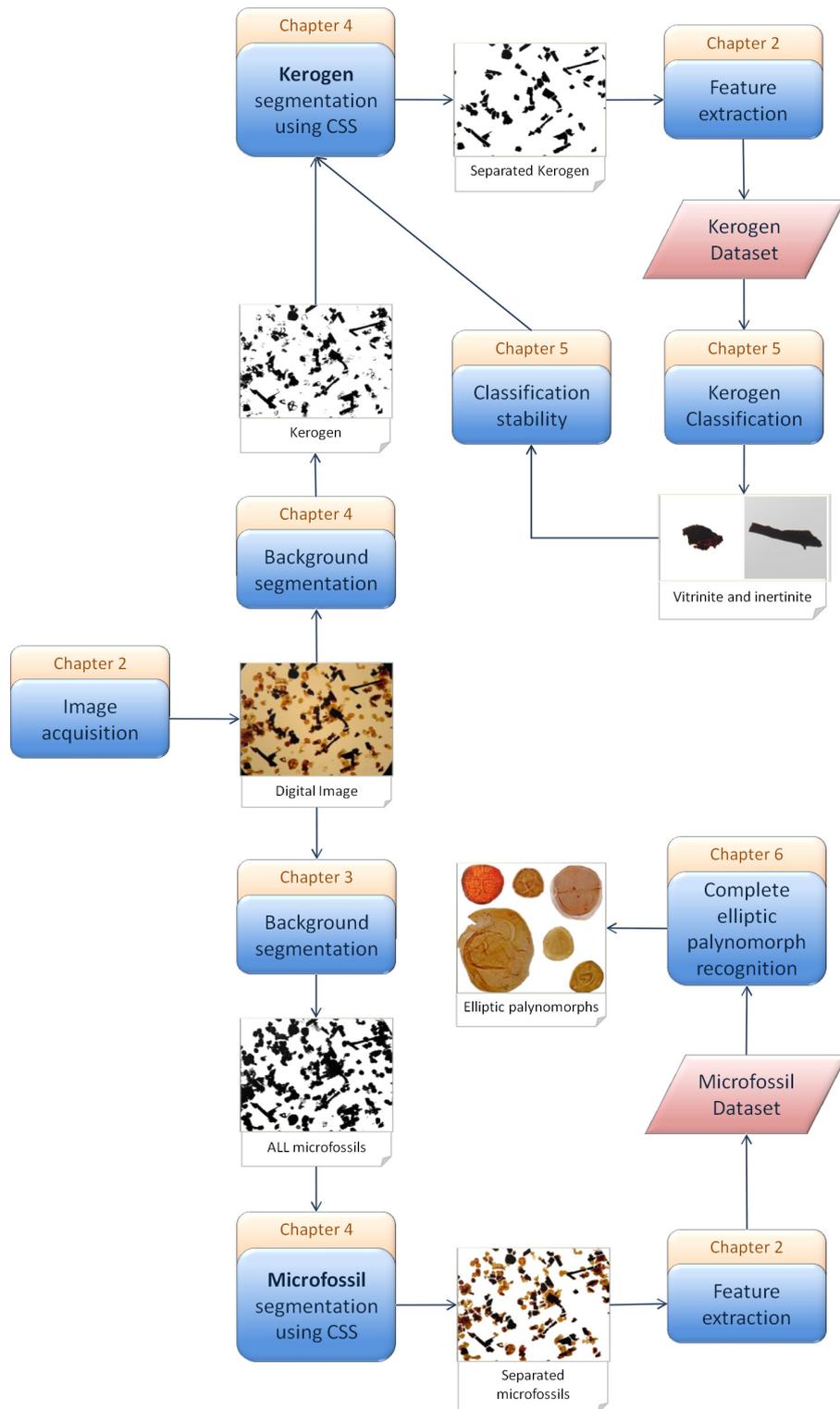


Figure 1.1: Chapter content in terms of system processes

Chapter 2

Image analysis of palynofacies

Microfossils are used extensively by the petroleum industry when exploring for oil. Hydrocarbon palaeontologists consider them to be one of their main tools as they provide extensive information on the depositional environment. The two most common applications of microfossils within the petroleum industry are biostratigraphy and paleoenvironmental analyses. The study of stratigraphy involves looking at rocky layers to determine the processes that created it. The law of superposition states that layers of fossil-bearing stone are stacked in a sequence according to the age of deposition. It was discovered that a specific sequence will be present within many different locations. One of the earliest geologists to discover this was William “Strata” Smith (Palmer, 2005). He found that rock layers in road cuts and quarries were stacked in the same way in different parts of England. This type of study can help form a correlation between various locations.

Biostratigraphy is a branch of stratigraphy that only uses fossil assemblages contained within the rock layers; the usual aim is to form a correlation between land sites. This type of stratigraphy is important because the age of rock layers can be calculated based upon the fossils they contain rather than the type of rock or sediment that makes

up the layer. Furthermore, palyoenviromental analysis is the interpretation of the type of environment that formed the rock, based upon the fossils it contains.

When drilling for oil a fluid lubricates the drill bit and helps flush small pieces of rock from the bottom of the drill hole, these small pieces are known as cuttings. Cuttings contain microfossils that are mostly undamaged by the drilling process due to their small size ($< 1\text{mm}$). A sample of cuttings contains an assemblage of organic microfossils known as *Palynofacies*, a term that was first introduced by a french geologist Combaz in 1964 (Combaz, 1964). Although this assemblage contains many different types of microfossil they can be broadly classified into three sub-groups including palynomorphs, kerogen and amorphous material. Palynomorphs are the organic microfossil representing some form of living matter such as a cell, spore or tissue. Kerogen is the woody, plant material and amorphous matter is formed by bacterial and chemical activities of decaying palynomorphs or kerogen.

2.1 Palynomorphs

Because palynomorphs are extremely resistant to most forms of decay other than oxidization, they are preserved when buried deep underground. They are around $5\mu\text{m}$ to $500\mu\text{m}$ in size and composed of sporopollenin, chitin or other related compounds.

The type of palynomorph found in rock are helpful for palyoenviromental analysis and source potential for hydrocarbons (Al-Ameri and Batten, 1997). Also the colour of the palynomorph can be a crucial indicator for hydrocarbon exploration. The organic walls of palynomorphs change colour with increasing burial temperature and can be used to interpret post-depositional geothermal gradients. Colour changes can be reproduced by heating experiments in the laboratory (Epstein et al., 1977) and it was found that these



Figure 2.1: Examples of palynomorphs all taken under the same microscope magnification.

colour changes are irreversible and can be assigned to different temperature regimes (Pross et al., 2007). Temperature plays a key part in hydrocarbon generation hence reconstructing the thermal history of sediments is a crucial task when exploring for petroleum. Example images of palynomorphs can be found in figure 2.1. Palynomorphs are pale brown to brown with a sharp distinct outline and maybe some internal structure.

2.2 Kerogen

Kerogen is an organic chemical compound found in sedimentary rock that releases oil or gas upon heating. We can classify kerogen into two types, inertinite and vitrinite. These compounds are both reflective, inertinite being the most reflective. The maximum temperature subjected to a sample can be determined by a measure of vitrinite/inertinite reflectance (Burmham and Sweeny, 1989), making this an invaluable property. Furthermore the shape of these pieces can be used to predict the distance from the sediment source (Hoelstad et al., 1994; Tyson and Follows, 2000; Buckley, 2004).

Inertinite is a black opaque fragment with sharp angular edges and usually lath-shaped. Very slight rounding can occur. These types of macerals are common in many samples but are most abundant in sandstone. Inertinite is derived from the tissues of higher plants and gelified amorphous material.

Vitrinite is a term introduced by Stopes (1935) to describe the maceral derived from woody tissues of roots, stems, barks and leaves composed of cellulose and lignin (International Committee for Coal and Organic Petrology (ICCP), 1998). During the decomposition process cell structures can be lost or preserved and are visible to varying extents. Vitrinite is dark brown to black in colour and appears as an angular or rounded grain.

There are only slight visual differences between inertinite and vitrinite. This is illustrated in figure 2.2, where vitrinite is grouped within a dashed boundary and inertinite within a solid boundary. Both types of microfossil can either be lath shaped or rounded; lath shaped pieces have been highlighted in grey. Notice that inertinite exhibits a sharp distinct outline with no internal structure whereas the internal structure of vitrinite is visible mainly nearer the periphery.

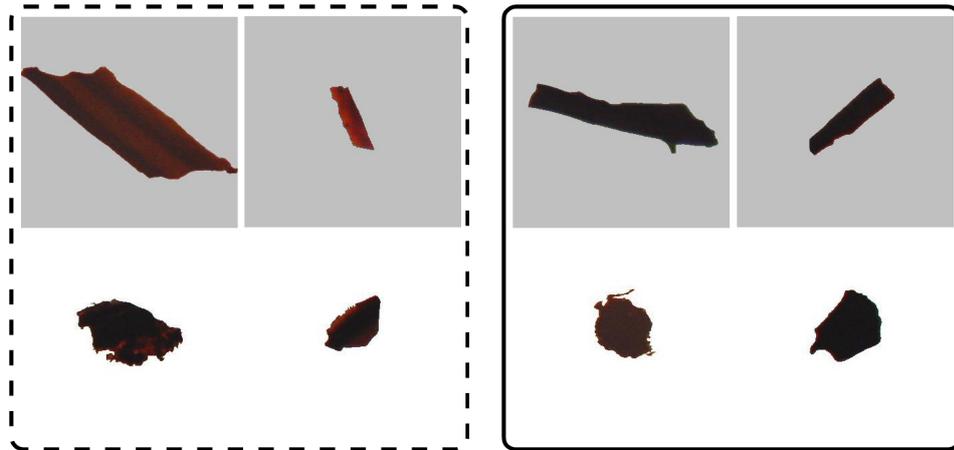


Figure 2.2: Vitrinite is indicated by a dashed boundary, inertinite by a solid boundary. Rounded kerogen is shown and lath shaped pieces are highlighted in grey.

2.3 Amorphous material

The term amorphous material is used to describe organic microfossils that do not possess a specific shape, structure or obvious outline. There are many possible origins of amorphous material but it is mainly derived from chemically and physically degraded plants, animal debris, structured kerogen or original structureless material. It is very difficult to identify the original source of amorphous organic matter both optically or chemically.

Amorphous organic microfossils (AOM) are pale to brown, grey or yellow and can be distinguished in four ways based upon textural differences (Thompson and Dembicki, 1986). Oil-prone samples are defined by two types of AOM. The first appear as chunky compact masses with mottled network or weak polygon textures; the second is thin, platy or has rectangular individual grains. Gas-prone samples contain two different types of AOM. The first is very small, dense, elongate, oval or rounded individual grains; the second has clumps with granular, fragmented or globular texture. Example images of extracted amorphous material can be seen in figure 2.3.

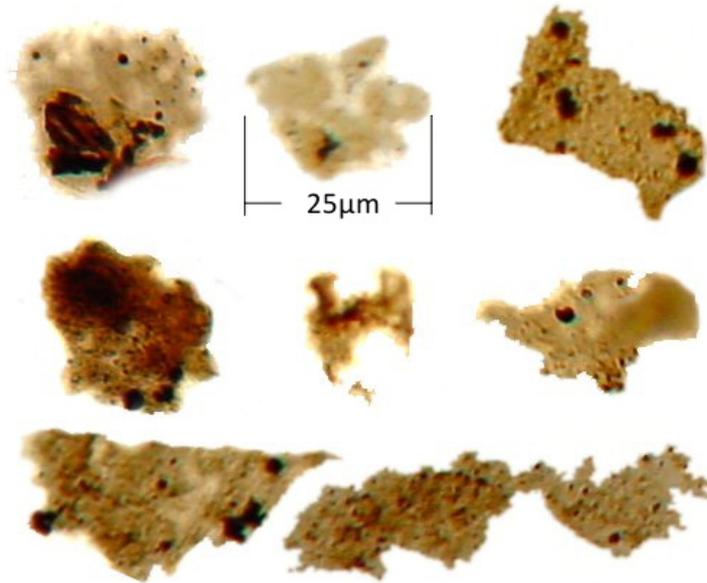


Figure 2.3: Examples of amorphous material all taken under the same microscope magnification.

2.4 Image acquisition

To form a microscope image containing palynofacies a sample of rock cuttings are taken, washed, sieved and then mounted on a microscope slide (a more detailed review of palynomorph preparation is provided by Riding and Kyffin-Hughes (2004)). These are known as dispersed preparation slides.

The image of a microscope slide is stored using the JPEG format. This format was used due to hardware restrictions. In subsequent stages we will be applying image analysis techniques to the image to perform segmentation and ultimately classification. The JPEG format is not a lossless compression method, hence the compression factor should remain fixed between images ensuring any measurements taken are consistent. Also the compression may cause minor changes to the microfossil boundaries and so most of the features we use to represent a microfossil such as size, shape and colour are chosen in order to ignore this

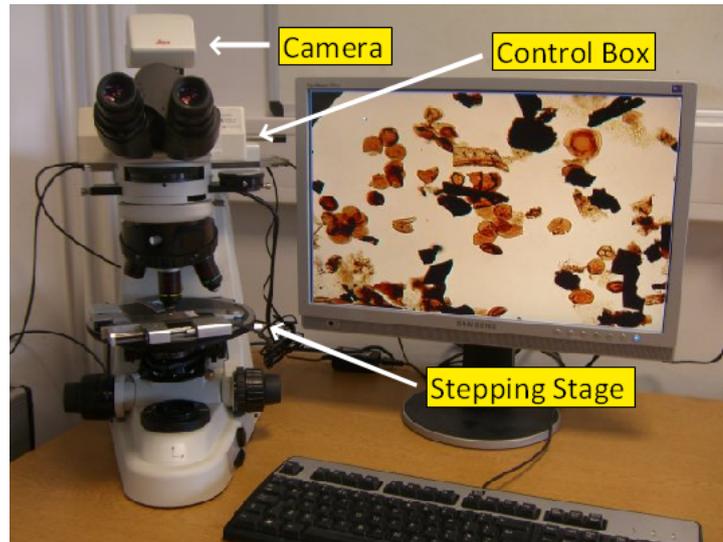


Figure 2.4: Illustration of microscope setup

artefact.

2.4.1 Microscope Stepping Stage

High resolution images (typically 1704 by 2272 pixels) are captured under transmitted light using a digital camera attached to a microscope. The microscope setup consists of an automatic stepping stage that can be attached to any microscope and is controlled through Petrog software (Wells, 2008). A digital firewire camera and control box are used to capture high resolution images and transfer them to a computer.

The stepping stage holds the slide with a spring loaded arm. It will automatically move the slide in a discrete lattice arrangement. This allows the digital camera to capture over 400 high resolution images covering the entire sample. An illustration of this setup is shown in figure 2.4.

2.5 Digital Images

A continuous grey scale image is defined as a two dimensional function $f(x, y)$. The spatial coordinates are represented by the values of x and y . The intensity of the image at coordinates (x, y) is the output f . A digital image is formed by sampling the function $f(x, y)$ at discrete values of x and y and through quantisation of the intensity f .

2.5.1 Digital monochrome images

For a monochrome image the intensity ranges between black and white. In a digital image it is common to define the intensities as integers where black is intensity 0 and white is intensity 255.

Subsequent to sampling and quantisation the digital image is represented as a matrix of integer values containing M rows and N columns; we say the digital image is of size $M \times N$. The elements of the matrix are known as pixels. For clarity the discrete coordinates x and y are integer valued and indicate row and column positions respectively. It is common to define the origin of the digital image as $(0, 0)$, however the image processing toolbox in Matlab defines the origin to be $(1, 1)$ and so we adopt this convention. For example the next pixel below the origin will be at $(2, 1)$ and the pixel in the last row and last column is at position (M, N) . It is important to note that the coordinate (x, y) in the digital image is the position of pixels and not the position of sampled points in the continuous image. The matrix representation of a digital image $f(x, y)$ is shown below:

$$f(x, y) = \begin{pmatrix} f(1, 1) & f(1, 2) & \dots & f(1, N) \\ f(2, 1) & f(2, 2) & \dots & f(2, N) \\ \vdots & \vdots & \ddots & \vdots \\ f(M, 1) & f(M, 2) & \dots & f(M, N) \end{pmatrix}$$

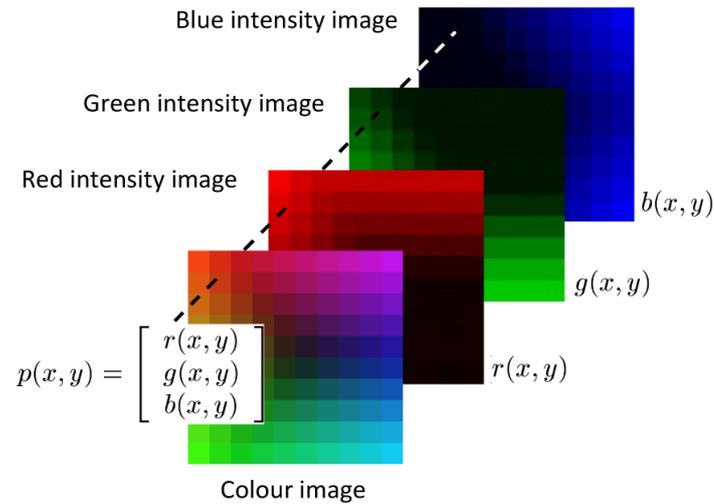


Figure 2.5: Digital colour image representation using the RGB colour system.

2.5.2 Digital colour images

Colour images are composed of individual monochrome images. The RGB colour system uses three 2D component images $r(x, y)$, $g(x, y)$ and $b(x, y)$ representing the red, green and blue values respectively. The colour image can be thought of as a $M \times N \times 3$ array where the red, green and blue component images are stacked along the 3rd dimension. An illustration of this is shown in figure 2.5. The colour of a pixel (x, y) is characterised by a three dimensional vector $[r(x, y), g(x, y), b(x, y)]$. When all three values are identical the pixel is displayed as grey.

The monochrome images used to construct the colour image define the total number of colours available. The red, green and blue component images can contain 2^b unique grey intensity values. Where b is the number of bits used to store a number on computer. For example an 8-bit image can contain 256 grey levels and a 16-bit image can contain 65536 grey levels. Therefore a colour image contains $(2^b)^3$ unique colours.

Other colour systems can be used to represent an image but, throughout this thesis we have adopted to use the RGB colour system as it is based on the human perception of

colours.

2.6 Feature extraction

Subsequent to image segmentation, object regions are identified. These regions will ideally represent individual objects (in our case microfossils) but can also contain non-objects. To further analyse these regions it is necessary to create a numerical representation usually in the form of a vector. Each element of the vector describes a different feature. A vector representation allows the application of classification algorithms and the extraction of important statistical information. A total of 32 numerical features are used accounting for shape, size, colour and texture. The majority of these features are general object features and are not specific to palynofacies. Similar studies (Weller et al., 2005) have used these features by extracting them using the image processing package Halcon¹. Here all features are extracted using Matlab.

2.6.1 Colour features

Colour features are more commonly processed on the RGB colour image. It was shown using the dichromatic reflection model proposed by (Shafer, 1985) that photometric phenomena such as shadows, specularities and illumination influence the red, green and blue values in an image. Because of this, colour feature detection algorithms have been developed so that they are invariant to these photometric affects (Gevers and Stokman, 2004; Zickler et al., 2008).

It was demonstrated by Gevers and Stokman (2004) that there exists a balance between the discriminative power of colour features and their invariance to photometric

¹Halcon is a software environment for image processing (MVTec Software GmbH, 2008).

changes. Therefore the chosen colour features should only be invariant to the various forms of lighting that can occur within the image of interest. For best discrimination, colour features should be chosen so that they are only invariant to uneven lighting in the image. A normalised colour can be obtained that is invariant to surface orientation, illumination direction, and illumination intensity. Suppose the colour of a pixel is (r, g, b) then it is proven (Gevers and Stokman, 2004) that the normalised colour is $(r, g, b)/(r + g + b)$. It is not necessary to use normalised colour provided the illumination across the microscope slide has been normalised beforehand. The colour features used are calculated from the normalised image; they comprise of the mean colour intensities.

Mean colour intensities

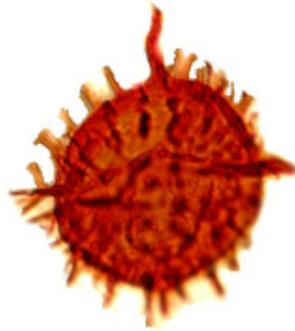
An extracted colour object is shown in figure 2.6. The object is a palynomorph, captured using 256 distinct intensity levels for the red, green and blue component images. Let O be the set of all object pixels. The component images from the colour image are used to find the features *mean red* \bar{r} , *mean green* \bar{g} and *mean blue* \bar{b} :

$$\bar{r} = \frac{1}{|O|} \sum_{\mathbf{p} \in O} r(\mathbf{p}), \quad \bar{g} = \frac{1}{|O|} \sum_{\mathbf{p} \in O} g(\mathbf{p}), \quad \bar{b} = \frac{1}{|O|} \sum_{\mathbf{p} \in O} b(\mathbf{p}).$$

The RGB colour image is converted to a grey level intensity image in Matlab using the function `rgb2gray()`. This function applies the following transform to the colour image

$$gr = 0.2989r + 0.5870g + 0.1140b$$

Where $gr(x, y)$ is the grey intensity image. The coefficients assigned to each colour channel were designed by Matlab to best describe human perception. The *mean grey* \bar{gr} feature is



$$\bar{r} = 174.13, \quad \bar{g} = 49.14, \quad \bar{b} = 19.12, \quad \bar{gr} = 83.06.$$

Figure 2.6: Extracted palynomorph showing colour features, \bar{r} , \bar{g} , \bar{b} and \bar{gr}

therefore:

$$\bar{gr} = \frac{1}{|O|} \sum_{\mathbf{p} \in O} gr(\mathbf{p}).$$

2.6.2 Size features

The image of an object is a projection of the object from 3D to a 2D plane. In this study we measure object size in 2D. The most basic size feature *area* is simply the total number of pixels comprising the entire object. Object contour, C , is important for describing shape as well as size. For example *perimeter* is the number of pixels in C and the *diameter* of an object is given by the largest distance between two pixels in C . Another feature, *distance*, is the mean distance from the centre of gravity of the object to all contour pixels in C . The set of object pixels O is used to calculate the centre of gravity \mathbf{c} :

$$\mathbf{c} = \frac{1}{|O|} \sum_{\mathbf{p} \in O} \mathbf{p}.$$

Then *distance* is

$$\delta = \frac{1}{|C|} \sum_{\mathbf{q} \in C} \|\mathbf{c} - \mathbf{q}\|,$$

where $\|\cdot\|$ is the Euclidean distance between two vectors. Hence δ is a scalar, invariant to object rotation and translation.

An alternative approach to measuring size is to bound the object in some way and measure the dimensions of the boundary. For instance, *inner radius* r_{in} is the radius of the largest circle completely contained within the object. The *Distance function* $D(\mathbf{p})$ of pixel $\mathbf{p} \in O$ is

$$D(\mathbf{p}) = \min_{\mathbf{q} \in C} \|\mathbf{p} - \mathbf{q}\|,$$

i.e. the minimum distance from each object pixel $\mathbf{p} \in O$ to a contour pixel $\mathbf{q} \in C$. Inner radius can be calculated as the maximum of all these distances:

$$r_{\text{in}} = \max_{\mathbf{p} \in O} D(\mathbf{p}).$$

The distance function can be computed on a binary image with the function `bwdist()` in Matlab. For each pixel in the binary image it will assign a number that is the distance between that pixel and the nearest nonzero pixel of the image.

The feature *outer radius* r_{out} is the radius of the smallest circle containing the whole object. An algorithm to calculate the smallest enclosing circle in linear-time is presented by Megiddo (1983). The size feature combining both inner and outer radii is *circle difference* and found as $r_{\text{out}} - r_{\text{in}}$.

Size features are illustrated in figure 2.7 on two kerogen microfossils. The kerogen piece on the left demonstrates where *diameter* is equal to the diameter of the smallest bounding circle. This is not always true, as shown in the image to the right. Such differences between size features can be used as shape features.

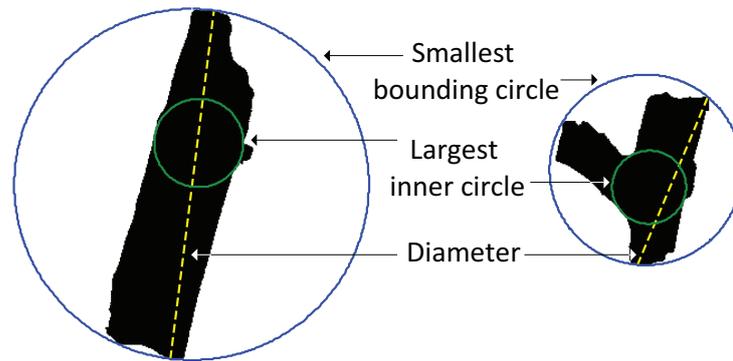


Figure 2.7: Kerogen microfossils displaying size features

2.6.3 Texture Features

All texture features are extracted from the grey-level image. An object's texture contains information regarding the structure of its surface and its relation to the surrounding environment. All objects contain some type of texture and it is easy for human observers to recognise and describe it using terms such as coarse, smooth, fine, irregular etc. The textural properties of an image hold useful information for discrimination purposes.

Entropy is a statistical measure of randomness that is used to characterise the texture content of an image. For example, an image that contains only one intensity value will have zero entropy. An image of rough rock which is highly textured will have high entropy. To calculate entropy, a histogram h_i of the relative frequencies of the grey values $gr(\mathbf{p})$ of all pixels $\mathbf{p} \in O$ is formed. The index $i = 0, \dots, (N_g - 1)$ represents the grey value. N_g is the total number of distinct grey levels in the image, e.g. for an 8-bit image $N_g = 256$.

$$h_i = \frac{1}{|O|} \sum_{\mathbf{p} \in O} \begin{cases} 1, & \text{if } gr(\mathbf{p}) = i \\ 0, & \text{otherwise} \end{cases} .$$

Entropy is then found using the following formula:

$$\text{entropy} = - \sum_{i=0}^{N_g-1} h_i \log_2(h_i).$$

The symmetry of the grey value distribution of the image is measured using the feature *anisotropy*. Let k be the smallest possible value such that $\sum_{i=0}^k h_i = 0.5$, then,

$$\text{anisotropy} = \frac{1 + 2 \sum_{i=0}^k h_i \log_2(h_i)}{\text{entropy}}.$$

Inertinite is a black opaque object whereas the internal structure of vitrinite can be seen mainly towards the edges of the microfossil. One of our main goals is to distinguish between inertinite and vitrinite and so the feature *rim variability* was constructed. This measures the quantity of internal material visible around the periphery of the microfossil.

Rim variability is computed using the grey scale image of the microfossil. The set of pixels R_v is formed as follows: $R_v = \{\mathbf{p} \mid D(\mathbf{p}) \leq \frac{1}{5}r_{\text{in}}\}$. This set comprises of all pixels in the object that are less than or equal to one fifth of the inner radius away from the object boundary. Let $gr(\mathbf{p})$ be the grey level intensity of pixel \mathbf{p} , then rim variability is calculated as follows:

$$\begin{aligned} \bar{g}r_{\text{rim}} &= \frac{\sum_{\mathbf{p} \in R_v} gr(\mathbf{p})}{|R_v|} \\ \text{rim_variability} &= \frac{1}{|R_v|} \sum_{\mathbf{p} \in R_v} (gr(\mathbf{p}) - \bar{g}r_{\text{rim}})^2. \end{aligned}$$

This is a maximum likelihood estimate of the variance of grey values for the set of pixels R_v .

The grey level co-occurrence matrix (GLCM)

The GLCM was originally used for texture measurements by Haralick et al. (1973) and is a computationally quick method for analysing textural properties of an image. The GLCM is defined over a grey level image and contains the distribution of neighbouring intensity values in a specific direction defined by an offset vector $\mathbf{d} = (\delta x, \delta y)$. Let I be the grey intensity image of size $(m \times n)$ and $P^{\mathbf{d}}$ be the non-normalised GLCM defined as follows:

$$P^{\mathbf{d}}(i, j) = \sum_{p=1}^n \sum_{q=1}^m \begin{cases} 1, & \text{if } I(p, q) = i \text{ and } I(p + \delta x, q + \delta y) = j \\ 0, & \text{otherwise} \end{cases},$$

where $i, j = 0, \dots, (N_g - 1)$, and the variable N_g is the total number of distinct grey levels in the image.

The GLCM with offset $\mathbf{d} = (0, 1)$ is computed for the image in figure 2.8(a) displaying 4 grey levels. The results are shown in figure 2.8(b). The offset parameter specifies the distance and angle between the pixel of interest and its neighbour. In this example pixels horizontally adjacent to each other are compared. Altering the offset will control the direction of compared pixels. The highlighted red pixels in figure 2.8(b) indicate the pixels of interest with intensity 1 on the left and 3 on the right. There are two such occurrences hence the entry $P^{\mathbf{d}}(1, 3) = 2$ (note that the indexing of columns and rows in the matrix begins at 0 and not 1). The element in this matrix highlighted in green demonstrates the only occurrence of a pixel of interest with intensity 0 to the left of a pixel with intensity 1.

Due to the offset parameter, GLCM is sensitive to rotation of the input image. In order to introduce some degree of rotational invariance four GLCM's with angular relationship of 0° , 45° , 90° and 135° using offsets $\mathbf{d}_1 = (0, d)$, $\mathbf{d}_2 = (-d, d)$, $\mathbf{d}_3 = (-d, 0)$ and $\mathbf{d}_4 = (-d, -d)$ respectively, are used. The parameter d specifies the distance between

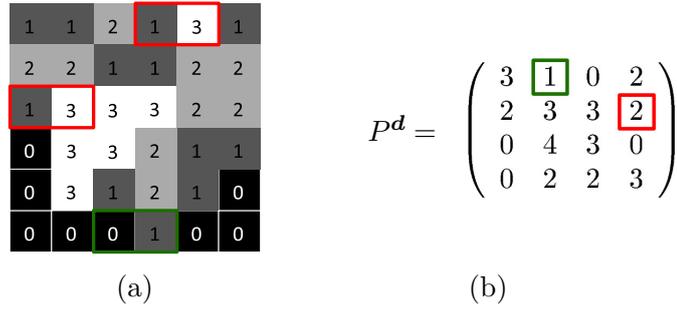


Figure 2.8: (a) Image containing 4 grey levels. (b) GLCM of the image with offset $\mathbf{d} = (0, 1)$.

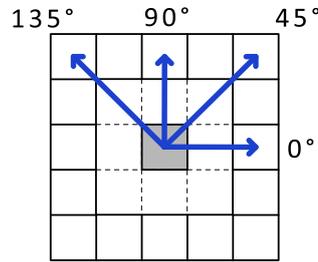


Figure 2.9: Arrows show angular relationship 0° , 45° , 90° and 135° between pixel of interest (coloured grey) and neighbouring pixels. Offsets used are $\mathbf{d}_1 = (0, d)$, $\mathbf{d}_2 = (-d, d)$, $\mathbf{d}_3 = (-d, 0)$ and $\mathbf{d}_4 = (-d, -d)$ respectively.

neighbouring pixels. A demonstration of the angular relationships is shown in figure 2.9. In our example $d = 1$. The four GLCM's are shown in figure 2.10. A final GLCM with rotation invariance is formed by calculating the average of these four matrices.

This process does not produce a symmetric matrix. For instance, the number of times 1 appears to the left of 2 would not be the same as the number of times 2 appears to the left of 1. A symmetric matrix equivalent to the GLCM described by Haralick et al. (1973) is formed by addition of two non-symmetric GLCM's. For example a symmetric GLCM P with angular relationship 0° is computed as follows

$$P = P^{\mathbf{d}_1} + P^{-\mathbf{d}_1}$$

Similarly this can be done with GLCM's of other angular relationships. The function

$$\begin{aligned}
 P^{\mathbf{d}_1} &= \begin{pmatrix} 3 & 1 & 0 & 2 \\ 2 & 3 & 3 & 2 \\ 0 & 4 & 3 & 0 \\ 0 & 2 & 2 & 3 \end{pmatrix} & P^{\mathbf{d}_2} &= \begin{pmatrix} 1 & 1 & 1 & 3 \\ 0 & 3 & 3 & 1 \\ 0 & 3 & 3 & 0 \\ 0 & 2 & 1 & 3 \end{pmatrix} \\
 P^{\mathbf{d}_3} &= \begin{pmatrix} 3 & 4 & 0 & 1 \\ 0 & 2 & 5 & 1 \\ 0 & 3 & 3 & 2 \\ 0 & 2 & 1 & 3 \end{pmatrix} & P^{\mathbf{d}_4} &= \begin{pmatrix} 1 & 2 & 1 & 1 \\ 0 & 2 & 3 & 2 \\ 0 & 3 & 1 & 3 \\ 1 & 2 & 2 & 1 \end{pmatrix} \\
 & \underbrace{\hspace{15em}}_{\text{Average all four matrices}} \\
 \text{Rotation invariant GLCM} &= \begin{pmatrix} 2.00 & 2.00 & 0.50 & 1.75 \\ 0.50 & 2.50 & 3.50 & 1.50 \\ 0.00 & 3.25 & 2.50 & 1.25 \\ 0.25 & 2.00 & 1.50 & 2.50 \end{pmatrix}
 \end{aligned}$$

Figure 2.10: Example demonstrating the calculation of a rotation invariant GLCM. The four matrices used were computed with respect to figure 2.8(a) and have offsets $\mathbf{d}_1 = (0, 1)$, $\mathbf{d}_2 = (-1, 1)$, $\mathbf{d}_3 = (-1, 0)$ and $\mathbf{d}_4 = (-1, -1)$, respectively.

`graycomatrix()` in Matlab can be used to create both symmetric and non-symmetric GLCM's.

Once a GLCM is formed, it can be normalised by dividing by a normalisation factor R calculated as the sum of all elements in the GLCM. For a GLCM formed on an image containing N_g grey levels, the normalisation factor is calculated as:

$$R = \sum_{i=0}^{(N_g-1)} \sum_{j=0}^{(N_g-1)} P^{\mathbf{d}}(i, j).$$

For clarity we define P_{ij} to be the $(i, j)^{\text{th}}$ element in the normalised rotation invariant GLCM. This specifies the probability of a pixel with grey intensity i being a neighbour at distance d from a pixel with grey intensity j .

Texture features derived from the GLCM

GLCM mean. Two means can be calculated from the GLCM. The first μ_r is based upon the pixels of interest,

$$\mu_r = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} iP_{ij},$$

this is the mean intensity of the pixel of interest. The second mean μ_c is based upon the neighbour pixels and is calculated as follows:

$$\mu_c = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} jP_{ij},$$

this is the mean intensity of neighbour pixels. For a symmetric GLCM these two means will be identical because each pixel is counted once as a pixel of interest and once as a neighbour.

GLCM Variance. Two variances of the GLCM are calculated depending upon the choice of reference pixels. The first variance σ_r^2 is based upon the pixels of interest,

$$\sigma_r^2 = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} (i - \mu_r)^2 P_{ij}.$$

The second variance σ_c^2 is based upon the neighbour pixels and is calculated as follows:

$$\sigma_c^2 = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} (j - \mu_c)^2 P_{ij}.$$

These variances are identical if the GLCM is symmetric.

Energy. This feature is also known as the angular second-moment and describes the uniformity of the image.

$$\text{energy} = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} P_{ij}^2.$$

The energy measure ranges between 0 and 1. An image with constant intensity will have energy 1.

Contrast. The intensity contrast between a pixel and its neighbour is measured over the whole image using the following function:

$$\text{contrast} = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} |i-j|^2 P_{ij}.$$

The contrast of an image ranges between 0 and $(N_g - 1)^2$. A contrast measure of 0 corresponds to an image of constant intensity.

Homogeneity. This measure evaluates the closeness of the distributed elements in the GLCM to the GLCM diagonal.

$$\text{homogeneity} = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} \frac{P_{ij}}{1 - |i-j|}.$$

High values correspond to a uniform image where there is little change in pixel intensities. Low values represent a more random image (Williams et al., 1998). This measure is also known as the *inverse difference moment*.

Correlation. The linear dependency of neighbouring pixels is evaluated using the correlation measure.

$$\text{correlation} = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} \frac{(i - \mu_r)(j - \mu_c)P_{ij}}{\sigma_r \sigma_c}.$$

The measure or correlation ranges between -1 and 1. A value of 1 or -1 means the image is perfectly correlated. A singularity occurs for an image with constant intensity. The $(k, k)^{\text{th}}$ entry of a GLCM constructed on an image with constant intensity k will be the only entry greater than zero. Hence $P_{kk} = 1$ and $P_{ij} = 0$ for $i \neq k$ and $j \neq k$. In this case the denominator of the correlation measure $\sigma_r \sigma_c = 0$, therefore the measure is undefined. Matlab handles this by defining the correlation to be NaN (Not-a-Number), however the correlation of the original pixel values is perfect, so preferably the GLCM correlation of a constant image should be 1.

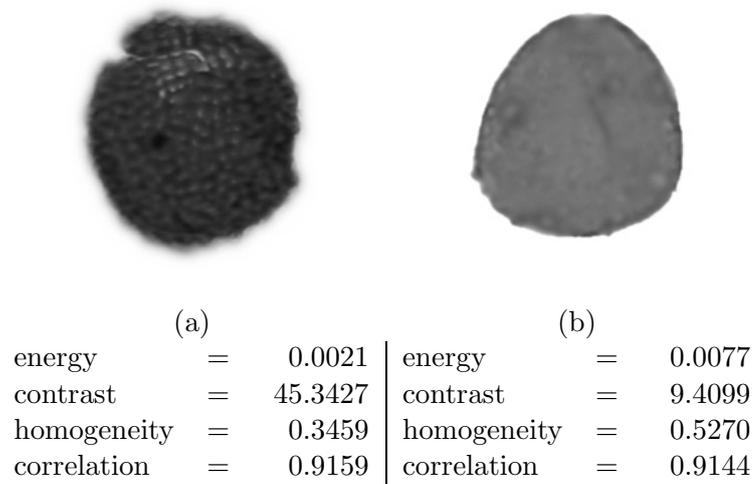


Figure 2.11: GLCM features calculated for two different palynomorphs (a) and (b).

Matlab can be used to calculate energy, contrast, homogeneity and correlation using the function `graycoprops()`. This function requires as input the GLCM of the greyscale image.

Example of GLCM texture features

Two different types of palynomorph are presented in figure 2.11(a) and 2.11(b). The palynomorph in figure 2.11(a) has a dappled texture whereas figure 2.11(b) is smooth. The GLCM texture features are shown under each image. The most distinguishing features are contrast and homogeneity indicating that in comparison to figure 2.11(b), figure 2.11(a) has a more random texture with relatively large changes in pixel intensities. Both images show high correlation; for the dappled palynomorph this signifies a uniform dappling.

To illustrate the differences between the GLCM's of the two palynomorphs, figure 2.12(a) and 2.12(b) show the GLCM's of figure 2.11(a) and 2.11(b) respectively. The GLCM's are displayed as an image where higher intensities correspond to larger values within the matrix.

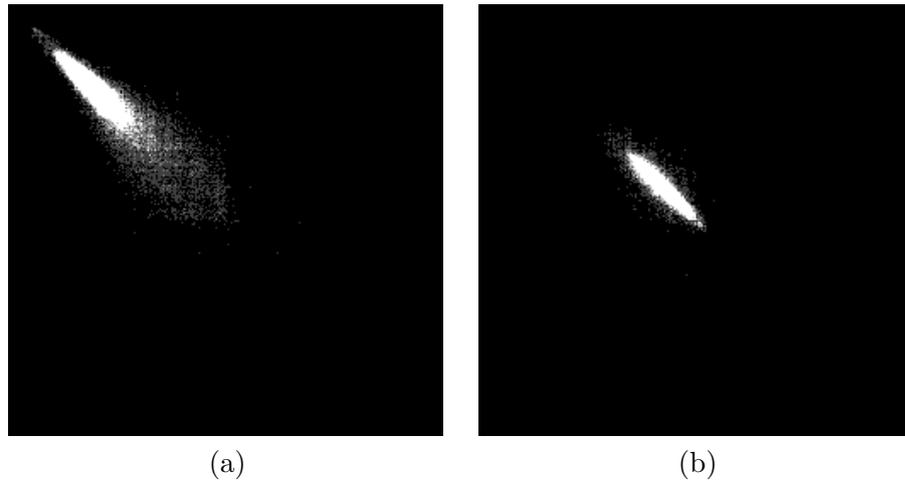


Figure 2.12: Illustration of GLCM's constructed on figure 2.11(a) and 2.11(b). Numbers within GLCM are represented as intensities, higher intensities for larger numbers.

The extraction of texture features can also be accomplished using alternative methods to the GLCM. These methods include Gabor filters, Markov random fields, or local binary patterns.

2.6.4 Shape features

A human would describe shape in an approximate way using adjectives such as long, thin, fat, round, or we use a well known item for comparison e.g. “The object is shaped like a...”. Hence expressing shape numerically is not straightforward. The usual approach is to combine size features in such a way that the dimensions cancel out. There are many types of size features resulting in many more dimensionless combinations that could be used as shape features. Due to this it is important to realise the inconsistency between the naming conventions of various shape features.

Object shape can be numerically compared to well known geometric shapes, such as circles, ellipses and rectangles. In most cases a size feature of the object is chosen and a common shape is constructed with the same size feature. The measure of dissimilarity between the common shape and object is found through comparison of other size features.

For example *compactness* of an object is measured as a ratio of areas between object and a circle with circumference equal to the object perimeter.

Silhouette moments

These moments are based upon the silhouette of the object. Let $O = \{(x, y) \mid (x, y) \text{ is an object pixel}\}$. The normalised central moments of O are given by

$$\mu_{jk} = \frac{1}{A} \sum_{(x,y) \in O} (x - \mu_x)^j (y - \mu_y)^k,$$

Where μ_x and μ_y are the mean coordinates in the x and y direction respectively. The normalisation factor $A = |O|$ is the area of the object. The three features most commonly used are known as the variance in the x direction μ_{20} ; the variance in the y direction μ_{02} and the covariance μ_{11} .

Elliptic dimensions

The semi-major and semi-minor axis lengths of an ellipse that approximately fits the shape of the object are used as features. These can be found by using silhouette moments. A covariance matrix can be formed from the moments as follow:

$$C = \begin{pmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{pmatrix}.$$

The direction of largest and smallest variation can be found by computing the eigenvectors of C . The variances in these directions are the eigenvalues, the largest variance λ_1 and smallest variance λ_2 .

$$\lambda_1 = \frac{1}{2} \left(\mu_{20} + \mu_{02} + \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2} \right)$$

$$\lambda_2 = \frac{1}{2} \left(\mu_{20} + \mu_{02} - \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2} \right)$$

These eigenvalues are used to find the dimensions of an approximating ellipse. If only the boundary pixels are considered when calculating the moments then $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$ are the semi-major and semi-minor axis lengths of the ellipse (Tsai et al., 1999). When all object pixels are used the approach adopted by (Rocha et al., 2002) finds the rectangle with equivalent covariance matrix. The length l and width w of the rectangle is found as

$$l = \sqrt{6\lambda_1},$$

$$w = \sqrt{6\lambda_2}.$$

These dimensions are used as the semi-major and semi-minor axis lengths. In our case we choose to find the dimensions of an ellipse with equivalent covariance matrix to the object.

An ellipse centred at the origin with major and minor axes parallel to the x and y axes has the equation

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1.$$

Here a is the major-axis length and b is the minor-axis length. The variance of an ellipse in the x direction is equivalent to the normalised moment μ_{20} . This can be calculated in the form of a double integral

$$\mu_{20} = \frac{1}{\pi ab} \int_{-a}^a \int_{-y(x)}^{y(x)} x^2 dy dx,$$

where the normalisation factor πab is the area of an ellipse. Substituting for the parametric representation of an ellipse $x = a \sin t$ and $y = b \cos t$, $t \in [0, 2\pi]$ enables us to integrate as

follows:

$$\begin{aligned}
\mu_{20} &= \frac{2}{\pi ab} \int_{-a}^a x^2 y(x) dx \\
&= \frac{2}{\pi ab} \int_{-\pi/2}^{\pi/2} a^3 b \sin^2 t \cos^2 t dt \\
&= \frac{2a^2}{\pi} \int_{-\pi/2}^{\pi/2} \sin^2 t \cos^2 t dt \\
&= \frac{a^2}{2\pi} \int_{-\pi/2}^{\pi/2} \sin^2(2t) dt \\
&= \frac{a^2}{2\pi} \left[\frac{1}{2}t + \frac{1}{4}\sin(2t) \right]_{-\pi/2}^{\pi/2} \\
&= \frac{a^2}{4}.
\end{aligned}$$

Similarly the normalised variance of the ellipse in the y direction is $\mu_{02} = b^2/4$. By equating the variances of the ellipse with the eigenvalues of the object covariance matrix, we can solve for a and b :

$$\begin{aligned}
\frac{a^2}{4} &= \lambda_1, & \frac{b^2}{4} &= \lambda_2, \\
a &= 2\sqrt{\lambda_1}, & b &= 2\sqrt{\lambda_2}.
\end{aligned}$$

The angle of the ellipse can also be calculated but this is withheld from our list of features as it is not rotation invariant.

Shape features derived from elliptic axes

The dimensions of the fitted ellipse are combined to form shape features. The feature *anisometry* measures the inequality in the dimensions of an object:

$$\text{anisometry} = \frac{a}{b}.$$

For a perfect circle, i.e. an object with equal dimensions, the anisometry = 1. The relation of elliptic area to object area is used to determine the *bulkiness*

$$\text{bulkiness} = \frac{\pi ab}{A}.$$

A circle with radius equal to the semi-major axis length a is compared to the object by combining two shape features into one feature known as *structure factor*:

$$\begin{aligned} \text{structure factor} &= \text{bulkiness} * \text{anisometry} - 1, \\ &= \frac{\pi a^2}{A} - 1. \end{aligned}$$

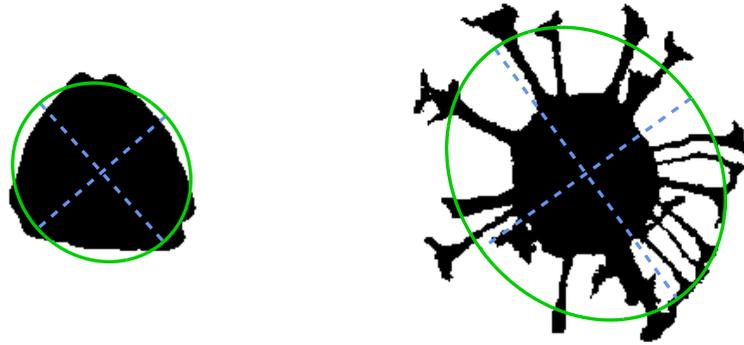
Elliptic shape features have been calculated for two palynomorphs, their silhouettes are shown in figure 2.13(a) and 2.13(b). Overlaid on each image is the ellipse with equivalent covariance matrix, the axes of the ellipse are shown with a dashed line and feature values are displayed underneath. Anisometry for both palynomorphs is close to 1 indicating their circular/rounded appearance. The ‘legs’ of the palynomorph in figure 2.13(b) are characterised by the features bulkiness and structure factor. Bulkiness is greater than 1 and structure factor is greater than zero, implying there are holes within the object or the object boundary differs to that of an ellipse/circle. Compare this to figure 2.13(a) where the palynomorph is rounded and contains no holes, bulkiness is near 1 and structure factor is near 0.

Other shape features

The most compact shape is a circle; the measure of compactness relates circle area to object area. *Compactness* is defined as the ratio of areas between object and a circle with circumference equal to the objects perimeter length p .

$$\text{compactness} = \frac{4\pi A}{p^2}$$

Standard deviation of all distances from the centre of the object to each perimeter pixel is used as a measure of object irregularity along the periphery. Let P be the set of all



	(a)		(b)
anisometry	=	1.0910	anisometry = 1.2002
bulkiness	=	1.0328	bulkiness = 1.6396
structure factor	=	0.1268	structure factor = 0.9679

Figure 2.13: Elliptic shape features for the silhouettes of a rounded palynomorph (a) and palynomorph with legs (b)

contour pixels. The measure *sigma* σ is calculated as follows:

$$\sigma = \left[\frac{1}{|P|} \sum_{(x,y) \in P} ((x - \mu_{10})^2 + (y - \mu_{01})^2) \right]^{1/2},$$

This feature is used together with *distance* in order to obtain the new feature *roundness*:

$$\text{roundness} = 1 - \frac{\text{sigma}}{\delta}$$

Convexity is the ratio of the convex hull area A_c to the object area.

$$\text{convexity} = \frac{A}{A_c}$$

The basic rectangle of an object is defined as a bounding rectangle with major axis length equal to the object *diameter* d . The ratio of minor to major axis length of the rectangle forms the feature *eccentricity*:

$$\text{eccentricity} = \frac{w}{d},$$

where w is the minor axis length of the rectangle.

The number of sides required to construct the shape if a regular polygon were used is known as the feature *sides* (MVTec Software GmbH, 2008). It is defined as follows:

$$\text{sides} = 1.41 \left(\frac{\delta}{\sigma} \right)^{0.4724}.$$

Finally we chose to include the feature *equant lath* which measures the equant to lath ratio of the object. This is an important feature concerning kerogen microfossils. The distance travelled by kerogen from its initial deposit is correlated with equant to lath ratio (Tyson and Follows, 2000; Hoelstad et al., 1994), providing valuable information regarding hydrocarbon potential. Equant to lath ratio is defined as the ratio between lengths *inner radius* r_{in} and *diameter* d

$$\text{equant lath} = \frac{r_{\text{in}}}{d}.$$

2.6.5 Summary of features

A total of 32 features describing size, colour, texture and shape are chosen to numerically represent an object. These features are summarised in table 2.1 containing a brief explanation.

2.7 Stages of the image processing

A system for automatic identification of palynomorphs can be split into 4 image processing stages, Image acquisition; background segmentation; microfossil segmentation and classification. The image acquisition processes explained in section 2.4 results in a digital colour image of size 1704 by 2272 pixels. These images of interest contain microfossils and other organic debris on a light background. A typical example is shown in figure 2.14(a). These images are taken using a 10x objective microscope lens and the circular field of view is approximately 500 μm in diameter.

Table 2.1: Groups of features

Type	Feature	Notation	Explanation
Colour	mean red	\bar{r}	
	mean blue	\bar{b}	
	mean green	\bar{g}	
	mean grey	$\bar{g}r$	
Size	inner radius	r_{in}	Radius of the largest circle contained entirely within the object
	outer radius	r_{out}	Radius of the smallest circle which contains the entire object
	diameter	d	The maximum distance between 2 contour pixels
	perimeter	p	Number of pixels on the border between object and background
	circle difference	$r_{out} - r_{in}$	Difference between the outer and inner radii
	area	A	Total number of pixels comprising the object
	distance	δ	Mean distance from centre of gravity to all contour pixels
Texture	entropy		Entropy of the grey-level histogram taken as a pdf
	anisotropy		Symmetry of the grey-level histogram about its median
	correlation		Correlation between grey level intensity of neighbouring pixels
	homogeniety		Homogeniety of neighbouring pixels in the grey level image
	contrast		Contrast of neighbouring pixels in the grey level image
	energy		Energy of neighbouring pixels in the grey level image
	*rim variability		Variance of the grey level intensity in a "rim" of width $r_i/5$
Shape	anisometry	a/b	Ratio of the lengths of the major and minor elliptic semi-axes
	eccentricity	d_-/d	Ratio of the length of the minor axis of the object to d
	rectangularity	A/A_b	Ratio of object area to the area of smallest bounding rectangle
	bulkiness	$\frac{\pi ab}{A}$	Ratio of the areas of a corresponding ellipse and the object
	convexity	A/A_c	Ratio of the object's area to its convex area
	variance x	μ_{20}	Variance across x-axis with respect to centre of gravity
	variance y	μ_{02}	Variance across y-axis with respect to centre of gravity
	covariance	μ_{11}	
	compactness	$4\pi A/p^2$	Ratio of the area to that of a circle with the same perimeter
	sigma	σ	Standard deviation of distances from centre of gravity to contour
	roundness	$1 - \sigma/\delta$	
	sides	$1.41 \left(\frac{\delta}{\sigma}\right)^{0.4724}$	Number of pieces of a regular polygon
	*equant/lath	r_{in}/d	Equant/lath ratio
structure factor	$\frac{\pi(a)^2}{A} - 1$	anisometry \times bulkiness - 1	

*Features developed for this application.

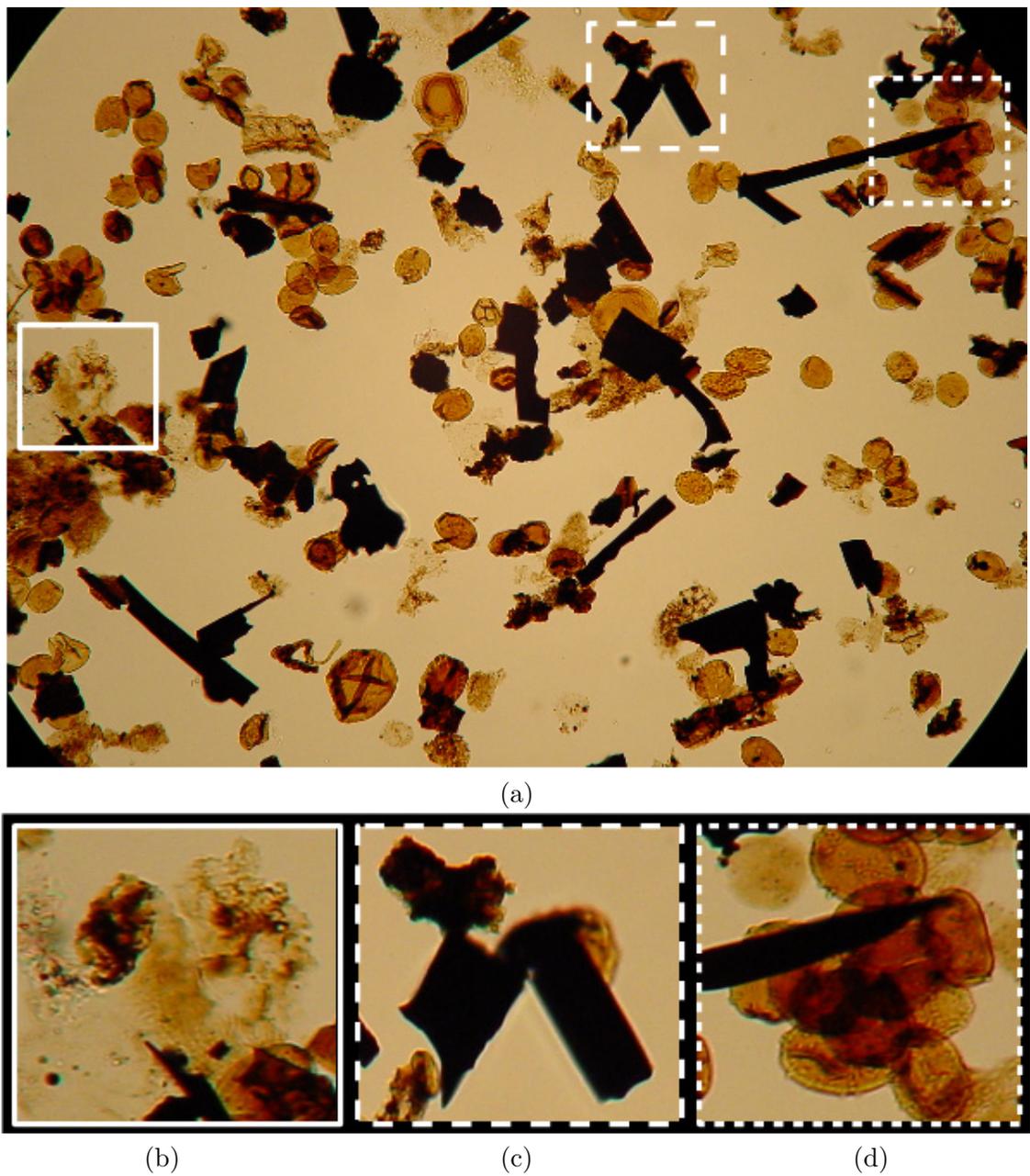


Figure 2.14: Typical example of microscopy image containing palynofacies

The next step in the system is to identify regions containing only microfossils. This is accomplished through background segmentation. The main difficulty arises due to uneven light intensity across the image. This prevents the use of simple approaches such as applying a global threshold to grey intensity values. To solve this we develop a new local thresholding technique which is detailed in chapter 3.

Regions categorised as microfossil are fed into the next level of segmentation. Touching and overlapping microfossils are split into individual pieces by an algorithm presented in chapter 4. The material on the slide is tricky to segment even by human eye. Some palynomorphs are semi-transparent and form clumps that hide certain features including edge information. This makes it near impossible to recognise individual palynomorphs from clumps. In manual procedures, the operator uses a small brush to push the clumps apart; an automatic system (being unmanned) has no such luxury. An example of clumped palynomorphs is shown in figure 2.14(d).

Amorphous material by nature does not have a clear outline and can even merge with background intensities. Although this effect hinders background segmentation it causes problems when distinguishing between closely packed amorphous material. These difficulties are illustrated by an example in figure 2.14(b).

The kerogen matter poses a different problem. Kerogen has a sharp distinct outline that is easily identified on a light background. However, due to its dark colouring this outline is lost when kerogen is overlaid or touches other kerogen material. These effects are shown in figure 2.14(c).

Next, extracted microfossils are classified in chapter 5. Each fossil is represented as a vector of features and used as input to a classification algorithm. The first step is to analyse kerogen and group it into inertinite or vitrinite classes. In chapter 6 we extend the automatic recognition to identification of complete palynomorphs.

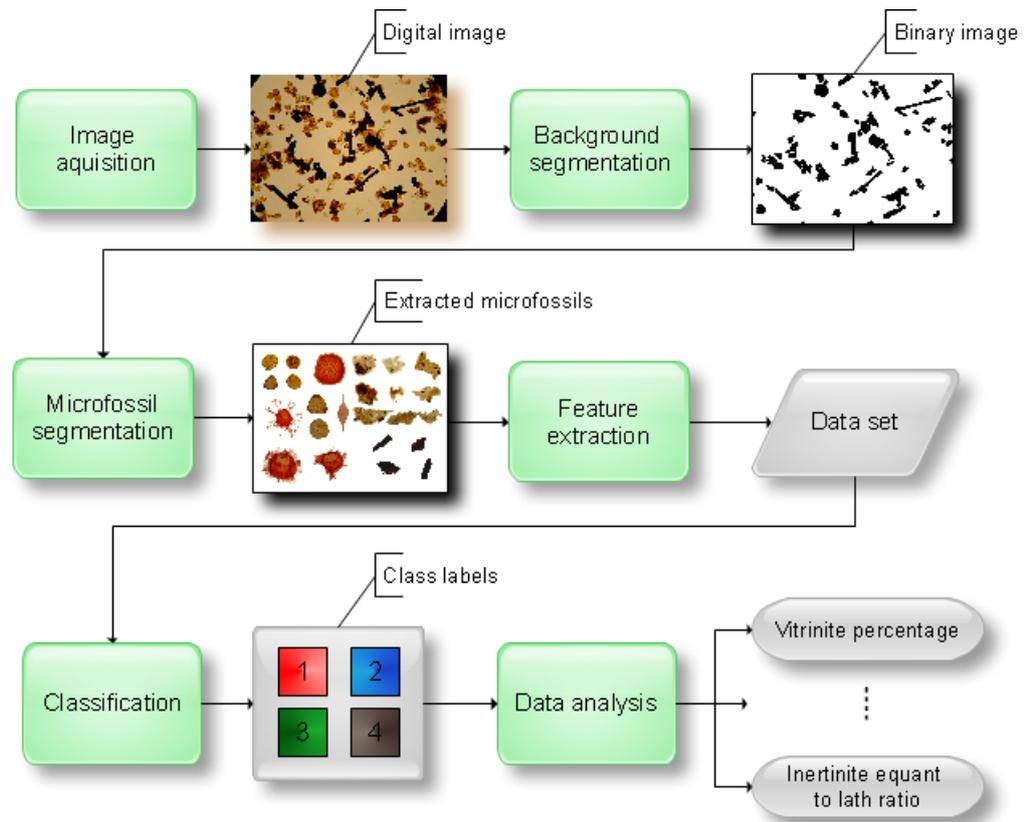


Figure 2.15: Flow diagram showing processes for automatic analysis of palynofacies

Analysis of the microscope slide concludes by extracting important information such as percentage of inertinite or mean equant to lath ratio of kerogen. The final analysis depends on the output of previous processes, so it is important to study the sensitivity of a stage output with respect to the input. For example small changes in background segmentation would ideally not affect classification accuracy. A flow diagram in figure 2.15 shows the image processing stages.

Nomenclature 1

C Covariance matrix.

D Euclidean distance function/transform.

P Set of all contour pixels.

P^d Non-normalised GLCM with offset vector d .

P_{ij} Normalised rotation invariant GLCM.

π Mathematical constant ≈ 3.14159265359 .

b Blue intensity of a pixel.

g Green intensity of a pixel.

gr Grey intensity of a pixel.

p Perimeter length.

r Red intensity of a pixel.

u_{jk} Silhouette moment.

Chapter 3

Background segmentation

A non-trivial task in image analysis is that of segmentation. Usually this is the first step in preparing the image for input into an automatic system (Petrou and Bosdogianni, 1999). The process consists of partitioning the image into non-overlapping labelled regions. Each region should contain identical sets of attributes or properties (Gonzalez et al., 2004). Formally, the segmentation of an image R is a set of regions $\{R_1, R_2, \dots, R_N\}$, where R_i is a set of pixels from the image and i is the label given to a region. The segmented regions should satisfy the following requirements:

1. $\bigcup_{i=1}^N R_i = R$.
2. $R_i \cap R_j = \emptyset$, for $i \neq j$.
3. $P(R_i) = \text{True}$, $\forall i$.
4. $P(R_i \cup R_j) = \text{False}$, for $i \neq j$.

Where P returns true if all the pixels in a region have the same property, false otherwise. An example of a pixel property would be if it's grey intensity is equal to a predefined value. The outcome of the segmentation is based entirely upon the choice of properties. Requirement (1) ensures the whole image can be formed by combining all regions; (2) implies

no two regions overlap¹. Requirement (3) certifies a meaningful segmentation. The last requirement guarantees two regions labelled differently will not satisfy the same properties when combined.

There are three possible approaches to segmentation: Edge detection, boundary detection or direct detection of the regions. Edge detection techniques try to find the edges of objects and then combine them to form region boundaries. In noisy images a popular choice of edge detection algorithm is that proposed by Canny (1986). The idea of boundary detection is to bypass edge detection altogether and find the boundaries in one step. This can be accomplished through the use of level sets Osher et al. (2002); a curve is propagated until an energy function has been minimized.

There are many well established techniques to directly detect regions. These include thresholding, clustering, region merging, region growing, region splitting and merging and application of pattern recognition classifiers. In section 3.2 we detail methods that detect regions directly. These methods will be applicable to background segmentation within microscope images.

In the case of background segmentation only two regions are necessary, background and foreground. There are infinitely many types of images containing their own unique background and foreground (defined by the observer). Hence each one requires its own method of segmentation. The next section will outline the specific problems with background segmentation in microscopy images.

3.1 Microscopy images

A microscope is an optical system built from lenses, mirrors and or prisms. The digital camera used to capture the image generally consists of a lens and Charged Coupled

¹Overlapping regions will be treated later.

Device (CCD). Defects in the apparatus causes blurring, colour changes and geometric distortions in the captured image (Du Buf and Bayer, 2002).

Microscopy images are commonly afflicted with uneven brightness. The periphery of the image is usually poorly illuminated and this is known as *vignetting*. There are three main types of vignetting: mechanical, optical and pixel. Mechanical vignetting is caused by the physical construction of the optical viewing device while optical vignetting is inherent in the lens design. Pixel vignetting only occurs in digital cameras due to less light hitting a photon cell in the photon sensor at an oblique angle, i.e., towards the edges of the image. There are two types of photon sensor a Charged-coupled device (CCD) or less commonly used Complementary metal-oxide-semiconductor (CMOS).

The microscopy images of interest in this study contain dark microfossils on a light background. The images are taken with transmitted light microscopy. This uses a light source below the microscope slide for illumination purposes. The concentrated light source compounds the effect of vignetting causing even worse illumination across the image. An example image is shown in figure 3.1 (a). The contrast of the image has been adjusted to better show the uneven illumination effect in figure 3.1 (b). The vignetting effect can be reduced through careful setup of equipment. Inevitably though the equipment becomes misaligned and readjustment is necessary. For instance, the brightness of the halogen bulb illuminating the slide will dim with time.

The uneven lighting distribution is problematic when it comes to background segmentation. Therefore, either the lighting can be corrected prior to background segmentation, or accounted for within the segmentation method.

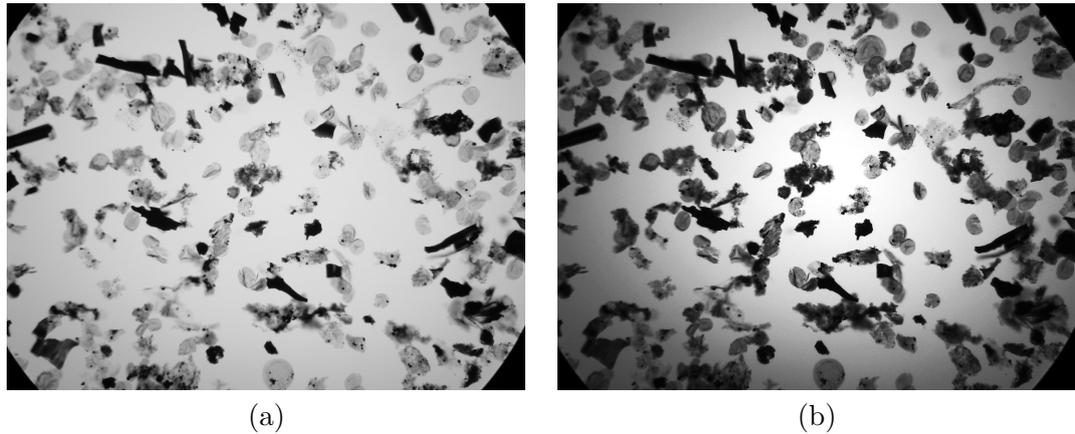


Figure 3.1: Image of a palynofacies slide illustrating the uneven lighting across the image. The slide is lighter in the centre and becomes progressively darker towards the edges. (a) Original gray scale image. (b) Contrast adjustment of original image to better show the effect of uneven lighting

3.2 Segmentation methods based on thresholding the original image

Background segmentation splits the image into two regions, background and foreground. This segmentation can be represented by a binary image consisting of ones and zeros. There is no convention specifying that the foreground should be labelled 1 or 0. Here we represent foreground as 0 and background as 1. The binary image is displayed in black and white. A pixel is white if 1 and black if 0. Hence foreground will be shown as a black region on a white background.

3.2.1 Global thresholding

One of the simplest segmentation methods is that of thresholding grey levels in the image (Weszka, 1978; Sankur and Sezgin, 2004). Our images of interest contain dark objects on a light background, hence the darker pixels should be set to foreground. Let M be the set of all image pixels. The binary image B is formed by thresholding M at

$T \in [0, N_g]$, where N_g is the total number of distinct grey levels:

$$B(x, y) = \begin{cases} 1, & \text{if } gr(x, y) > T \\ 0, & \text{if } gr(x, y) \leq T \end{cases}, \quad (3.1)$$

here $gr(x, y)$ is the grey intensity at pixel (x, y) . The value x and y are the column and row positions of the pixel respectively, as described in section 2.5. The threshold can be chosen manually by adjusting T until a good segmentation is achieved. In most image processing applications this process is usually aided by using the grey level histogram of the image. Some automatic threshold selection methods will also use histogram information, such as shape and curvature.

This histogram is formed by counting the number of pixels with a specific grey intensity. For an image with dark objects on a light background the histogram will contain two peaks. The peak containing lower intensities represents the foreground and the peak containing higher intensities represents the background. The challenge is to find a threshold value that best separates these two peaks. This is a relatively simple task for a bimodal histogram as the threshold can be chosen where the grey value is at a minimum between the two peaks of the histogram. In most cases the histogram will contain noise and require smoothing or approximating (Ramesh et al., 1995) prior to automatic analysis. A refined analysis can be conducted by using vector quantisation to reduce the number of histogram bins.

Optimal threshold

If the foreground and background pixels have a known grey level probability distribution then the image can be optimally thresholded (Kittler and Illingworth, 1986). The optimal threshold for grey scale image is as follows. Let $p_b(x)$ be the probability density function of the grey levels in the background region. Let $p_f(x)$ be the probability density

function of the grey levels in the foreground region. Suppose θ is the proportion of background pixels then $(1 - \theta)$ is the proportion of foreground pixels. The optimal threshold value T is found by solving

$$\theta p_b(T) - (1 - \theta)p_f(T) = 0. \quad (3.2)$$

For real world images the distributions are usually unknown. Even so, there are algorithms that can approximate this optimal threshold (Ramesh et al., 1995).

Clustering image histogram

Clustering methods can also be used to approximate the optimal threshold. Grey level values are used as input to clustering techniques, two clusters are searched for that represent the two peaks of the histogram. A number of authors propose fitting a mixture of Gaussians to the histogram of relative frequencies (Sankur and Sezgin, 2004). In this case we can solve equation (3.2). Let the background Gaussian have mean μ_b and variance σ_b^2 and the foreground Gaussian have mean μ_f and variance σ_f^2 . Then the distributions are as follows:

$$p_b(g) = \frac{1}{\sqrt{2\pi}\sigma_b} \exp\left(-\frac{(g - \mu_b)^2}{2\sigma_b^2}\right), \quad (3.3)$$

$$p_f(g) = \frac{1}{\sqrt{2\pi}\sigma_f} \exp\left(-\frac{(g - \mu_f)^2}{2\sigma_f^2}\right), \quad (3.4)$$

where g is grey intensity. Substituting into equation (3.2) yields the quadratic

$$AT^2 + BT + C = 0, \quad (3.5)$$

where

$$\begin{aligned} A &= (\sigma_b^2 - \sigma_f^2), \\ B &= 2(\mu_b\sigma_f^2 - \mu_f\sigma_b^2), \\ C &= \mu_f^2\sigma_b^2 - \mu_b^2\sigma_f^2 - 2\sigma_b^2\sigma_f^2 \ln\left(\frac{\sigma_b(1-\theta)}{\sigma_f\theta}\right). \end{aligned}$$

This can be solved for T , there are two solutions but only one will be feasible.

Another popular clustering method is that proposed by Otsu (1979). It does not depend upon modelling the probability density functions of grey levels in background and foreground regions. A threshold value is chosen that minimises the within class variance and maximises the between class variance. A comparison of over 20 global thresholding techniques conducted by Sahoo et al. (1988) showed Otsu's method to have best performance. Two drawbacks to this technique are that the two clusters are only described by their means and variance; this may not be sufficient to accurately model them. Secondly the method assumes a bimodal histogram and this may not necessarily be true. However, Otsu's method of thresholding can be generalised to more than two classes using a multilevel threshold selection procedure (Ping-Sung Liao and Chung, 2001). The Otsu method of thresholding is coded as a standard Matlab function, `graythresh()`, in the Image Processing Toolbox (IPT).

Our images of interest contain darker and lighter objects (two foreground classes) and the background. The dark and light objects correspond to kerogen and palynomorph or amorphous material respectively. Because three classes are present in the grey level histogram, Otsu's method of thresholding becomes unsatisfactory. We propose searching for three clusters within the image histogram. As an example we fit three Gaussians to the grey level histogram of the image in figure 3.1 using maximum likelihood estimation. The three fitted Gaussians are overlaid on the histogram in figure 3.2. A threshold is found by

solving equation (3.5) for the two Gaussians with highest means, this is shown in figure 3.2.

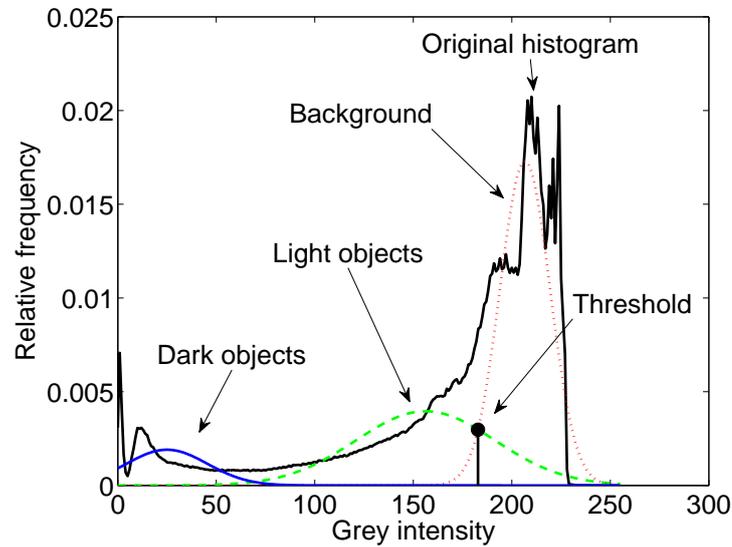


Figure 3.2: The grey level histogram of the image in figure 3.1 (a) and the fitted mixture of three Gaussians. The threshold is marked with a large dot.

Fitting three Gaussians gave consistently better segmentation results than Otsu's method when applied to our images. Nevertheless, due to the uneven lighting across the slide microfossils towards the edge of the image were lost within a black rim, see figure 3.3 (a). In fact there exists no global threshold that will separate the background from the foreground in these images. We can try decreasing the threshold found by fitting three Gaussians to correctly segment the outer microfossils. However, microfossils in the centre of the image become lost, see figure 3.3 (b).

This type of problem is encountered frequently in segmenting microscopy images. It was found by Bollmann et al. (2004) that certain prerequisites prior to global thresholding of microfossils are essential for optimal segmentation. They consist of constant light intensity and high contrast. If these conditions are not met then a common alternative is to use adaptive thresholding.

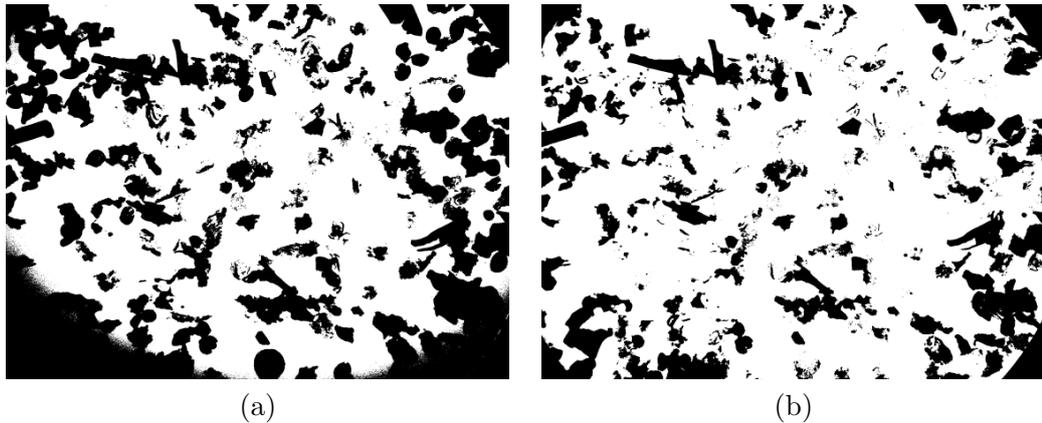


Figure 3.3: (a) Grey-scale image from figure 3.1 thresholded by fitting three Gaussians. (b) Microfossils at the centre of the image are lost in an attempt to correctly segment microfossils at the edge by lowering the threshold

3.2.2 Adaptive thresholding

A global threshold applied to images with non-uniform background illumination could work well for some image regions and poorly for others. In these cases a threshold function can be applied that changes depending upon position; such a method is called *local/adaptive thresholding* (Nakagawa and Rosenfeld, 1979). Let M be the set of all image pixels. The binary image B is formed by thresholding M at $T(x, y) \in [0, N_g]$, where N_g is the total number of distinct grey levels:

$$B(x, y) = \begin{cases} 1, & \text{if } gr(x, y) > T(x, y) \\ 0, & \text{if } gr(x, y) \leq T(x, y) \end{cases} . \quad (3.6)$$

Hysteresis thresholding

Sometimes the valley between the two peaks is not very well defined. This occurs when background regions contain the same intensity values as foreground regions. In such cases it is possible to use two thresholds, this is known as *hysteresis thresholding* (Canny, 1986).

Ideally the segmented image obtained using the highest threshold should contain all microfossil regions in black; misclassified microfossil pixels should be disconnected from these regions. However, for our images this condition cannot be upheld. For this reason hysteresis thresholding is unsuitable and causes incorrect segmentation around the edge of the image.

Windowing

Another method of adaptive thresholding is to divide the image into sub-images and apply a global threshold to each sub-image separately (Eikvil et al., 1991; Sund and Eilertsen, 2003). When the thresholded images are recombined some adjustment may be necessary because threshold values may differ greatly between neighbouring sub-images.

As an example we adaptively threshold the image shown in figure 3.4 (a). This image is of size 1704 by 2272 pixels. A windowing method can be applied by subdividing the image into squares of width 450 pixels which we call windows. A threshold is found for each window by fitting three Gaussians (discussed in section 3.2.1). Results can be seen in figure 3.4 (b). The threshold value used for each window is shown as an image in figure 3.4 (c).

The drawback of this method is its sensitivity to window size. A window has to be chosen so that it contains enough foreground and background to produce a histogram that can be split into three classes: light objects, dark objects and background. On the other hand, it has to be small enough so that the background is of near uniform intensity. For demonstration we set the window size to 200; the results can be seen in figure 3.4 (c). Notice that windows containing only one or two of the three classes are incorrectly segmented.

Microfossils vary in size within individual images and also between images due to changes in magnification. Therefore, this method is best used under supervision whereby

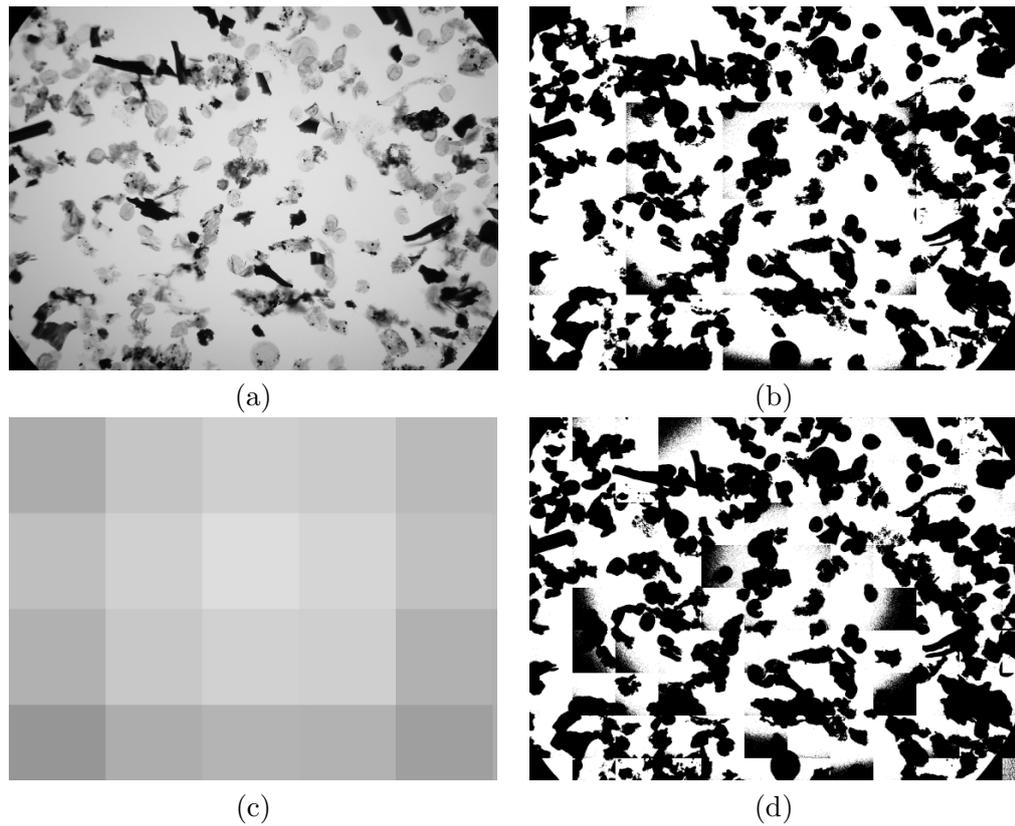


Figure 3.4: (a) Microscopy image containing palynofacies. (b) Adaptive threshold on microscopy image using square window of width 450 pixels. (c) Local threshold function displayed as an image. (d) Adaptive threshold using square window of width 200 pixels. Incorrectly segmented regions are shown.

the user can adjust window size until a good segmentation is achieved.

Background fitting

The threshold function generated through the method of windowing can have large intensity changes between neighbouring windows, see figure 3.4 (d). One solution is to interpolate a smooth function at the threshold values in the centre of each window (Nakagawa and Rosenfeld, 1979).

Alternatively, it was suggested by Yanowitz and Bruckstein (1989) that a surface could be interpolated at points where the image gradient is high. Points of high gradient are usually at the edges of foreground objects. Intensities at these positions are between background and foreground grey levels. The surface is fitted through successive over-relaxation as the solution of the Laplace equation. Other more recent techniques also use these points of high gradient as interpolation points (Blayvas et al., 2006; Chan et al., 1998).

A quadratic function has been used to model the background illumination with some success in astronomy images (Montage, 2007). The effect of vignetting is similar through a telescope as it is through a microscope. Thus we will attempt to fit a quadratic surface to the background illumination in the grey-scale microscopy image shown in figure 3.1 (a) using the following steps: Let the function $gr(x, y)$ be the grey intensity of a pixel (x, y) . Set the value of i equal to 1.

1. An estimate of the initial background illumination is calculated using a 2D quadratic function $z_0(x, y) = a + bx + cy + dx^2 + ey^2 + fxy$. The coefficients a, b, c, d, e and f are found using a least squares approximation to $gr(x, y)$.
2. The most “certain” foreground pixels are removed to form a reduced set $\{(x, y) \mid gr(x, y) > z_{i-1} - \sigma^2\}$, where σ^2 is the standard deviation of $(z_{i-1} - gr)$. A new background estimate $z_{i+1} = a + bx + cy + dx^2 + ey^2 + fxy$ is formed using a least

squares approximation to the reduced set.

3. $i \leftarrow i + 1$.
4. Repeat from step 2 until $i = 4$.

It was found empirically that repeating from step 2 until $i = 4$ gave the best results. The outcome z_3 is an approximation to the background illumination. This can be used directly as a threshold function.

Advantages of this method are that it requires no input parameters; it is completely automatic and can be computed relatively quickly compared to previous method. On the downside however in most of the images we tested the method would fail to segment some microfossils. An example segmentation using this method was applied to the image in figure 3.1 (a). Figure 3.5 (a) demonstrates the approximation z_3 to the background illumination. Segmentation using this as a threshold function is illustrated in figure 3.5 (b). Many microfossils in the lower section of the image are only partially captured or missing. This is possibly due to lack of flexibility within the quadratic function. However, fitting polynomials of higher degree using the same approach resulted in even worse segmentation results.

Grey-scale morphology - Bothat transform

Grey-scale morphology can be used to reduce the effects of uneven illumination. A secondary image is constructed where the darker regions are suppressed. This is then used to correct the original image. Grey-scale dilation of the image f with a structuring element b is defined as

$$(f \oplus b)(x, y) = \max \{gr(x - x', y - y') + b(x, y) \mid (x', y') \in D_b\}. \quad (3.7)$$

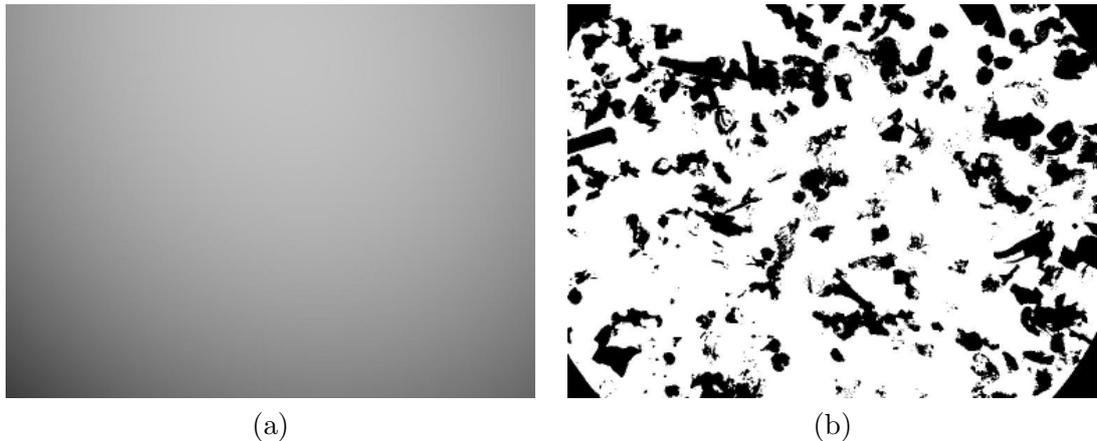


Figure 3.5: (a) 2D quadratic approximation of background illumination for image in figure 3.1 (a). (b) Segmentation using fitted 2D quadratic as a threshold function.

The function $gr(x, y)$ is the intensity of the pixel at (x, y) . A structuring element is a small image containing an origin. The shape of the image is defined by the domain D_b . The origin is defined as the position $(0, 0) \in D_b$. We will only be concerned with flat structuring elements where $b(x, y) = 0$ i.e. have a height of 0. For example, we will create a flat structuring element shaped as a disk. The origin of the structuring element will be at the geometric centre. This can be viewed pictorially in figure 3.6. Black pixels show the domain D_b . A structuring element can be created in Matlab using the function `strel()`.

With a flat structuring element image dilation becomes

$$(f \oplus b)(x, y) = \max \{ gr(x - x', y - y') \mid (x', y') \in D_b \}. \quad (3.8)$$

When the structuring element is positioned with its origin over a pixel in the image, D_b is used to define its neighbourhood. Grey-scale dilation with a flat structuring element can be thought of as replacing pixel intensity with the maximum intensity within its neighbourhood.

The opposite effect to dilation is grey-scale erosion. When a flat structuring ele-

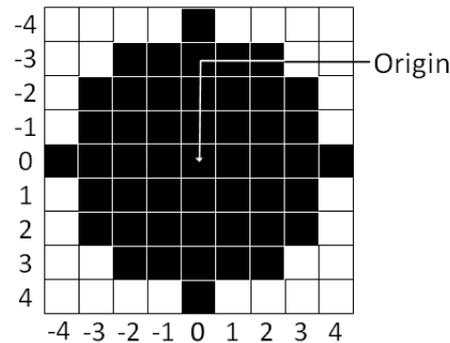


Figure 3.6: Flat disk shaped structuring element. Origin positioned at disk centre. Domain D_b is shaded in black.

ment is used, grey-scale erosion is defined as

$$(f \ominus b)(x, y) = \min \{gr(x - x', y - y') \mid (x', y') \in D_b\}. \quad (3.9)$$

Dilation and erosion can be implemented in Matlab with the functions `imdilate()` and `imerode()` respectively.

A combination of dilation and erosion can be used to suppress darker regions within an image. This is known as the *closing* (Gonzalez et al., 2004). The closing of f is defined as

$$f \circ b = (f \ominus b) \oplus b, \quad (3.10)$$

this is the erosion of f followed by the dilation of the result.

Subtracting the original image from the morphological closing is a common technique used to remove noise. This is called the *bothat transform* (Meyer, 1978). Used with the appropriately shaped and sized structuring element, the bothat transform can be used to remove uneven lighting across the image. A global threshold can then be applied. The size and shape of the structuring element should be chosen so that it is not swallowed up by the dark regions in the image.

As an example we applied the bothat transform to the image shown in figure

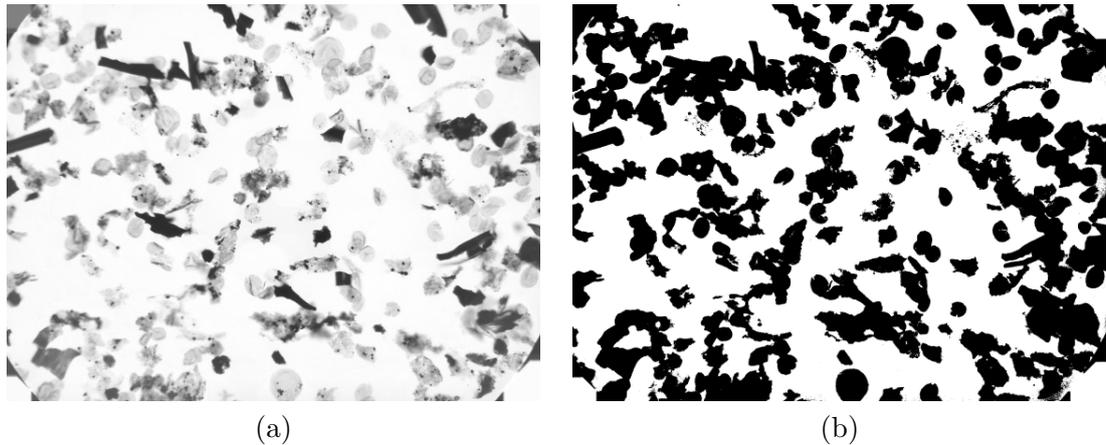


Figure 3.7: (a) Bothat transform of image in figure 3.4. (b) Manual threshold of image in (a).

3.4 (a). A disk shaped structuring element was used with radius 100 pixels. The Matlab IPT function `imbothat()` carries out the bothat transform on an image given as an argument. The output image is shown in figure 3.7 (a). A global threshold is chosen manually, the result is shown in figure 3.7 (b).

An ideal structuring element should be larger than the maximum sized foreground region. On the other hand, it should also be small enough to remove uneven background illumination. For an automatic approach the most appropriately sized structuring element can be manually chosen on a set of images and then fixed for subsequent images. For our images of interest microfossils vary in size between slides. A single sized structuring element could not be found that would produce satisfactory results. Microfossils were partly captured or completely missing in the segmentation output. A refinement to this technique can be found by using various structuring elements of different size and shape and then combining the results.

3.2.3 Clustering

Segmentation splits the image into non-overlapping regions. Each region satisfies a set of properties. Therefore it is natural to think of segmentation in the form of a clustering problem. Image pixels are represented by vector data points identifiable by features. For example this could be, position or colour. A feature space can be formed depending upon the choice of features. Similarity between pixels is measured as distance within the feature space. Pixels belonging to background are expected to be similar to one another and dissimilar to foreground pixels. Two clusters of data points should then be present. Clearly the choice of features is important for a successful segmentation. Prior to clustering, features can be automatically selected to improve the clustering method. A method using wavelets is proposed by Porter and Canagarajah (1996).

A common choice of clustering algorithm to use is *K-means* (Duda et al., 2001). Let X be the set of all data points. Initially k means are arbitrarily chosen from the elements of X , denoted $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_k$. In each iteration these means are updated as

$$\boldsymbol{\mu}_i = \frac{1}{|X_i|} \sum_{\mathbf{x}_i \in X_i} \mathbf{x}_i, \quad (3.11)$$

where $\mathbf{x}_i \in X_i$. The set X_i are all data points that are closest to $\boldsymbol{\mu}_i$ using the square Euclidean distance $\|\mathbf{x} - \boldsymbol{\mu}_i\|^2$, where $\mathbf{x} \in X$. The means are updated using equation (3.11) until there is no change in $\boldsymbol{\mu}_i$. Specific clustering algorithms for image segmentation are available. A clustering system proposed by Cinque et al. (2004) adopts a fuzzy approach for image segmentation.

As an example we apply this method to the image in figure 3.8 (b) containing palynofacies. These types of images contain three regions: light objects, dark objects and background. Hence better results were found when three clusters are used. A dataset is formed by representing each pixel as a data point. The red, green and blue values of a

pixel are chosen to be the features. The clustering method using k-means was applied to a random sample of 150 pixels from the image in figure 3.8 (b). Three clusters were identified, their projection onto the “red” and “green” dimension is shown in figure 3.8 (a). The covariance matrices of the clusters were estimated and the pixels in the original image were then labelled into foreground and background using the Mahalanobis distances to the cluster centres as follows: Let Σ_i and $\boldsymbol{\mu}_i$ be the covariance matrix and mean of the i^{th} cluster respectively. Also let pixel (x, y) be represented as $\boldsymbol{x} = (r, g, b)$, where r , g and b are the pixels red, green and blue intensity values respectively. Without loss of generality let cluster $i = 1$ represent background pixels. The pixel (x, y) is labelled as background iff:

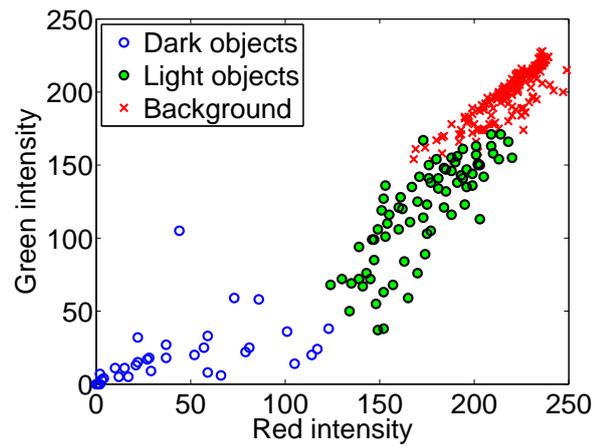
$$(\boldsymbol{x} - \boldsymbol{\mu}_1)\Sigma_1(\boldsymbol{x} - \boldsymbol{\mu}_1)^T \leq (\boldsymbol{x} - \boldsymbol{\mu}_i)\Sigma_i(\boldsymbol{x} - \boldsymbol{\mu}_i)^T, \quad (3.12)$$

for $i \neq 1$. The final segmentation is displayed in figure 3.8 (c). Although this approach is completely automatic the results are not acceptable. Some microfossils are incorrectly segmented near the edge of the image. No improvement was found when using a larger sample of pixels.

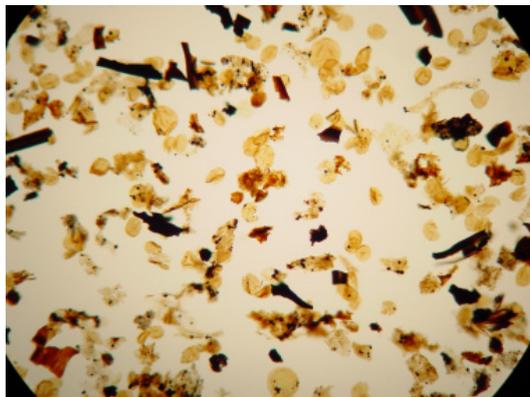
3.3 Lighting correction methods

As previously demonstrated, global thresholding produces poor segmentation results under uneven illumination. However, global thresholding becomes a powerful tool if the lighting is corrected first.

In microscopy images a simple solution is to acquire a background image where a blank slide is used in place of a sample. This image can be used to “flatten” subsequent images in a process called *background subtraction*. This term can often be used incorrectly. If the capturing device is logarithmic (i.e. uses a CMOS photon sensor) then the correct procedure is to subtract the background image (pixel by pixel) from the sample image.



(a)



(b)



(c)

Figure 3.8: (a) Three clusters identified using k-means. (b) Colour image of palynofacies. (c) Output segmentation using k-means clustering.

A digital logarithmic capturing device measures the rate of photons hitting the photon sensor continuously and produces an output that is proportional to the logarithm of the light intensity. More often the capturing device is linear (these can use CCD or CMOS photon sensors) and measures the number of photons hitting the sensor over a discrete period of time. These devices produce an output that is linearly proportional to the light intensity. In these cases the sample image should be divided by the background image. This process has been used for microfossil images by Weller et al. (2005) and France et al. (2000). There is a similar approach used in video segmentation where the currently recorded scene is “subtracted” from the next recorded scene to correct or remove background (Iwamoto et al., 2001).

An image f taken with a linear capturing device can be represented as the product of two functions

$$f(x, y) = i(x, y)r(x, y). \quad (3.13)$$

The intensity of the illuminating light source is given by $i(x, y)$ (the illumination image) and the reflectance of the objects within the image by $r(x, y)$ (the reflectance image). An image of a blank slide will correctly estimate $i(x, y)$. Dividing $f(x, y)$ by $i(x, y)$ yields only the reflectance image. Multiplying this by a constant will restore a uniform illumination to the image.

Obtaining a blank slide is not ideal for an automatic procedure. The lighting across a slide is affected by many factors including, microscope setup, bulb brightness and the thickness of the sample under examination. For this reason it is better to estimate the background lighting or filter out the uneven illumination through homomorphic filtering (Pan et al., 2004; Gonzalez and Wintz, 2002; Brinkmann et al., 1998).

3.3.1 Homomorphic filtering

Using the model of an image in equation (3.13), the reflectance can vary between zero and one. A perfectly reflective surface has a reflectance of one but an opaque surface has reflectance of zero. The illumination will tend to vary smoothly across the image and can be viewed as a signal of low frequencies. On the other hand, object surface will vary according to its properties such as physical texture. For this reason reflectance can be thought of as a signal containing high frequencies. Therefore the two functions should be separable within the frequency domain. We first compute the logarithm of the image,

$$\ln(f) = \ln(i) + \ln(r). \quad (3.14)$$

A high-pass filter can be applied to the logarithm of the image. This will suppress the illumination and enhance the reflection. This type of filtering is called *homomorphic filtering*.

Homomorphic filtering assumes that the illumination field $i(x, y)$ and reflectance field $r(x, y)$ are separable, in general this is not the case. This is why filtering will usually introduce unwanted artefacts. When applied to our microscopy images this method failed to produce reasonable results.

3.3.2 Estimating image background

With an estimated background image the uneven illumination across the sample image can be corrected (as demonstrated at the beginning of section 3.3). There are many proposed methods for acquiring an estimated background image. The state of the art techniques are outlined in this section.

Convolution

Convolving the image with a low pass filter such as a Gaussian kernel will remove high frequency information. The idea is to apply the filter until the image is devoid of fore-

ground material (Leong et al., 2003; Universal Imaging Corporation, 2004). This technique is un-automated and will need the assistance of a human controller to set parameters and make corrections within graphics editing software.

Image averaging

Many microscopy slides can be averaged to produce a background estimate. This method relies on foreground material varying in position between slides. The position of palynofacies on a slide upholds this condition. However, the averaging still produces unreasonable results due to foreground microfossils occupying a large proportion of the image area and frequent alteration in lighting conditions. Due to these changes in lighting a single estimate will not be suitable for all images. This is why we seek a method for unique background estimation based solely on the image provided.

Background fitting using a B-spline

Because the variation in background brightness is smooth with relatively low gradient it can be approximated by simple functions such as polynomials. Surface polynomials of order greater than two have also been used but, typically these are applied to images captured using techniques where vignetting is not the cause of uneven illumination (Zawada, 2003).

In the case of microscopy images the lighting is distorted due to vignetting, uneven illumination from the bulb and variation in sample thickness. It was proposed by Yu and Chung (2004) that a hyperbolic function should be used to correct for vignetting only. This method requires calibration for the camera or optical sensor. First the image of an ordinary white paper captured using the camera is used as a reference sample. A hyperbolic function is fitted to this image which models the intensity profile of the camera. This method will remove imbalanced brightness due to vignetting, provided the image has been captured

under uniform lighting.

A popular choice for background fitting is to use uniform cubic B-spline functions (Bartels et al., 1987). These functions are always C^2 continuous (i.e. the function can be differentiated twice) and flexibility is adjusted using control points. Fitting a cubic B-spline function can be expressed as a least squares problem and is therefore solved efficiently. A uniform cubic B-spline is completely determined by the set of control points.

It was suggested by Zijdenbos et al. (1991) that a B-spline function could be used to model the RF field inhomogeneity within magnetic resonance images (MRI). Intensities of certain tissues that should be similar throughout an image are distorted due to the radio frequency (RF) field produced by the imaging device. In this application the user manually picks the control points on the image.

As shown previously, an image $f(x, y)$ can be modelled using the illumination image $i(x, y)$ and reflectance image $r(x, y)$. An iterative procedure proposed by Lindblad and Bengtsson (2001) fits a uniform cubic B-spline function to the illumination image $i(x, y)$. Lindblad's method assumes $i(x, y)$ to be smooth and contain no sharp edges i.e. C^2 -differentiable. The image capturing device used is assumed to be logarithmic and so the image is modelled as

$$f(x, y) = i(x, y) + r(x, y). \quad (3.15)$$

If the image capturing device is linear then the logarithm of the image can be taken and modelled in the same way. The objective to correct the background is to remove $i(x, y)$ from the equation, resulting in $f(x, y) = r(x, y)$. To fit the B-spline function control points located only in the background of the image have to be established, this is done in an iterative manner:

1. Let Φ be the set of background pixels, initially we set this to contain all image pixels.

Let $k = 0$. A poor first guess at the corrected image is made, $\hat{f}_k = f$.

2. Estimate background pixels by thresholding \hat{f}_k at $2\sigma_k$, where σ_k is the standard deviation of grey intensity values for all pixels in Φ .
3. Update Φ so that it only contains the estimated background pixels.
4. Fit a uniform cubic B-spline \hat{i}_k to the intensities of estimated background pixels in Φ , using a least squares fit. A 5x5 grid of control points are selected to minimize $\sum_{(x,y) \in \Phi} (\hat{i}_k(x,y) - f(x,y))^2$.
5. Calculate a new estimate to the corrected image $\hat{f}_{k+1} = f - \hat{i}_k$.
6. if the standard deviation of $(\hat{i}_k - \hat{i}_{k-1}) \leq \delta$ we are finished, otherwise set $k \leftarrow k+1$ and repeat from step 2. It is suggested by Lindblad that the stoppage criterion $\delta = 1/1000$ of the range in intensity values of the input image.

The result should be an image \hat{f}_{k+1} with uniform background. The background image estimate will be given by \hat{i}_k .

This method was applied to the image in figure 3.4 (a). The corrected image is shown in figure 3.9 (a). The background estimate is shown in figure 3.9 (b).

3.4 Crossing Stripe Parabolas (CSP)

Rather than fitting a two-argument function to obtain a background estimate, we propose to fit multiple “horizontal” and “vertical” parabolas (Charles et al., 2008b). This estimate will then be used as a thresholding function across the entire image. The uneven lighting across microscopy images is predominantly due to the bulb, located beneath and central to the slide. Suppose we take a horizontal stripe across the image. An example of a horizontal stripe of a microscopy image containing microfossils is shown in figure 3.10 (a).

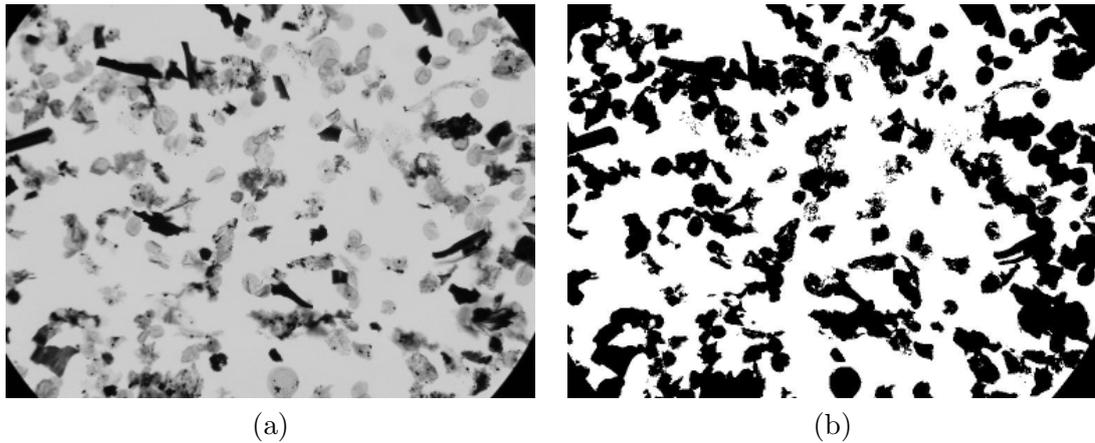


Figure 3.9: (a) Corrected background lighting of image from figure 3.4 (a) using a method by Lindblad and Bengtsson (2001). (b) Background estimate of image in figure 3.4 (a) using a uniform cubic B-spline.

The intensities on each stripe are averaged across the smaller dimension of the stripe so that a single mean line is obtained. The intensities across this line are plotted against horizontal position in figure 3.10 (b) as a continuous black line.

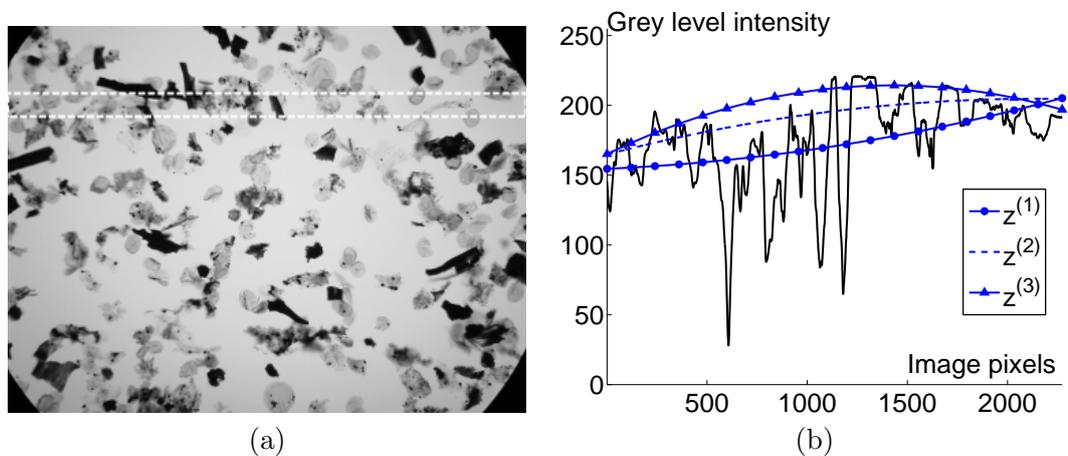


Figure 3.10: (a) An image of palynofacies with a stripe cut along the x-axis (shown in a dashed rectangle). (b) Grey level intensity of the mean line and the three subsequently fitted parabolae for the stripe in (a).

A 1-dimensional parabolic curve can be fitted to model the background intensities in the mean line (see figure 3.10 (b)). The whole image can be segmented by segmenting

many stripes using their fitted parabolic curve in a method described below:

3.4.1 The method

We have named this method *crossing stripe parabolas (CSP)* due to the nature of the algorithm. The grey level intensity image is split into K_y vertical and K_x horizontal stripes. A mean line is calculated for each stripe. A parabola is fitted on each mean line using an iterative procedure similar to that found in estimating background in powder diffraction patterns or x-ray spectra (Steenstrup, 1981; Vekemans et al., 1995; Bruckner, 2000). Consider horizontal stripe i . Denote the intensities on the mean line of the stripe by $gr_i(x)$, where x spans the width of the image. Using least squares, fit a parabola $z_i^{(1)}(x) = a_i + b_i x + c_i x^2$ to approximate $gr_i(x)$. As the mean line includes intensities of both background and foreground pixels, $z_i^{(1)}(x)$ will not model the background only. Figure 3.10 (b) shows the mean line for the stripe from figure 3.10 (a). Plotted with the dot marker is $z_i^{(1)}(x)$.

To exclude the foreground points, a second parabola is fitted, denoted $z_i^{(2)}(x)$, using a reduced set of points on the mean line $\{x \mid gr_i(x) > z_i^{(1)}(x)\}$. By requiring that the grey level intensity exceeds $z_i^{(1)}(x)$, the most “certain” foreground pixels are eliminated from the approximation. The resultant parabola $z_i^{(2)}(x)$ is shown in figure 3.10 (b) with a dashed line. A third iteration is carried out in the same way, this time using the set $\{x \mid gr_i(x) > z_i^{(2)}(x)\}$ to derive $z_i^{(3)}(x)$ (triangle maker in figure 3.10 (b)). It was found empirically that three iterations give a sufficiently good result.

Consider a pixel (x, y) with grey intensity $gr(x, y)$. Let the pixel be contained within the i -th horizontal stripe. To produce the *horizontal scan* we label the pixel as foreground iff

$$gr(x, y) < z_i^{(3)}(x) - T_i. \quad (3.16)$$

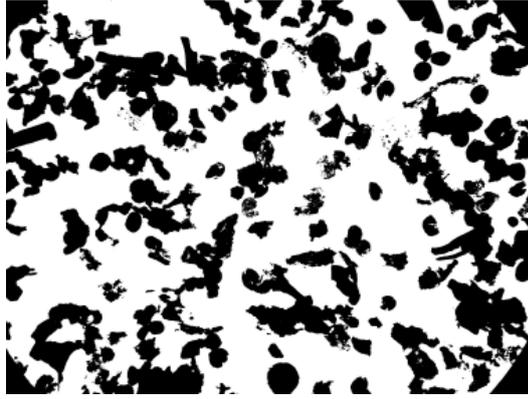


Figure 3.11: Final segmentation using the crossing stripes parabola method

Similarly we can produce a *vertical scan* by labelling pixels (x, y) found in the j -th vertical stripes as foreground iff

$$gr(x, y) < z_j^{(3)}(y) - T_j. \quad (3.17)$$

The thresholds T_i and T_j are automatically calculated as explained later. Sometimes unwanted artefacts in the form of skidmarks are present in both the horizontal and vertical scan. These are slashes of image regions incorrectly segmented. Such effects are caused by a large number of objects in the centre of the stripe. In these cases the fitted parabola would be a trough rather than a hill and cause an incorrect segmentation. It is unlikely the same effect will occur within the stripe orthogonal to this. To remove the artefacts both scans are combined to form the final segmentation. Only when a point is labelled as foreground in **both** images will its overall label will be returned as “foreground”. The label for a pixel (x, y) contained in the i -th horizontal stripe and j -th vertical stripe is foreground iff

$$gr(x, y) < \left\{ z_i^{(3)}(x) - T_i, z_j^{(3)}(y) - T_j \right\}, \quad (3.18)$$

The segmented image is obtained by labelling all pixels in the image in this way. The result of combining the scans is shown in figure 3.11.

The thresholds T_i and T_j are determined automatically from the respective parabolo-

las $z_i^{(3)}(x)$ and $z_j^{(3)}(x)$. The parabola gives the “middle” background intensity in the stripe. However, fluctuations about the curve may also belong in the background. The following heuristic threshold T_i is proposed for horizontal stripe i

$$T_i = \max_x z_i^{(3)}(x) - \text{mean}_x z_i^{(3)}(x) \quad (3.19)$$

T_j is calculated in the same way for the vertical stripes. Using the standard deviation of the points or the maximum residual are additional possibilities for constructing the thresholds.

3.4.2 Forming a background estimate

A background estimate can also be formed with a slight adaption to the CSP method. Let pixel (x, y) be contained in the i -th horizontal stripe and j -th vertical stripe. The background image $b(x, y)$ is calculated as

$$b(x, y) = \frac{1}{2} \left(z_i^{(3)}(x) + z_j^{(3)}(y) \right). \quad (3.20)$$

3.5 Evaluation and results

We ran the crossing stripes parabola method against the 5 most successful segmentation techniques for our images (global thresholding, bothat transform, fitting a 2D parabola, clustering and Lindblad’s method) and compared the results visually against the original. The comparison was conducted using 6 microscopy images. As shown in section 3.2.1, global thresholding will not work for background segmentation on images with uneven illumination. The next step was to try an adaptive thresholding technique. The best segmentation results in this category were found using the bothat transform, see section 3.2.2. However, this method was unsuitable for use in an automatic procedure as the correct sized structuring element was vital for a good segmentation. A fully automatic procedure of fitting a 2D parabola to the background illumination was suggested in section 3.2.2 but

Table 3.1: Errors on generated images shown as a percentage of misclassified pixels for Lindblad and CSP methods.

Ring radius (%)	5	10	15	20	25	30	35	40	45	50
Lindblad	1.79	2.78	3.66	5.41	9.30	12.20	18.80	28.20	6.27	4.04
CSP	0.36	1.23	1.86	2.90	4.24	7.52	13.32	16.88	17.85	6.45

the inflexibility of the model resulted in the loss of some microfossils. Clustering colour image pixels using the k-means algorithm was also a fully automatic approach and was an improvement over the 2D parabola. Nevertheless, microfossils near the edge of the image were still incorrectly segmented. Correcting the background illumination prior to applying a global threshold gave the most promising segmentation. Best results in this category were obtained by fitting a uniform cubic B-spline to the background illumination using Lindblad's method (section 3.3.2) and then thresholding by fitting three Gaussians to the image histogram (section 3.2.1). The quality of segmentation using this method closely matches that of CSP but, in certain circumstances the CSP is better than Lindblad's method.

To compare CSP against Lindblad's method we generated 10 non-uniform backgrounds of size 200 by 200 pixels. For each background a dark ring was placed in the centre as a foreground object. The CSP technique and Lindblad's method were both used to segment the ring from the image. The error was estimated by calculating the percentage of pixels misclassified by the methods.

Ten rings of constant thickness (30 pixels) with increasing inner radius were created and placed one at a time in the centre of the background. The inner radii of the circles, expressed as a percentage of the image width, were 5%, 10%, ..., 50%. The average error for both methods over all 10 images is shown in table 3.1. For images with inner radius less than 5% to 40%, the CSP method was better than Lindblad's method while at radii 45% and 50% Lindblad's method was better.

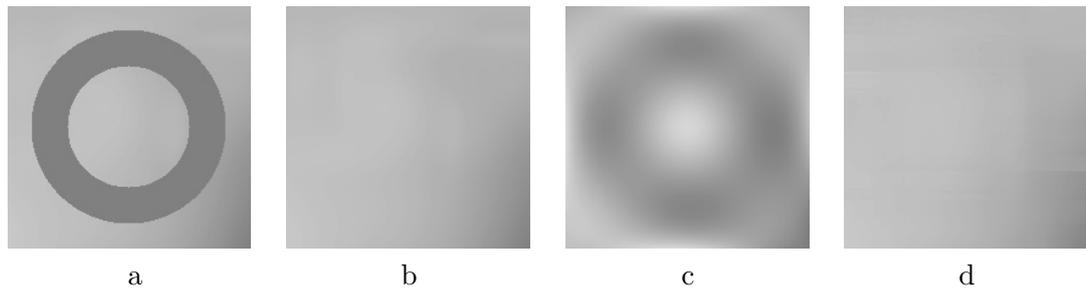


Figure 3.12: (a) Generated image. (b) The true background. (c) The background estimated by Lindblad's method. (d) The background estimated using the CSP algorithm.

Figure 3.12 demonstrates why CSP works better than Lindblad's method. Figure 3.12 (a) shows the generated image with a dark ring. Figure 3.12 (b) shows only the generated background. Figure 3.12 (c) shows the estimated background using Lindblad's method. The CSP algorithm can also estimate the background of the generated image; this is shown in figure 3.12 (d). Notice that the foreground has pulled the background estimate of the B-spline towards lower intensity values however the crossing stripes parabola ignores these low intensity values creating an adequate background estimate.

The problems associated with the most popular background segmentation techniques are solved using the crossing stripes parabola method. Background fitting using this method is flexible enough to model the background and ignore foreground information. The number of parameters that are tuned automatically within CSP far exceeds those of the standard methods and this is why a better segmentation is found. Fitting a quadratic function entails tuning 6 parameters for the coefficients of the function. Clustering in RGB space uses 27 parameters, each of the three clusters has a centre in three dimensions and an associated covariance matrix. The covariance matrix contains 9 values but due to the symmetry only 6 of these are independent. The B-spline method uses a mesh of size 5×5 as the control points for the surface, hence 25 parameters are used. The CSP method uses 3 coefficients of a parabola fitted to each mean row and column. In our example we used 15

Table 3.2: Computation times (shown in seconds) for the top 5 segmentation methods are displayed against the computation time for the proposed CSP algorithm.

Method	Image1	Image2	Image3	Image4	Image5	Image6	Average
3 Gaussians	8.16	3.94	3.17	5.87	4.32	10.13	5.93
Bothat	2.57	2.11	2.09	2.07	2.08	2.08	2.17
2D Parabola	0.42	0.15	0.14	0.14	0.14	0.14	0.19
Clustering	1.41	1.19	1.21	1.21	1.07	1.14	1.20
Lindblad	58.92	76.15	63.02	58.99	68.55	80.53	67.69
CSP	0.85	0.97	0.86	0.75	0.89	0.67	0.83

parabolas for the horizontal fit and 19 for the vertical fit, which results in 102 parameters.

The input parameters of the crossing stripes parabola method specifies the number of stripes used within the horizontal and vertical direction, K_x and K_y respectively. The accuracy of the segmentation increases as the number of stripes is increased. By decreasing the number of stripes the computational speed is decreased. For an image of size 1704 by 2268 pixels, we found that a good compromise between speed and accuracy is to set $K_x = \text{ceiling}(\text{No. Rows}/40)$ and $K_y = \text{ceiling}(\text{No. Columns}/40)$. These parameters can be calculated like this for microscopy images with resolutions up to 10 times lower than 1704 by 2268 with unobservable loss in accuracy.

The computation speed for each method was timed on 6 microscopy images of size 568 by 768 using a PC running a 2.0GHz CPU and 1GB of RAM (all methods were tested using Matlab). The results are shown in table 3.2. The segmentation offered by Lindblad's method is in most cases as accurate as the one obtained by the crossing stripes parabola method. However, the crossing strips parabola method takes a fraction of the time Lindblad's method needs.

The CSP method accounts for vignetting and central light source within its correction procedure. It is likely that CSP can be applied to any image were these type of effect occurs but particularly within microscopy or astrophotography.

Nomenclature 2

B Binary image.

M Set of all image pixels.

N_g Number of distinct grey levels.

R Set of regions.

R_i Set of pixels.

T Threshold value.

μ_b Mean grey intensity of background region.

μ_f Mean grey intensity of foreground region.

σ_b Standard deviation of grey intensity of background region.

σ_f Standard deviation of grey intensity of foreground region.

θ Proportion of background pixels.

b Structuring element.

f Digital image.

gr Grey intensity of a pixel.

i Illumination image.

p_b Probability density function of grey levels in the background region.

p_f Probability density function of grey levels in the foreground region.

r Reflectance image.

Chapter 4

Microfossil segmentation

An image is a projection from 3D to 2D, hence overlapping microfossils on the microscope slide will appear as one connected object or touching objects within the microscopy image. This is known as the *shelving* or *Holmes* effect. Furthermore, the material being analysed has arisen from biological remains. These remains are subjected to initial distress at time of deposition and subsequently altered and deformed by burial stresses and tectonic deformation. The remains are then retrieved from their current position deep in the Earth by techniques which were not designed primarily for optimum sample preservation. Kerogen in particular can be altered considerably by such stresses and the shape of these pieces is unknown prior to examination.

The next stage in our automation of microfossil analysis is to quantify the kerogen material present on a microscope slide by counting the total number of separate pieces and measuring features as discussed in chapter 1. Counting objects in an image is straightforward for disconnected objects or objects of a particular known shape. However, counting connected or overlapping objects of arbitrary shape can prove difficult.

Shape and size features maybe important when we come to classify kerogen into vitrinite and inertinite. Touching or overlapping microfossils inhibits the extraction of these parameters. Separation of these touching particles is therefore crucial in calculating an

accurate estimation of microfossil size and shape. In our case the use of an automated image analysis technique is prompted by the impracticalities of a physical separation method.

4.1 Segmenting kerogen from the background

Kerogen corresponds to the darker regions in the image. The first step is to segment these areas out as foreground. This is achieved in a similar way to segmenting all palynofacies from the slide (see section 3.4).

A background estimate is formed using the CSP method and “subtracted” from the input image as explained in section 3.4.2. The grey level histogram of the corrected image has two distinct peaks, the left peak corresponding to the kerogen material and the right peak representing everything else. A segmentation threshold is then applied, chosen to be the intensity corresponding to the minimum between the peaks. This is found by fitting two Gaussians to the grey level histogram as discussed in section 3.2.1. After thresholding, a binary image is obtained where black regions indicate the kerogen and white regions represent the background.

The image shown in figure 4.1 (a) is a typical example of a slide containing palynofacies. The result of applying the above method for locating kerogen regions is illustrated in figure 4.1 (b).

4.2 Segmentation of touching objects

Having recovered all kerogen regions using the background removal procedure, further analyses will involve separating “touching” kerogen objects. This will need to be accomplished as an automatic procedure. A pixel \mathbf{p} with coordinates (x, y) has an δ -neighbourhood consisting of the set $N_\delta(\mathbf{p}) = \{(x + 1, y), (x - 1, y), (x, y + 1), (x, y - 1), (x + 1, y - 1), (x + 1, y + 1), (x - 1, y - 1), (x - 1, y + 1)\}$. Two pixels \mathbf{p} and \mathbf{q} are δ -connected if

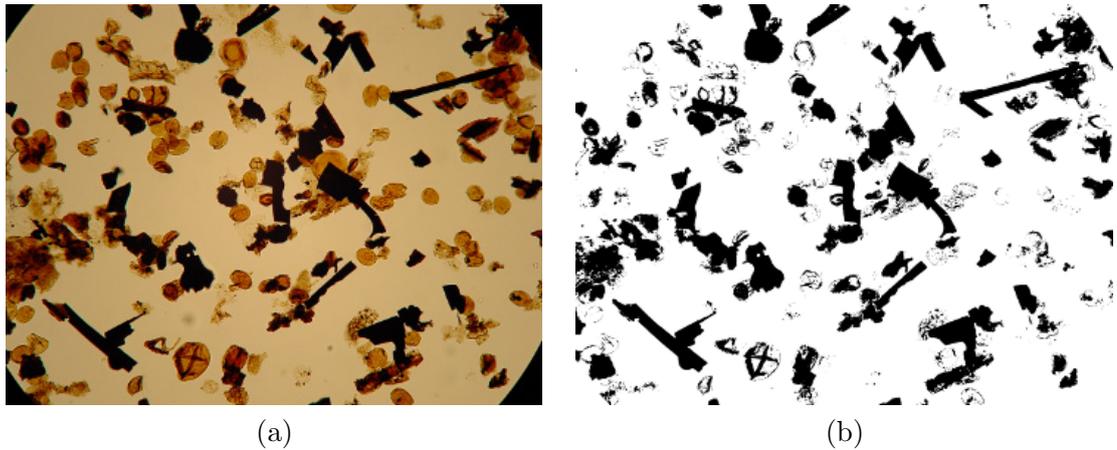


Figure 4.1: (a) Typical example slide containing palynofacies. (b) Binary image displaying automatic location of kerogen shown as black regions.

there exists a path of pixels between p and q where each pixel in the path is of the same intensity and in the 8-neighbourhood of the next pixel.

Definition 1. A set of pixels that are all 8-connected to one another is called a **connected component**.

Definition 2. We use the term **object** to define a connected component where all pixels in the connected component are from a single item of interest within the image. To clarify, an object is a connected component with meaning.

In some circumstances it can be beneficial to analyse an image using only one of the three colour channels. In the majority of images it is found that the blue channel will contain the most amount of noise. Hence, this channel could be removed. However, due to the dark colouring of kerogen in our images, edges separating the touching pieces are not visible. The only visual guide to splitting the connected regions into kerogen objects is the

shape of its silhouette. Therefore, it is only necessary to consider the binary image where kerogen regions are labelled as 0 and non-kerogen regions are labelled as 1. To split up these regions we intend to use a segmentation method based only upon this binary image.

4.2.1 Recent methods

There are methods found in the literature that attempt to separate touching particles based upon their shape. Zhang et al. (2005) separates grains by fitting ellipses using a direct least squares algorithm. However, the method was only successful if the grains themselves were elliptic. It was described by Visen et al. (2001) how ellipses with equivalent inertia are fitted to each region classed as containing grains. The ratio of region area to ellipse area is computed and a threshold value is determined to establish whether a region contains more than one grain. Once touching grains have been identified the region perimeter is analysed for corners. The grains are then split by identifying pairs of corners. The algorithm was only tested on groups containing at most 3 touching grains. It is undetermined whether this method would work for larger groups. In a similar algorithm proposed by Pla (1996) overlapping circles are separated by identifying perimeter segments based on the derivative of the curvature. Segments are then clustered using a suitable criterion. For our application we cannot assume the shape of overlapping objects; this feature will need to be assessed *after* segmentation.

An increasingly popular method of segmentation is to identify the intersection points of touching objects (van den Berg et al., 2002; Honkanen et al., 2005; Pla, 1996; Visen et al., 2001). A line of separation can then be drawn between opposite corners (van den Berg et al., 2002). For images containing many objects this approach is usually very slow to compute.

The method based upon erosion and dilation of the binary image requires the use

of structuring elements to separate touching objects (Russ, 2006; Shatadal et al., 1995). For example, we can erode the binary image until all objects have been separated but not completely removed. Dilation is then applied in order to grow the separated regions back to their original size. Restrictions are set so that while regions are growing they cannot become larger than their original size or touch other neighbouring regions. To separate touching seeds in an image Shahin and Symons (2005) used an iterative application of erosion and dilation to the binary image and then later separated them based on size and shape. This type of system is inappropriate for our images as different sized structuring elements are required to separate objects with various degree of overlapping. Also this technique is prone to deforming objects and it is even possible that some objects will be completely removed from the image if a structuring element is not chosen carefully.

4.2.2 Watershed segmentation

The watershed algorithm (Meyer, 1994) is a traditional method of segmenting touching objects. It cannot be applied to the binary image directly and it is necessary to first compute the Euclidean distance transform (EDT).

Euclidean distance transform

This transform converts a binary image into a greyscale image by assigning every pixel a value equal to the straight line (Euclidean) distance from the nearest background pixel. To calculate the Euclidean distance from all background pixels to each foreground pixel in order to determine the shortest distances would be incredibly inefficient. To speed up this process, approximations to the Euclidean distance can be used where distance is measured in only a few directions. Examples include measuring in 90° directions only, this is known as the *city block* distance. If 45° directions are included then the measure is known as the *chessboard* distance. However, results obtained using the watershed algorithm improve

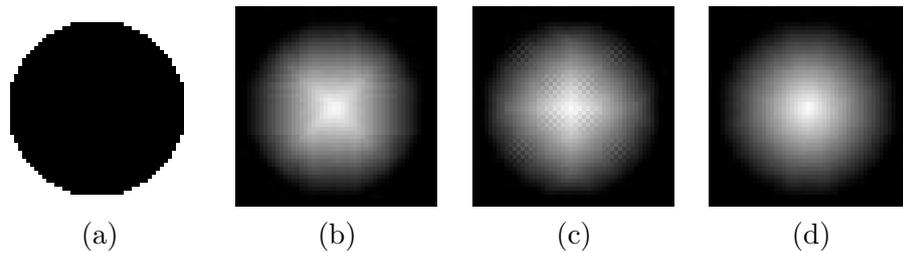


Figure 4.2: (a) Binary image, black is foreground, white is background. Distance function of (a) using cityblock, chessboard and Euclidean distances are shown in (b), (c) and (d) respectively.

as the approximation approaches the true Euclidean distance. The image processing toolbox (IPT) in Matlab uses a fast algorithm `bwdist()` to compute the exact Euclidean distances for the transform. The amount of computation to perform this operation is proportional to the number of pixels in the binary image i.e. $O(nm)$ where n and m are the number of image rows and columns respectively. The algorithm used is the second algorithm described by Breu et al. (1995). Formally we can express the Euclidean distance transform (EDT) on a binary image B as a distance function D , where

$$D(\mathbf{p}) = \text{distance from pixel } \mathbf{p} \text{ to the nearest background pixel in } B \quad (4.1)$$

The EDT can be viewed as a greyscale image by rescaling the distances, for example an 8-bit image can be formed by rescaling the distances between 0 and 255. As an example we compute the EDT of the binary image shown in figure 4.2 (a) using the cityblock, chessboard and Euclidean distances. The results are shown in figures 4.2 (b),(c) and (d) respectively¹.

¹The type of EDT used within Centre Supported Segmentation (see section 4.3) is irrelevant and results will be very similar or identical.

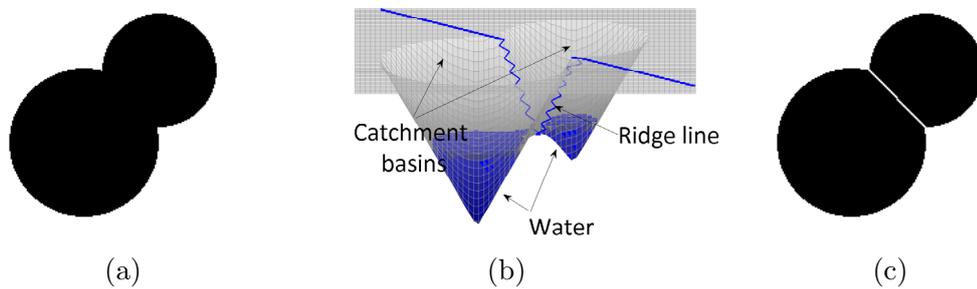


Figure 4.3: (a) Two overlapping circles. (b) Negative distance function represented by a surface. Watershed algorithm finds the watershed ridge line separating the two circles by filling the catchment basins with water. (c) Watershed ridge line is overlaid in white on top of the two circles to produce the final segmentation.

Watershed method

One can consider the EDT as a surface with the distance function $D(\mathbf{p})$ being the height for pixel \mathbf{p} . The surface representation of $-D$ for the binary image in figure 4.3 (a) is shown in figure 4.3 (b). Two troughs represent the two overlapping circles whereby the larger circle has a deeper trough.

The watershed transform is applied to the negative of the distance function $-D$. An illustration is shown in 4.3 (b). The inverted peaks can be thought of as catchment basins. Water falling onto the surface will collect within these catchment basins. As water begins to rise, a boundary line is formed where water from both pools meet. These lines are called watershed ridge lines. These lines can be overlaid on the original image in white to produce the final segmentation shown in figure 4.3 (c). We use an efficient and accurate algorithm for computing the watershed segmentation using the Matlab IPT function `watershed()` based on the algorithm by (Meyer, 1994).

The watershed algorithm works well for circular objects but is erroneous when separating concave objects. Also over-segmentation occurs with objects having irregular boundaries (especially elongated objects) and is due to many local minima forming in the

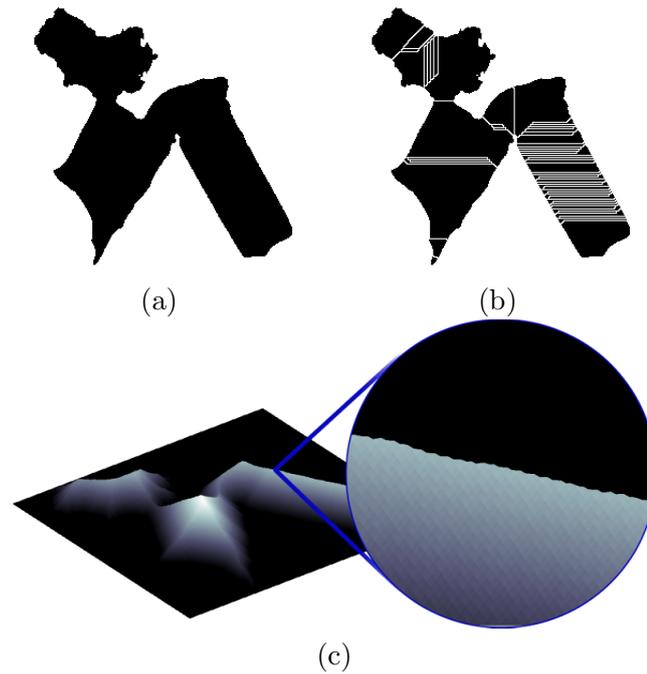


Figure 4.4: (a) Binary image displaying overlapping pieces of kerogen extracted from figure 4.1 (b). (b) Watershed segmentation resulted in 56 regions while we are looking for just three objects. (c) Visualisation of the EDT for the image in (a) depicted as a surface. A zoomed section shows the irregularities along the ridge of the EDT causing over-segmentation.

EDT. This is a big problem for overlapping kerogen as its boundaries are usually not smooth. This effect is illustrated in figure 4.4 (a) displaying overlapping pieces of kerogen (an extract from figure 4.1 (b)). The number of regions the watershed method partitions the foreground material into is 56 (shown in figure 4.4 (b)) while we are looking for just three objects. A closer look at the EDT for the image in figure 4.4 (a) can be seen in figure 4.4 (c) where a zoomed section shows many irregularities along one of the ridges.

An improvement to this approach would be to smooth the EDT prior to applying the watershed algorithm. Surface irregularities of the EDT could be removed while preserving object contours by using isotropic filtering as a smoothing procedure. However, even with this slight modification, the watershed algorithm is prone to over-segment objects. There are two main solutions: the first is to merge regions subsequent to segmentation. It

was suggested by Wahlby et al. (2004) that regions separated by a weak border should be merged. Borders are classed as weak if the mean intensity of the pixels along the border within the gradient image is less than a preset threshold. The second solution is to mark the objects in a process known as *marker controlled segmentation*. When separating pistachio nuts Casasent et al. (1996) detected the centre of each nut and used this prior knowledge as markers. In semi-automatic systems markers can be chosen manually. We found that for our images marker controlled segmentation produced excellent results when markers were chosen manually i.e. at the centre of each kerogen piece. Hence we choose to study automatic procedures for obtaining these markers.

4.2.3 Marker controlled watershed

The over-segmentation of the watershed algorithm can be reduced by finding markers for each object and only allowing water to fill from the markers position. This is known as marker-controlled watershed segmentation (Vincent, 1993; Beucher, 1992; Landini and Othman, 2003). Provided each object only contains one marker, the segmentation will be near perfect. However, too few or too many markers will result in under or over-segmentation respectively. One possible solution is to place the markers manually. However this defeats the purpose of an automatic system. An alternative to the manual approach is to identify specific features found inside individual objects (Clocksin, 2003; Lindblad et al., 2003). For example, when segmenting images of overlapping cells an ideal marker would be a single feature that presents itself in the centre of each cell. Locating the position of a cell nucleus would provide us with an excellent marker system. In the case of overlapping kerogen there is no distinct indicator locating the centre of each object and so we seek to find the centres using alternative procedures.

Extended h -maxima transform

Another possibility to identify markers is by using gray-scale morphology. The extended h -maxima transform (Soille, 2003) is an example of this approach. It is widely used in applications for separating touching objects of similar size in gray-scale images (Malpica et al., 1997; Wahlby et al., 2004). The transform can be applied to the EDT of the binary image or to the original grey-scale image itself. The method will filter out all local maxima whose heights are smaller than the pre-defined threshold h . A low value of h will result in many markers and a high value will produce only a few markers. This transform is dependent upon the scale of the binary image. For example, if an image is rescaled, different numbers of objects may be found for a fixed value of h , even though the same number of objects are present in the image. The best value for h is usually determined through evaluating the segmentation on a small sample of images by eye. This value is then fixed when segmenting the other images.

While the extended h -maxima transform has been very successful for separating touching objects of similar size in gray-scale images, this may not be the case in images containing objects of various sizes and shapes. The success and failure of this method is demonstrated using the binary image of size 300 by 200 pixels shown in figure 4.5 (a). This image contains overlapping circles and long objects. The markers obtained for $h = 2$ are shown as green regions in figure 4.5 (b). Using the green regions as markers within marker-controlled watershed segmentation correctly segments the image into 21 objects shown in figure 4.5(c). However, the outcome of segmentation is sensitive to the value of h . To demonstrate we segment the image again, this time with a slight increase in the value of h . The result displayed in figure 4.5 (d) shows the method (at $h = 4$) to have incorrectly segmented the smaller circles and long objects. The total possible range for h is between 0 and 22.2. Only 11% of this range will result in correct segmentation.

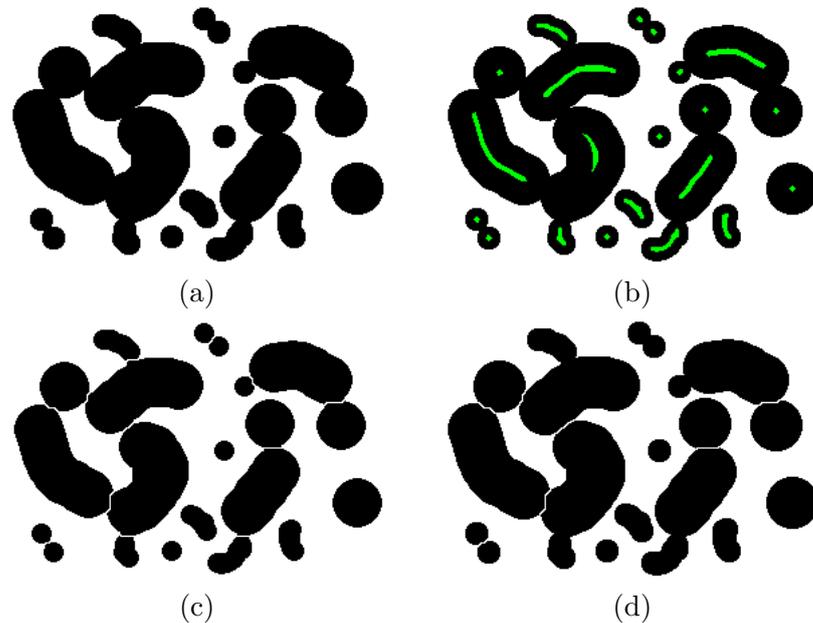


Figure 4.5: (a) Binary image containing overlapping circles and long objects. (b) Markers found by the extended h -maxima transform with $h = 2$, shown in green. (c) Segmentation using marker-controlled watershed at $h = 2$. (d) Inaccurate segmentation using marker-controlled watershed segmentation at $h = 4$.

4.3 Centre supported segmentation (CSS)

We propose an alternative segmentation method based on an intuitive, scale-independent *overlap* parameter (Charles et al., 2008a). This method will eliminate over-segmentation and successfully segment both circular and elongated objects. Centre Supported Segmentation (CSS) is based on automatic identification of a centre point for each object. CSS is applied on the black and white image where the black foreground are the objects to be segmented. The result from CSS is a set C of object centres. The centre of an object is needed for several reasons: (1) counting the number of objects, (2) viewing an object by moving the scanning camera to the centre and (3) cropping the object for further analysis and classification.

Before we begin with the CSS algorithm method, it is first necessary to define the

term *object center*.

Definition 3. Let D be the distance function on a binary image containing objects. The **centre** of an object is any pixel \mathbf{p} with the largest distance $D(\mathbf{p})$ within the object.

For example, the centre of a filled-in circle will be its geometrical centre. However, a doughnut-shaped object will have infinitely many centres, none of which will be the geometrical centre of the figure.

4.3.1 Stage 1 of CSS - locating object centres

CSS is applied to the distance function of the binary image. It first identifies the centres of all possible objects and then filters out the centres which are likely to be noise. The first stage of the algorithm creates two lists: the list C of centres and the list V of their *merging heights*.

Finding the list C

The list C is essentially a set of coordinates for the local maxima found within the distance function of the binary image. The list V is of the same size as C and for each centre $\mathbf{q} \in C$ it records the height of the saddle point between \mathbf{q} and another neighbouring centre. The neighbouring centres are chosen by the CSS algorithm. These pairs of centres are said to *merge*.

List C is constructed in such a way that the merging heights can be found in the same process. Initially C is empty. Let D be the distance function of the binary image. Suppose that $m_1 = D(\mathbf{q}_1) = \max_{\mathbf{p}} D(\mathbf{p})$ is the unique maximum of D . Pixel \mathbf{q}_1 is taken to be the first centre in C . Consider as an example the image in figure 4.6 (a). The boxes delineated by the grid are the pixels in the image. Figure 4.6 (b) displays the distance

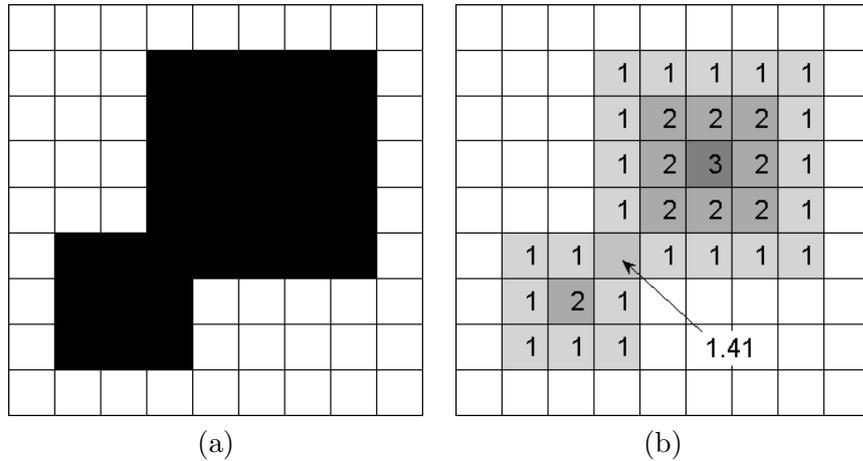


Figure 4.6: Example of a 9-by-9 binary image (a) and its distance function (b)

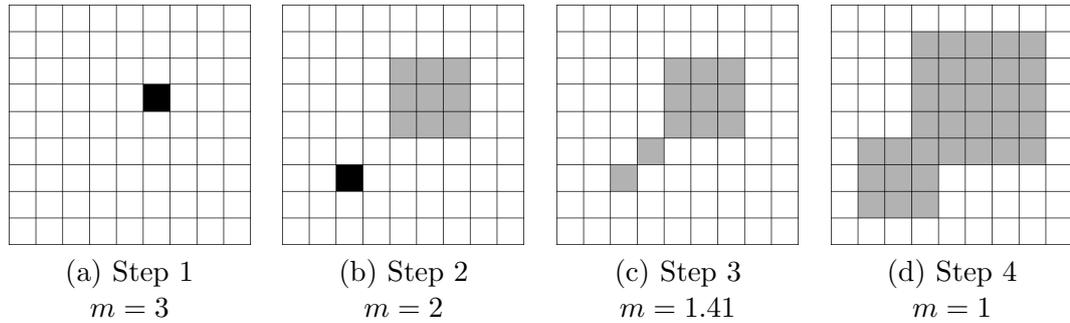


Figure 4.7: Finding the centres using CSS. Two centres will be stored in C , $\mathbf{q}_1 = (6, 2)$ at threshold $m = 3$ and $\mathbf{q}_2 = (3, 7)$ at threshold $m = 2$.

function for the image. All white pixels have $D(\mathbf{p}) = 0$. For this example, the maximum of D is 3, found at pixel $\mathbf{q}_1 = (6, 4)$. Then C is updated by $C \leftarrow C \cup \{(6, 4)\}$.

By thresholding D at m_1 , we produce a binary image B_1 so that all pixels \mathbf{p} where $D(\mathbf{p}) \geq m_1$ are set to black and the rest are set to white. For $m_1 = D(\mathbf{q}_1)$, there will be a single black dot at \mathbf{q}_1 . Figure 4.7 (a) shows this first step. The black pixel is \mathbf{q}_1 .

The next maximum height, $m_2 = \max_{\mathbf{p} \neq \mathbf{q}_1} D(\mathbf{p})$ is identified. The new black and white image B_2 resulting from the thresholding with m_2 will contain more black points than the previously thresholded image B_1 , including points around pixel \mathbf{q}_1 . Figure 4.7 (b)

displays B_2 found by thresholding the EDT D at the next lower value $m_2 = 2$.

Since the object with centre \mathbf{q}_1 is already accounted for, we remove this connected component from B_2 . The component to be removed from B_2 is coloured light grey in figure 4.7 (b). The remaining connected components each represent other local maxima within the distance function. The centres for each of these components are appended to C . Because the remaining pixels in B_2 are all found at exactly height m_2 , any pixel within a connected component can serve as a centre, only one pixel from each connected component is chosen arbitrarily. In our example only one connected component remains and $C = \{(6, 4), (3, 7)\}$.

A subsequent threshold $m_3 < m_2$ is applied to produce image B_3 from D , where m_3 is the next lower value in the distance function D after m_2 . All connected components with centres in C are then removed from B_3 in the order they were stored in C . The remaining connected components are used to find new centres, and so on. Figure 4.7 (c) shows the third step where B_3 is obtained by thresholding at $m_3 = 1.41$. There is one connected component in this image, which will be removed because it contains the first entry in the set of centres C , \mathbf{q}_1 . Finally, figure 4.7 (d) shows B_4 with $m_4 = 1$. Again, only one connected component is found and subsequently removed.

Finding the list V

To eliminate over-segmentation a parameter called *merging height* is attached to each centre and stored in the list V . Each subsequent threshold of the original binary image dilates existing connected components and may produce new ones. As thresholding continues connected components will begin to merge with other connected components. The merging height of centre \mathbf{q}_i , denoted v_i , ($v_i \leq D(\mathbf{q}_i)$), is the lowest threshold value at which \mathbf{q}_i defines a connected component disjoined from any connected components of \mathbf{q}_j

such that $D(\mathbf{q}_j) > D(\mathbf{q}_i)$. For any value lower than v_i , \mathbf{q}_i and another centre at a larger value in D will share a connected component. Figuratively speaking, the object of smaller size (smaller peak $D(\mathbf{q}_i)$) is eclipsed by an object of a bigger size ($D(\mathbf{q}_j) > D(\mathbf{q}_i)$). For the example in figures 4.6 and 4.7, the merging height of \mathbf{q}_1 is $v_1 = 0$, and the merging height of \mathbf{q}_2 is $v_2 = 2$ because this is the lowest height where the component of \mathbf{q}_2 is separate from the component containing \mathbf{q}_1 .

The list V is computed in parallel with finding the list C . Initially the merging height for a centre is set at the threshold it was discovered. At each subsequent threshold, connected components with centres in C are removed in the order they were stored in C . Let $C = (\mathbf{q}_1, \dots, \mathbf{q}_n)$. If at threshold m_k we remove a connected component with centre \mathbf{q}_i then we update the merging height $v_i = m$. On the other hand if we find that the connected component has already been removed by some centre in $(\mathbf{q}_1, \dots, \mathbf{q}_{i-1})$ then it is known that two previous connected components (initially found at higher threshold values) have merged. In this case we no longer continue updating v_i . At this instance \mathbf{q}_i is said to *merge* with the centre that *first* removed the connected component containing \mathbf{q}_i in the image thresholded at m . In our example \mathbf{q}_2 merges with \mathbf{q}_1 .

Formal description for stage 1 of CSS

We previously outlined the main steps of the CSS algorithm by walking through an example. This section formally clarifies the first stage of the algorithm in table 4.1. If we apply this process to the image in figure 4.4 (a), 56 centres will be found, each one located in its own separate region defined by the watershed algorithm. As with the watershed method, small shape irregularities on the periphery of the object will result in a jagged peak of the distance function with many local maxima of similar heights. Each little spike will generate a centre.

Table 4.1: The Centre Supported Segmentation (CSS) algorithm. Stage 1: obtaining centres C and merging heights V .

CENTRE SUPPORTED SEGMENTATION (CSS) ALGORITHM: C AND V	
1.	Given is a binary image B . Initialise $C = \emptyset$, $V = \emptyset$
2.	Find the distance function D of B and sort all the distinct distance values in descending order: $m_1 > m_2 > \dots > m_T$.
3.	For $i = 1 : T$
(a)	Find binary image B_i by thresholding D at m_i .
(b)	Find the set of all connected components $K_i = \{K_{i,1}, \dots, K_{i,k}\}$ within the image B_i .
(c)	For $j = 1 : C $ (each centre in C)
i.	find the connected component $K_{i,j}$ containing the j^{th} centre in C .
ii.	if $K_{i,j} \in K_i$, then remove $K_{i,j}$ from the set K_i and set $v_j = m_i$.
(d)	For each remaining component $K_{i,t} \in K_i$ find the centre of this component* as $\mathbf{q}_{i,t} = \arg \max_{\mathbf{p} \in K_{i,t}} D(\mathbf{p})$.
(e)	Let $c = K_i $.
(f)	Augment C and V
	$C \leftarrow C \cup \{\mathbf{q}_{i,1}, \dots, \mathbf{q}_{i,c}\}, \quad V \leftarrow V \cup \underbrace{\{m_i, \dots, m_i\}}_c$
4.	Return C and V
* Note: After removing the connected components from the set K_i in (3.c), the remaining connected components will all contain pixels at exactly the same height m_i in the distance function. This is because all possible distances are checked. These may be single points or clusters of points at the same height. Thus any point from $K_{i,t}$ can serve as the centre of the component.	

The merging heights of centres of large objects will be low even if they overlap with smaller objects. On the other hand, centres corresponding to noise at the peak will have high merging heights. The centres with large v_i relative to their height within the distance function will be candidates for elimination.

4.3.2 Stage 2 of CSS - removing redundant centres

In Stage 2 of the CSS algorithm redundant centres are eliminated. A threshold s can be applied to account for the minimum allowable size of an object. All objects with centres \mathbf{q} , such that $D(\mathbf{q}) < s$, are discarded. If the algorithm is run with $s = 0$, it will find all the specs in the image as objects of interest. Another option for removing the specs is to apply erosion and dilation directly to the binary image but, this will again require a parameter to determine the size of the structuring element which implicitly sets a value for s . Hence such an option will not eliminate the need for an extra parameter. On the other hand, the value of s can be estimated by eye or can be learned from a sample of training images where the objects of interest have been pre-labelled by hand. Weller et al. (2005) propose an empirical threshold of 14 μm .

The overlap parameter

The image in figure 4.4 (a) looks like three touching objects, however it may also be a genuine set of 56 tightly packed objects. We introduce a parameter d to determine which centres need to be removed. The degree of overlap is defined using two intersecting circles as demonstrated in figure 4.8 (a), and is measured with respect to the smaller circle. If the two circles are of the same size, either of the two can be chosen. The overlap value is defined by the two intersection points A and B . Denote by t_i the minimum distance from the mid-point of the segment AB to the edge of the smaller circle. The overlap is defined as the ratio of t_i to the radius of the circle, $D(\mathbf{q}_i)$. The length t_i is found using v_i , the merging

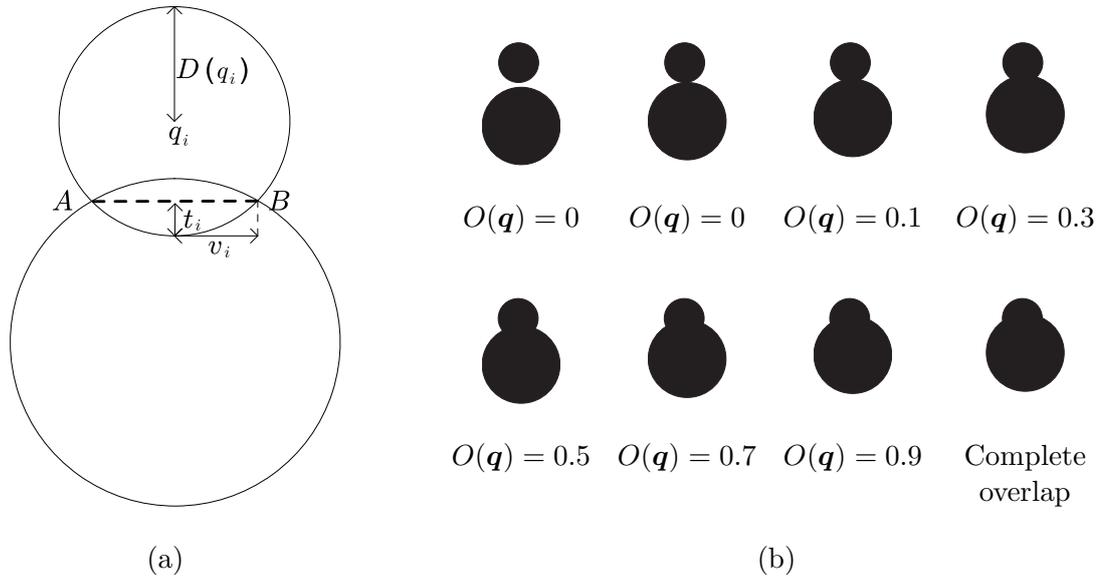


Figure 4.8: (a) Two overlapping circles define the concept of overlap. (b) Illustration for various values of overlap $O(\mathbf{q})$

height of the centre \mathbf{q}_i . The merging height is half the length of AB and is shown as v_i in the diagram. Since $t_i = D(\mathbf{q}_i) - \sqrt{D(\mathbf{q}_i)^2 - v_i^2}$, the degree of overlap for a centre \mathbf{q}_i is

$$O(\mathbf{q}_i) = 1 - \sqrt{1 - (v_i/D(\mathbf{q}_i))^2}. \quad (4.2)$$

By definition $D(\mathbf{q}_i) > v_i$ and so $O(\mathbf{q}_i) \in [0, 1)$. A centre \mathbf{q}_i with overlap $O(\mathbf{q}_i) = 0$ means that the object is isolated. As the overlap approaches 1, the object is increasingly covered by a larger item. The overlap of two circles is demonstrated in figure 4.8 (b). The small circle is increasingly covered by the larger circle. The overlap value $O(\mathbf{q})$ is also shown. As soon as the two circles merge so that $\mathbf{q}_i = t_i$, the CSS algorithm will continue to recognise one object, in this case we have complete overlap.

It is interesting to consider the effects this definition of overlap has on the results of Centre Supported Segmentation. Other possible areas of investigation could consider defining overlap value using other shapes, such as, overlapping ellipses.

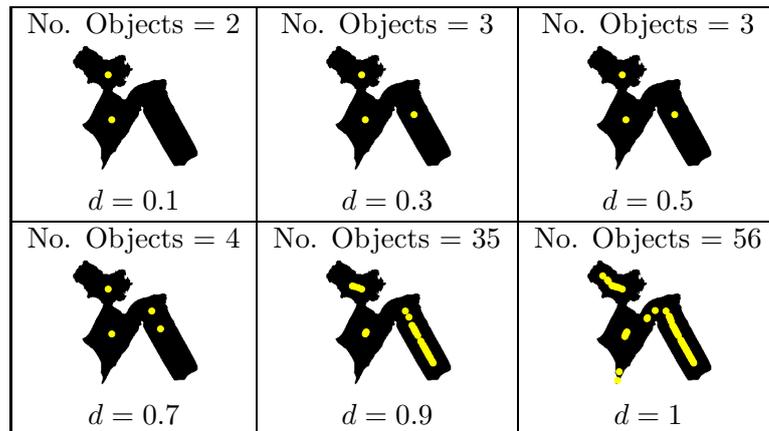


Figure 4.9: Effect of the allowable overlap d on the image segmentation. Best segmentation (3 centres) is obtained for d between 0.3 and 0.5

Filtering centres using the overlap

To remove centres from C we set a limit d on the amount of overlap such that if $O(\mathbf{q}_i) \leq d$ then centre \mathbf{q}_i is kept in C and discarded otherwise. If we wish to segment an image to its maximum detail then the threshold is set to $d = 1$. This would return the same number of segments as the watershed algorithm. At a value of $d = 0$ each connected component in the binary image is labelled as an object. By adjusting d over-segmentation can be prevented. The “noisy” centres occur due to small deformations in the shape and this will correspond to relatively large merging heights yielding large overlap values. Hence setting a threshold d not only specifies the connectivity of objects but also eliminates the “noisy” centres.

The CSS algorithm is applied to the image of 3 overlapping kerogen objects from figure 4.4 (a). The effect of setting a maximum allowable overlap can be seen in figure 4.9. At $d = 0.5$ we have a “correct” segmentation into three objects and at the limiting case of $d = 1$ we obtain the 56 segments that are produced by the watershed algorithm.

4.3.3 Segmenting the objects using centres

After filtering the centres, corresponding individual objects can be segmented. CSS belongs in the class of marker-controlled segmentation algorithms. Hence the marker-controlled watershed algorithm is applied to the negative of the distance function with centres acting as markers. To extract the objects, the watershed boundaries are overlaid in white on top of the black and white image. Thus the new binary image will consist of black connected components corresponding to objects, which can be easily extracted for further analysis.

The segmentation appears to perform best when the overlap filter $d = 0.5$. We found that a human can most easily separate two overlapping objects provided that their overlap is no greater than 0.5. Hence a chosen value of $d = 0.5$ will best describe this behaviour.

CSS was applied to the binary image of kerogen regions displayed in figure 4.1 (b) with $d = 0.5$ and $s = 4$. The centres found (shown in figure 4.10 (a)) were used as markers in marker-controlled watershed segmentation. To illustrate the segmentation of kerogen into kerogen objects, figure 4.10 (b) shows the watershed ridge lines overlaid in white on top of the original colour image. The 30 largest kerogen objects are illustrated in figure 4.11. Row 1 columns 1 through to 3 and row 2 column 1, show the only four incorrect segmentations present in this illustration.

To further improve on these results, perhaps a corner detector could be introduced to provide a confidence value in separating the objects after CSS has been applied. If a sharp corner is present at the separation line then we will be more confident CSS is correct at this point.

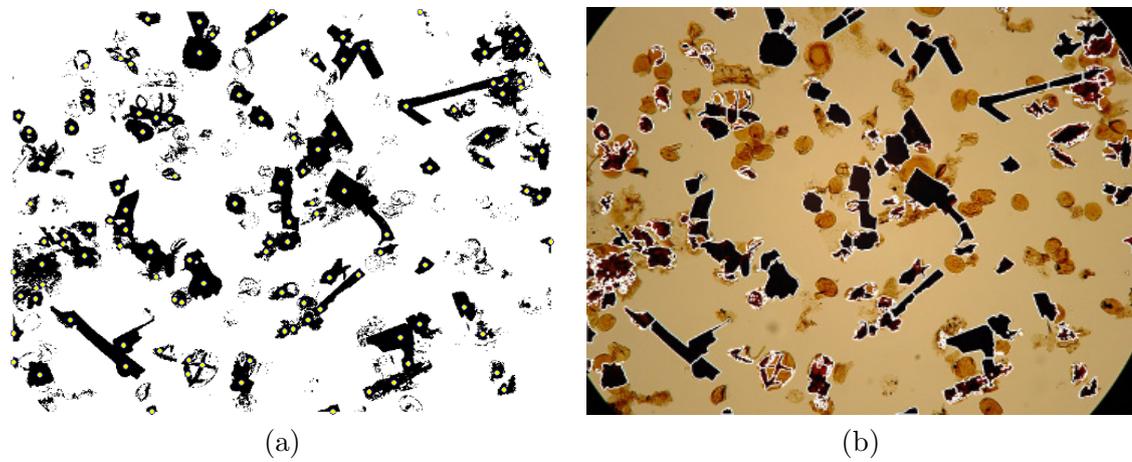


Figure 4.10: (a) Centres found by CSS are shown as yellow dots. (b) Centres used as markers in marker-controlled watershed segmentation. Final segmentation is overlaid in white on top of the original colour image.

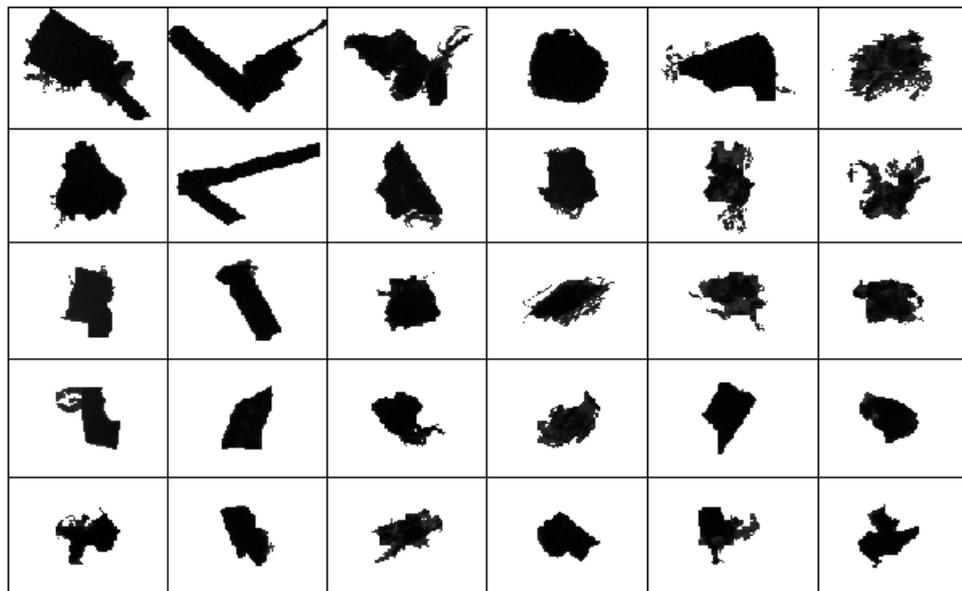


Figure 4.11: Kerogen objects extracted using marker-controlled watershed segmentation with centres acting as markers

4.4 Segmentation evaluation

In order for the quality of segmentation to be judged, a validation procedure needs to be introduced. A survey was conducted by Zhang (1996) where evaluation methods for image segmentation were categorised as analytical and empirical. *Analytical methods* examine the principles and properties of the segmentation algorithms themselves. *Empirical methods* evaluate the output of the segmentation algorithms on test images.

The quality of segmentation can be judged by goodness measures. These measures are usually defined by human perception of a “good” segmentation. They are calculated on the segmented image alone and do not require a reference segmentation. For instance, the entropy of the partitioned image, intraregion uniformity, region shape or colour uniformity, are all examples of goodness measures. Empirical methods that use goodness measures are called *goodness methods*.

The most straightforward approach to validation is by comparing the automated segmentations within a ground truth segmentation, obtained manually. The performances of segmentation algorithms can be evaluated based upon inconsistency or distance between the automated segmentation and the ground truth. These types of methods are called *discrepancy methods*. Because segmentation ground truth is based upon human perception, which is varied, evaluation of segmentation performance is difficult. However, it should be noted that even though there are inherent inconsistencies in an operator’s performance and perception, this approach is considered to produce a truth model. The precision or accuracy of segmentation can be defined according to a figure of merit. The choice of the figure of merit is dependent on the application and can be based on four types of low level discrepancy approaches presented by Beauchemin and Thomson (1997): pixel, area, point-pair and boundary.

The pixel-based discrepancy approach is the most common one and consists of counting the number of misclassified pixels in the segmentation output relative to a reference partition. Using a similar approach Cardoso and Corte-Real (2005) formulate a general measure, an important asset of which is that it is a metric. Area-based methods evaluate area of overlap between corresponding segments (Hoover et al., 1996; Borsotti et al., 1998; Liu and Yang, 1994) while boundary-based schemes compare the perimeters of the segments. The point-pair discrepancy approach measures the agreement between two segmented images (Rand, 1971) without explicitly solving the correspondence problem between the regions.

Segmentation produces a partition of the pixels and hence can be thought of as a clustering technique. Thus the discrepancy between the obtained segmentation and a ground truth segmentation can be evaluated by any measure of agreement or similarity between two partitions. Along with the Rand index, various other measures of similarity have been proposed in the literature, the most widely used being Jaccard index, adjusted Rand index, correlation, mutual information and entropy (Rand, 1971; Ben-Hur et al., 2002; Hubert and Arabie, 1985).

We propose an evaluation measure that belongs to the point-pair group within the discrepancy approach (Charles et al., 2006). Two sets of centres are formed, one set represents the ground truth segmentation and the segmentation to be evaluated is represented by the other set. A figure of merit is obtained by measuring the degree of match between these two sets of centres. The proposed measure consists of three indices evaluating different aspects of the positioning of centres.

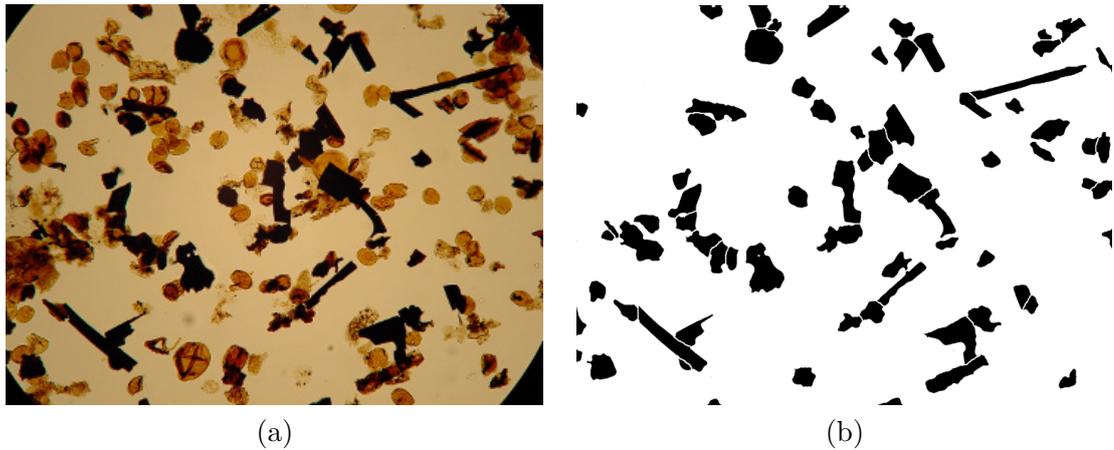


Figure 4.12: (a) Original image. (b) Ground truth segmentation of kerogen objects in (a).

4.4.1 Obtaining centres for a given segmentation

Any object of an image can be represented by a single centre as defined in Definition 3. If an item of interest in the image spans n connected components then it will be represented by n objects, one for each component. For example, if we were to segment a picture of a window then one object would represent the window frame. On the other hand, the item of interest representing the glass would consist of many separate objects, one for each pane.

Because a kerogen piece is a single complete microfossil it should be segmented into a connected component of its own and is classed as one object. To obtain the ground truth segmentation for one of our slides, the foreground kerogen is first automatically segmented from the background as in section 4.1. Next a human expert manually splits the kerogen into kerogen objects by “cutting” through the regions with a white line. The cutting procedure involves drawing white lines through the binary image using suitable image editing software. Small black regions that are deemed not to be an object are also manually removed. An example ground truth segmentation for the image in figure 4.12 (a) is displayed in figure 4.12 (b).

4.4.2 Centre-based measure

The three most desirable properties of a segmented image can be expressed as follows:

1. A perfectly segmented image exhibits no under- or over-segmentation.
2. There are no centres of objects which lie outside the objects boundaries.
3. The centre of each segment should coincide with the relevant object centre.

Let $C^* = \{\mathbf{q}_1^*, \dots, \mathbf{q}_n^*\}$ be the ground truth centres and $C = \{\mathbf{q}_1, \dots, \mathbf{q}_m\}$ be the centres of the segmentation we wish to evaluate. We propose to evaluate the quality of segmentation by the following three measures.

Definition 4. Let n_i be the number of centres placed by the automatic segmentation within object i (the object i is defined by the ground truth segmentation). The measure of **under- or over-segmentation** of i is

$$r_i = \begin{cases} 1 - 1/n_i, & \text{if } n_i > 0 \\ 1, & \text{if } n_i = 0 \end{cases} \quad (4.3)$$

If there is no centre in the object or if there are a large number of centres there, r_i approaches 1. The most desirable value of r_i is 0 which is achieved if there is only one centre in the object.

Definition 5. The measure of **background segmentation** is

$$b = 1 - \frac{1}{m} \sum_{i=1}^n n_i, \text{ where } m \text{ is the number of centres in } C. \quad (4.4)$$

Note that b is the proportion of automatic centres that are not contained within the boundaries of any objects. Thus $b = 0$ corresponds to the ideal situation where the background is free of centres placed by mistake by the segmentation algorithm.

The values r and b completely represent the under and over-segmentation of the image regardless of the location of the centres within the objects. Hence the third measure evaluates how close the approximations are to the ideal centres within the objects.

Definition 6. Let \mathbf{q}_i^* be the true centre of object i and let $\mathbf{q}' \in C$ be the nearest centre from the automatic segmentation which lies within object i , i.e.,

$$\mathbf{q}' = \min_{\mathbf{q} \in \text{object } i} \|\mathbf{q}_i^*, \mathbf{q}\| \quad (4.5)$$

The **centre discrepancy** is defined as

$$c_i = 1 - D(\mathbf{q}')/D(\mathbf{q}_i^*), \quad (4.6)$$

where $D(\mathbf{q})$ is the distance transform value for the pixel at position \mathbf{q} . By Definition 3, $D(\mathbf{q}_i^*)$ is the maximum distance within object i therefore $D(\mathbf{q}') \leq D(\mathbf{q}_i^*)$, and $c_i \in [0, 1]$

To illustrate the rationale for introducing the centre discrepancy measure c_i , consider the object in figure 4.13. The true centre for this object is marked with a cross. Two guesses for this centre have also been marked and labelled. Clearly Guess 1 is closer to the true centre but, Guess 2 sits on the next highest peak of the distance function and is a

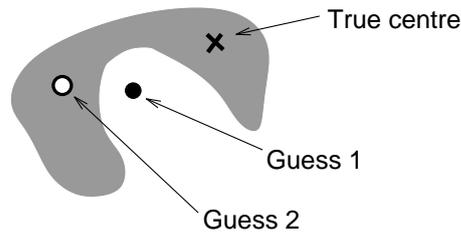


Figure 4.13: Examples of a true and two guessed centres.

much better representation of the centre for this object than Guess 1. The measure c_i will favour Guess 2 over Guess 1.

The three measures r_i , b and c_i can be combined so that the quality of the segmentation is measured by a single value. Each of the three measures is considered to be equally important and they are given the same weightings within the combined measure. Here we use a summation model for combining measures but a multiplicative model could be used.

Definition 7. The measure of **quality of segmentation** represented by the set of centres C with respect to a ground truth segmentation with a set of centres C^* is

$$S(C, C^*) = \frac{1}{3} \left(b + \frac{1}{n} \sum_{i=1}^n (r_i + c_i) \right), \quad (4.7)$$

Where $n = |C^*|$ and r_i , b and c_i are calculated as in Definitions 4-6, respectively.

The measure $S \in [0, 1)$ accounts for the three most desirable properties of a segmented image. The worst possible segmentation is at $S = 1$ and the best possible segmentation is at $S = 0$. We should note that this measure is centre based and does not directly compare the contours of objects. The next section applies this measure in an experiment to determine the quality differences between two popular segmentation methods and CSS.

4.4.3 When can we apply this measure?

Any segmentation that splits an image into objects can be evaluated using the proposed measure. This is because any segmented object can be represented by a centre as defined in definition 3. However, two important factors should be noted: 1) More than one segmentation can produce the same set of centres. 2) Some segmentation algorithms will provide labels for different objects, e.g. an image segmented into regions containing “sky”, “grass” and “mountains”. The measure S will not account for errors within the labels of different objects.

The measure S will be used in our case to assess the quality of a segmentation algorithm when splitting up foreground regions into kerogen objects. Even though the same two centres can be produced by splitting two touching kerogen pieces in multiple ways, these centres still accurately locate the objects. We are not concerned with the shape of the line that segments the kerogen and point 1 above is accounted for. The label of a kerogen object is unimportant to us at this stage, so point 2 is of no concern. Hence, the value of S will provide an accurate comparison between segmentations.

4.5 Experimental assessment

In this section we evaluate the performances of segmentation methods applied to our images.

4.5.1 Kerogen object segmentation

Six microscopy images of size 2272 by 1704 pixels were used for evaluation purposes. For each image the foreground was segmented as in section 4.1. This produces six binary images where the kerogen is black and the background is white. The performance of CSS, watershed segmentation and extended h -maxima transform was assessed according to

Table 4.2: Segmentation results for the watershed method, Centre Supported Segmentation (CSS) and extended h -maxima on 6 microscope images of palynofacies

Image no.	Object quantity	Watershed		h -maxima		CSS	
		$S(C, C^*)$	time (s)	$S(C, C^*)$	time (s)	$S(C, C^*)$	time (s)
1	73	0.43	7.37	0.33	19.10	0.14	48.87
2	72	0.58	7.32	0.45	46.65	0.22	107.07
3	82	0.60	7.59	0.48	44.10	0.29	94.21
4	14	0.55	7.31	0.39	33.58	0.13	73.05
5	77	0.53	7.39	0.41	38.89	0.22	87.61
6	25	0.50	7.30	0.38	16.89	0.11	31.90

Note: Small values of $S(C, C^*)$ indicate better match between the obtained (C) and the ideal (C^*) centres

how they segmented these six binary images. Ground truth segmentations were obtained by human expert as explained in section 4.4.1 by drawing white lines through the binary image after foreground/background segmentation and removing unwanted foreground material.

Ideal centres C^* were found for the ground truth images by locating the centre of each individual object according to definition 3. The S measure was used to evaluate the quality of segmentation; the lower the measure S the better the segmentation.

The parameter h for the extended h -maxima transform was determined by eye for a single image. This value was then fixed when segmenting other images. To decrease the processing time CSS was run on a sampled version of the binary images. Sampling was found not to adversely affect the output of CSS provided the sampled image can still be segmented into individual objects by human eye. We sampled the image every 3 pixels. Once the coordinates of centres have been found for the sampled image they were mapped back to the original image so that segmentation of objects can proceed. For the CSS algorithm, two parameters need to be set: The value of s (minimum allowed object size) was determined on a single image and then set at $s = 4$, centres are thresholded at $d = 0.5$.

The segmentation results in table 4.2 show the S measure for CSS is lower than the watershed algorithm and extended h -maxima transform for all images. This indicates a better quality of segmentation when using the CSS algorithm. As demonstrated previously the watershed algorithm will segment an image equivalent to CSS with parameters $s = 0$ and $d = 1$ resulting in over-segmentation. Therefore it is not surprising that the watershed algorithm comes in last.

Although the quality of segmentation achieved by CSS is superior to the two other methods, the running time is considerably longer. CSS is written as a script in Matlab whereas the watershed algorithm and extended h -maxima transform are built in functions. It is expected that coding CSS in a different language will make its running time more competitive. However, due to the offline nature of the application running time is not critical for the analysis.

4.5.2 CSS vs Extend h -maxima transform

The CSS algorithm outperformed the extended h -maxima transform in the experiment conducted previously. Here we intend to shed light on the reasons for this. Although the six images used in the previous experiment were exactly of the same size their resolutions were not identical. Slight changes in resolution occur due to differences in magnification levels of the microscope. The difference in resolution might have affected the segmentation results. The CSS overlap parameter d is constructed as a ratio of two lengths and is therefore scale independent. This property benefits the algorithm and CSS consistently segments objects regardless of the variability in the image resolution.

An image is constructed in figure 4.14 (a long wavy object and two overlapping circles) to demonstrate the sensitivity of the extended h -maxima transform to changes in h when segmenting a long object and circular object simultaneously. The extended h -maxima

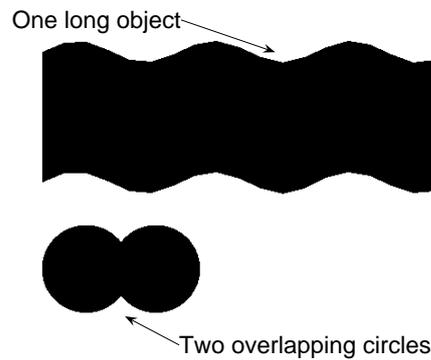


Figure 4.14: Image showing three objects, one long wave and two overlapping circles.

transform was applied to the distance function for 20 values of h spanning uniformly the whole range of possible values. Similarly the CSS algorithm was applied with the values of d spanning its range $[0, 1]$.

The graphs in figure 4.15 depict the number of segmented objects in each case. The long wave should be segmented as one object and the overlapping circles should be separated into two objects. The highlighted region shows the range in which this segmentation occurs. The extend h -maxima transform will only produce the correct results for a very small range of h . For larger values of h the circular objects disappear as no marker is placed there. The CSS algorithm will yield the correct result for a much larger range of values in d . In this experiment the extended h -maxima transform either correctly segmented the long object but under-segmented the circular ones or over-segmented the long object while correctly segmenting the circular ones.

A similar experiment was conducted with the binary image in figure 4.5 (a). The total possible range for h was between 0 and 22.2. Only 11% of this range resulted in correct segmentation. The CSS algorithm produced correct segmentation of this image when $0.37 \leq d \leq 0.64$ which amounts to 37% of the possible range for d .

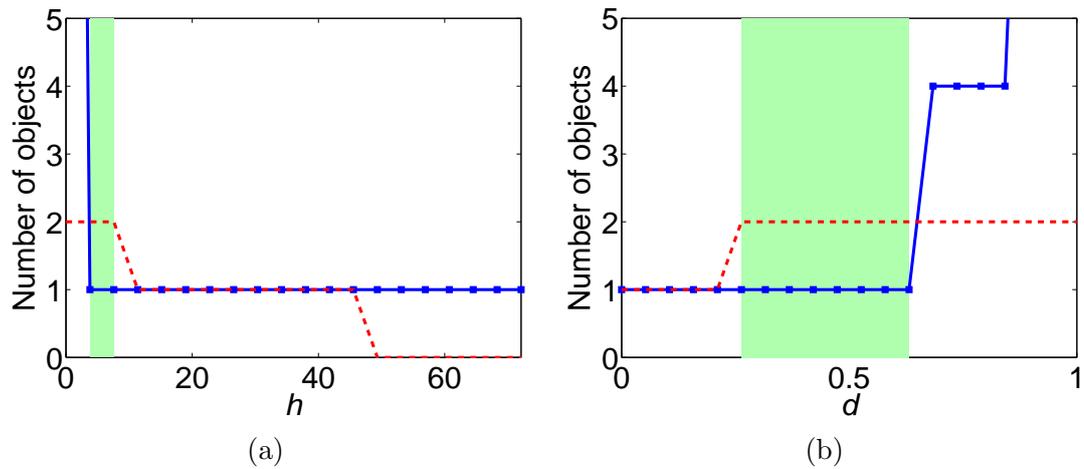


Figure 4.15: ((a) Segmentation results of extended h -maxima. (c) Segmentation results of CSS. Solid lines show the segmentation of the long object, dashed lines show the segmentation of the overlapping circles. The regions of correct segmentation are highlighted.

4.6 Summary

Foreground kerogen is segmented from the background by correcting the illumination across the image and then applying a global threshold. Because kerogen is dark in colour no edges can be seen when two kerogen objects overlap. In order for kerogen to be later classified into inertinite and vitrinite it is necessary to split these overlapping pieces into individual kerogen objects.

The CSS algorithm is applied to the binary image obtained after background removal. Centres for each kerogen object are located based upon two input parameters: The minimum allowed object size s and the overlap value d . The parameter s is chosen to comply with the physical size of the smallest particles of interest, a value for this is proposed by domain experts and used elsewhere. The overlap value d is scale independent and causes the CSS algorithm to be robust against changes in image resolution. The final segmentation is produced by using centres as markers in marker-controlled watershed segmentation.

A measure of segmentation evaluation S was defined and used to compare other

segmentation methods against CSS. It is shown that a good segmentation can be achieved using a much larger range in parameter values for CSS compared to the extended h -maxima transform method.

Application of CSS may be sought in various domains, e.g. segmenting cell nuclei and setting the initial position of active contours (Clocksin, 2003) or separating pollen grains for automated analysis (France et al., 2000). We will be using CSS to segment kerogen objects and extract each object into a sub-image of its own. The next step is to build a numerical representation for each kerogen object. Kerogen can then be further classified into two types: inertinite and vitrinite.

Nomenclature 3

B Binary image.

C Set of centres.

*C** Set of ground truth centres.

D Euclidean distance function/transform.

$N_8(\mathbf{p})$ The eight neighbourhood of a pixel \mathbf{p} .

$O(\mathbf{q})$ The overlap assigned to a centre \mathbf{q} .

S Measure of segmentation quality.

V List of merging heights found using CSS.

d The overlap threshold used within the CSS algorithm.

h Sole parameter of the extended *h*-maxima transform.

m_i Threshold applied to Euclidean distance transform.

s The minimum allowed size of an object.

Chapter 5

Kerogen classification

The visible remains of plant materials buried under high pressure and temperature are known as *kerogen*. It is formed from fossilised plant remains which are preserved through different redox (reduction-oxidation reaction) processes.

Kerogen can be classified into two types: *vitritinite* and *inertinite*, either one of these types are formed depending upon the nature of the preserved material and the conditions it has undergone (see section 2.2). Because vitritinite is less carbon rich than its partner inertinite, under transmitted light microscopy vitritinite is often seen as brown flakes. The colour is usually homogenous but can sometimes show residual cell structure. The term inertinite is used because its constituents are more inert than vitritinite. Inertinite is black and homogenous in colour and under reflected light, inertinite has a higher reflectance than vitritinite.

It is very difficult to distinguish between inertinite and vitritinite under transmitted light microscopy. Both types of maceral appear either rounded or lath shaped. Inertinite is usually lath-shaped, but slight rounding can occur. Vitritinite is usually more rounded than inertinite and cell structure can sometimes be seen, mainly towards the periphery of the microfossil. A consequence of this is a sharp distinct outline in inertinite and possibly a softer edging in vitritinite. Figure 2.2 in section 2.2 provides an example of both lath-shaped

and rounded inertinite and vitrinite pieces.

Vitrinite and inertinite can be used to study the geological events of the past. For instance, by analysing the colour of vitrinite and inertinite one can determine the maximum temperature a sample has experienced. Another important factor to consider is the distance from where the sample was taken to the source of deposition. The size and shape of vitrinite and inertinite are significant variables in predicting such a distance. Also, the main constituents of the sample site can be determined by investigating the quantity of vitrinite and inertinite found on the microscope slide. Such measurements are a vital component of oil and gas exploration (Tyson, 1998; Tyson and Follows, 2000).

The differences between inertinite and vitrinite can be seen more clearly under reflected light and this is usually how the procedure is accomplished. For our case, one of the stages in an automatic system for classifying palynofacies under transmitted light is to isolate the kerogen material. Therefore the natural progression is to classify kerogen into inertinite and vitrinite under these conditions. The advantages of developing such a system are two fold: 1) A human expert is relieved from the task of measuring inertinite and vitrinite shape and quantity. 2) The distinguishing features of inertinite and vitrinite under transmitted light can be explored. Knowledge of the salient features will help other experts (both human and machine) with classification under transmitted light conditions.

5.1 System overview

The background of the slide is removed producing a binary image where black regions contain mainly kerogen material. The CSS algorithm is applied to find centres for each kerogen object.

Input parameters for CSS include the overlap threshold value d and the allowable minimum object size s . Centres are filtered with $d = 0.5$ and $s = 4$ for our images of

size 2272 by 1704 pixels. The best setting for s can be found by trial and error on a single image and then set for subsequent images provided magnification of the microscope remains constant.

The centres found are used as markers in marker-controlled watershed segmentation where the foreground kerogen is segmented into individual kerogen objects. This stage in the system is crucial for the final automatic classification of kerogen into inertinite and vitrinite. It has been demonstrated that if careful extraction of the individual object is conducted, followed by a selection of features, even simple classification models will lead to good results (Flesche et al., 2000; Wang, 1995).

5.2 Classifiers

Let L be a set containing the possible class labels for an object. A classifier is a function whose inputs comprise of features and the output is an element from the set L . In the case of kerogen classification $L = \{\text{inertinite}, \text{vitrinite}, \text{other}\}$. The label “other” refers to objects that are neither inertinite nor vitrinite. These objects are possibly palynomorphs, amorphous material or a combination of various microfossils packed together.

Learning methods for classifiers involve reducing some form of error associated with labelling objects in the training data. If the training data is hand labelled then a measure of dissimilarity between the classifier labels and training data labels can be used as an error value. This is known as *supervised* learning. When the training data is not pre-labelled, clustering algorithms are used to find natural groupings based upon measures of similarity or agreement. Training for these methods is known as *unsupervised* learning. Because many similarity measures and cost functions can be applied, different clustering algorithms will likely result in different clusters. The number of clusters can be set before training begins or the best number can be determined through other types of measures. A

clustering algorithm known as the self-organising map (SOM) (Kohonen, 1989) was used to group images of palynological samples by Weller et al. (2006). The contour of diatom was analysed using a morphological curvature scale space by Jalba et al. (2005). Features derived from this were implemented in a K-NN clustering algorithm to identify the objects.

The classification of fossilised material has been of interest for decades. The earlier systems used rule based classifiers (Athersuch et al., 1994; Liu and Yang, 1994) but recently artificial neural networks (ANN) have become increasingly popular (Bollmann et al., 2004; Weller et al., 2005, 2006). A combination of two ANN's were used to increase the accuracy of a single ANN when identifying sedimentary organic matter in palynological preparations (Weller et al., 2007). Neural networks have also been used in other microscopy image domains, e.g., classification of pollen spores (France et al., 2000) and polyplankton species (Jonker et al., 2000; F. et al., 1999). A combination of statistical and knowledge-based approach to classification of pollen types has also been suggested by Bonton et al. (2002).

An ANN can be very sensitive to the parameters that define its construction, e.g., the number of layers and nodes in a multilayer perceptron or the type of activation functions used. Initially Weller et al. (2005) used a single ANN with sigmoidal activation function to label all microfossil on a palynological slide. Later Weller et al. (2007) found an increased accuracy could be achieved using a radial basis function ANN applied to a subset of the microfossils.

We ran experiments with 10 well known classifiers including ANN's. All 10 classifiers have been shown to achieve a good standard on a diverse range of datasets. The classifiers are as follows: Naive Bayes (Hand and Yu, 2001), Decision tree (Breiman, 1984), Logistic (Hastie et al., 2001) and Nearest Neighbour (Duda et al., 2001), these are standard (simple) classifiers. Next we use neural network classifiers: Multilayer Perceptron (MLP) (Bishop, 1995) and Support Vector Machine (SVM) (Cristianini and Shawe-Taylor, 2000).

The third group of classifiers used, known as classifier ensembles are: Adaboost (Freund and Schapire, 1997), Bagging (Breiman, 1996), LogitBoost (Friedman et al., 2000) and Random Forest (Breiman, 2001).

5.3 Training data

Kerogen objects were extracted automatically using the CSS algorithm, as described previously, from seven images of slides containing palynofacies. Each kerogen object was exported into a sub-image of its own ready for feature extraction. In total 609 kerogen objects were found. A total of 32 features were used, as specified in chapter 2, table 5.3. All features correspond to numerical values which can be used to represent an object in the form of a vector. Let $\mathbf{x} = (x_1, x_2, \dots, x_n)$, be a vector representing an object, where x_1, x_2, \dots, x_n are the values of features. In our case \mathbf{x} will be of length $n = 32$. A training dataset in the form of a table is created by cataloguing more objects in this manner. Each row of the training data corresponds to an object and the columns correspond to feature values. In this format the dataset is suitable for training an unsupervised classifier. If the objects are already labelled i.e. into vitrinite and inertinite then a supervised classifier can be trained.

The majority of the features used are standard features included in the image processing package Halcon (MVTec Software GmbH, 2008). These have been used in other palynological studies (Weller et al., 2005) as well as object recognition in general (Wang, 1995; Boucher et al., 2002). Although some features are derived from other features in the group, e.g., compactness is a scaled ratio of area against the square of the perimeter, we include them because a derived feature maybe more important than the features it is composed of.

Two features that are assumed to be important for distinguishing between the

two classes are also included: equant/lath ratio (see section 2.6.4) and rim-variability (see section 2.6.3). Inertinite is more prone to fragment into longer, thinner strips than vitrinite and the equant/lath ratio is a shape feature that measures this. In some cases the internal structure of vitrinite can be seen, mainly towards the periphery of the object. It is hoped that the feature rim-variability will detect this.

A human expert was asked to manually label each of these objects as either inertinite, vitrinite or other. Only 8 of these objects were labelled as other and the dataset is termed *imbalanced*. Therefore, we remove the objects labelled as other from our training data leaving 601 objects separated into two classes containing 236 inertinite pieces and 365 vitrinite pieces.

The final dataset is formed by standardising so that each feature has mean 0 and standard deviation 1.

5.4 Cross-validation

To determine the accuracy of a trained classifier we will use cross validation. The dataset is split into two sections called the training and testing set. We train the classifiers on the training set and then calculate its accuracy on the testing set by measuring the proportion of correctly classified samples. k -fold cross-validation partitions the dataset into k subsamples. One subsample is used for testing and the other $k - 1$ are used for training. This process is carried out k times where each of the k subsamples/folds are used exactly once for testing. The partitioning can be constructed so that each fold is stratified and contains approximately the same proportion of classes as in the whole dataset. At this point the k accuracies can then be averaged. However, to remove a bias towards the initial partitioning of the dataset the whole process is repeated n times each time using a different partitioning. The final accuracy obtained is found by averaging the nxk results. In this

study we choose to use 10-fold cross-validation 10 times.

5.5 Comparing classifiers

To compare classifiers we test the significance of the difference between their mean accuracies against one another. Let (x_1, \dots, x_N) and (y_1, \dots, y_N) be the mean accuracies from the $N = nk$ experiments for classifier X and Y , respectively. A new distribution is formed (d_1, \dots, d_N) by calculating the difference between the means of classifier X and Y for each experiment i.e. $d_i = x_i - y_i$. If we have enough experiments/samples then (d_1, \dots, d_N) has a normal distribution with mean equal to

$$\mu_d = \frac{1}{N} \left(\sum_{n=1}^N x_n - \sum_{n=1}^N y_n \right). \quad (5.1)$$

An estimate of the true variance of the differences is found by dividing the variance of the set (d_1, \dots, d_N) , σ_d^2 by N . This estimate is used within the paired t -test to determine if the two classifiers are equivalent. If this is true then the true mean of the differences will be zero, we test this null hypothesis using the t statistic

$$t = \frac{\mu_d}{\sigma_d / \sqrt{N}}. \quad (5.2)$$

The null hypothesis is rejected if μ_d falls outside the 95% confidence interval for the distribution of the differences. The limits of the interval are transformed to limits within the t -distribution (Venables et al., 1999). Hence, if t falls above or below these limits we reject the null hypothesis. This is known as the paired t -test (Demšar, 2006) and can easily be conducted in Weka¹. In a similar way we can also test to see if the mean of classifier X is greater than classifier Y and vice versa.

However, there is criticism that traditional estimates of the true variance does

¹Weka is a free software environment for machine learning and data mining. <http://www.cs.waikato.ac.nz/ml/weka/>

not account for variability in the choice of training data. Traditional estimates can lead to underestimation of the variance and an incorrect conclusion that one classifier is statistically better when it is not, e.g., increasing the number of folds will ultimately produce a significant difference because the value of t is only bound by the size of the training data. To answer this, a corrected estimate of the true variance is proposed by Nadeau and Bengio (2003) and used in Weka for performing a paired t -test. The corrected estimate of the variance is as follows:

$$\hat{\sigma}_d^2 = \left(\frac{1}{N} + \frac{N_{\text{testing}}}{N_{\text{training}}} \right) \sigma_d^2, \quad (5.3)$$

where N_{testing} and N_{training} are the number of testing and training samples respectively. Hence, the t statistic becomes

$$t = \frac{\mu_d}{\sqrt{\frac{1}{N} + \frac{N_{\text{testing}}}{N_{\text{training}}} \sigma_d^2}}. \quad (5.4)$$

5.6 Logistic classifier

Let \mathbf{x} be the object to be classified, where $\mathbf{x} = (x_1, \dots, x_n) \in \mathfrak{R}^n$. Let w_1, \dots, w_c be the class labels, $P(w_i)$ be the prior probabilities and $P(w_i|\mathbf{x})$ be the posterior probabilities for the classes, $i = 1, \dots, c$. The logistic classifier relies on the assumption that the log-odds of the posterior probabilities for any two classes can be approximated as a linear function. Without loss of generality, we can pick class w_c and fix its discriminant function to be $g_c(\mathbf{x}) = 0$ for any \mathbf{x} . The remaining $c - 1$ discriminant functions are calculated as

$$\begin{aligned} g_i(\mathbf{x}) &= \log \frac{P(w_i|\mathbf{x})}{P(w_c|\mathbf{x})} \\ &= \beta_{i0} + \sum_{j=1}^n \beta_{ij} x_j, \quad i = 1, \dots, c - 1, \end{aligned}$$

where β_{ij} , are the coefficients obtained through training the classifier. The training is done by the *Iterative Reweighted Least Squares (IRLS)* method using the *Newton-Raphson*

updates (Bishop, 2006).

For a two class classification problem let $g_2(\mathbf{x}) = 0$, without loss of generality. The only discriminant function required is $g_1(\mathbf{x})$ and this forms a hyperplane which splits the space of features into two areas, one above and below the hyperplane. Each area corresponds to a class label. An estimate of the posterior probability $P(w_1|\mathbf{x})$ for labelling an object as class w_1 is $\frac{1}{1+\exp(-g_1(\mathbf{x}))}$. Hence the posterior probability for labelling the same object as class w_2 is $P(w_2|\mathbf{x}) = 1 - P(w_1|\mathbf{x})$ and is estimated as $\frac{\exp(-g_1(\mathbf{x}))}{1+\exp(-g_1(\mathbf{x}))}$. These posterior probabilities can be used as certainties of classification. For example, a higher degree of certainty can be given to the classification of \mathbf{x} into class w_1 if the posterior probability $P(w_1|\mathbf{x})$ is near 1 rather than just above 0.5.

5.7 Classification experiment using all features

We used the Weka (Witten and Frank, 2005) implementation of the 10 classifiers in table 5.1 using their default parameter settings. The dataset for kerogen objects obtained in section 5.3 was used and the accuracy for each classifier was assessed using 10-fold cross-validation (Kuncheva et al., 2008). Each fold contains approximately 60 objects that are used for determining the accuracy. This was repeated 10 times, therefore the accuracy is found as an average of 100 results. The average classification accuracies for each classifier are shown in table 5.1 along with their standard deviation.

The logistic classifier has the highest accuracy and the lowest standard deviation and appears to perform best among the other classifiers. All classifiers are tested against the logistic classifier using the paired t -test. The classifiers which were significantly worse than the Logistic classifier according to the paired t -test are marked by a bullet in Table 5.1.²

²We note that the significance cannot be re-confirmed from the accuracies and the standard deviations

Table 5.1: Accuracy of the ten classifiers for the kerogen data set (10 times 10-fold cross-validation).

Type	Classifier	Accuracy[%] \pm std	
		All features	Top 6 features
Standard classifiers	Naive Bayes (Hand and Yu, 2001)	81.38 \pm 5.36 •	85.11 \pm 4.36 •
	Decision tree (Breiman, 1984)	84.01 \pm 4.53 •	86.94 \pm 4.20 •
	Logistic* (Hastie et al., 2001)	89.07 \pm 3.44	90.62 \pm 3.87
	Nearest Neighbour (Duda et al., 2001)	82.58 \pm 4.41 •	85.84 \pm 4.38 •
Neural networks	MLP (Bishop, 1995)	87.95 \pm 4.07	89.29 \pm 3.66
	SVM (Cristianini and Shawe-Taylor, 2000)	88.17 \pm 4.26	90.07 \pm 3.95
Classifier ensembles	AdaBoost (Freund and Schapire, 1997)	84.51 \pm 4.59 •	85.09 \pm 4.44 •
	Bagging (Breiman, 1996)	87.32 \pm 4.47	88.06 \pm 4.29
	LogitBoost (Friedman et al., 2000)	85.96 \pm 4.38 •	87.14 \pm 4.21 •
	Random Forest (Breiman, 2001)	86.71 \pm 4.57	88.07 \pm 4.15 •

*Chosen as the base for comparison

- The classifier is significantly worse than the chosen classifier

To reinforce the finding that the Logistic classifier is the best for our data, we counted the Win/Draw/Loss score for each classifier. For a given classifier X , Win is the number of classifiers significantly worse than X in the paired t -tests, Loss is the number of classifiers significantly better than X , and Draw is the remaining number of classifiers where no significant difference has been detected. A measure of total performance of X is therefore Total = Win–Loss. Table 5.2 shows the results for the ten classifiers in this study sorted by the Total measure. The Logistic classifier is again the best one among the selected 10 classifiers.

5.8 Feature selection

It is commonly known that an improved performance for a classifier can be obtained using a subset of the original features by removing the “noisy” ones. Not only

shown in the table because the t -tests were *paired* across the 100 testing results.

Table 5.2: Number of statistically significant Wins, Losses and Wins–Losses for the 10 classifiers on the kerogen data

Classifier	Total score Win–Loss	Win	Loss
Logistic	5	5	0
Bagging	4	4	0
SVM	4	4	0
MLP	4	4	0
Random Forest	2	2	0
LogitBoost	1	2	1
AdaBoost	–3	1	4
Decision tree	–4	0	4
Nearest neighbour	–6	0	6
Naïve Bayes	–7	0	7

does this lower the dimensionality of the data and speed up the learning process of some classifiers, but it can also result in better classification accuracy.

The accuracy of a classifier trained using a subset of features is used to assess the importance of this subset. Any classifier can be used but, we have chosen to use the logistic classifier.

A greedy stepwise approach to feature selection can be implemented directly in Weka. A single feature producing the best accuracy score is selected first. This feature is ranked 1. Out of the remaining features the one that produces the highest accuracy when paired with the first feature is selected second and given a rank 2. It should be noted that a negative increase in accuracy can occur when another feature is added; in this case the feature with the smallest decrease in accuracy is chosen. This process continues until there are no remaining features. The last feature chosen in our experiments will be ranked 32. Feature selection in this manner is repeated 10 times, one run for each fold of the 10-fold cross-validation. By averaging the 10 ranks for each feature, obtained from the 10 folds, we have a measure of “importance” for that feature.

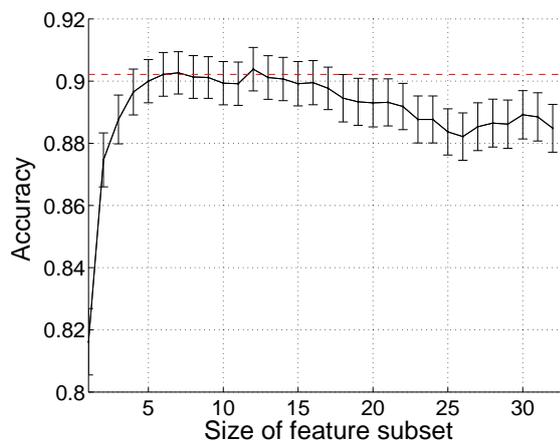


Figure 5.1: Classification accuracy versus size of the selected feature subset. The bars indicate the 95% confidence intervals.

To evaluate the contribution of each feature, we split the data randomly into 90% for training and 10% for testing and repeated the experiment 100 times. Figure 5.1 displays the averaged (testing) classification accuracy of the logistic classifier as a function of the number of selected features, chosen in order of their ranks (smallest rank first). A horizontal line is placed at 6 features to show that beyond that point the accuracy levels off and then declines, which signifies overtraining of the classifier. Therefore we propose to proceed with the top 6 features: mean red, mean blue, entropy, inner radius, rectangularity and diameter. Table 5.3 display these features according to their ranks. Surprisingly the features thought to improve accuracy of classification, equant/lath ratio and rim variability, were often picked towards the end of the feature selection procedure, bestowing upon them a poor ranking.

The discriminant function $g(\mathbf{x}) = \beta_0 + \sum_{j=1}^6 \beta_j x_j$ for an inertinite object is found by training the logistic classifier on all 601 objects using the top 6 features, where x_1, \dots, x_6 are the features and β_1, \dots, β_6 are the weights associated with each feature. A positive weight implies that the associated feature increases the posterior probability of an inertinite object. A negative weight decreases the posterior probability of an inertinite object. We can use this to form a descriptive discrimination between an inertinite object and vitrinite

Table 5.3: Features ranked by the greedy stepwise selection procedure with the logistic classifier (average of 10-fold cross-validation).

Feature	Accuracy \pm std	Rank \pm std
1 mean red	0.816 \pm 0.055	1.20 \pm 0.60
2 mean blue	0.875 \pm 0.044	2.00 \pm 0.00
3 entropy	0.888 \pm 0.040	2.80 \pm 0.60
4 inner radius	0.897 \pm 0.038	4.80 \pm 1.60
5 rectangularity	0.900 \pm 0.036	9.60 \pm 4.22
6 diameter	0.902 \pm 0.036	10.30 \pm 4.75

object under transmitted light microscopy.

Using the weights shown in table 5.4, we can firstly deduce that an inertinite object is more blue than a vitrinite object, which should appear more red in colour. Secondly the size features of an inertinite object both have negative weights, indicating that larger objects are more likely to be vitrinite; a conclusion which follows with our original understanding that inertinite is more easily fragmented than vitrinite. Thirdly an inertinite object should appear less rectangular than a vitrinite object; this is surprising because as inertinite becomes more fragmented its appearance becomes more lath shaped with strong angular edges. Finally, the feature entropy characterises the texture of the object, the more entropy the more rough the object will appear. This feature has a negative weight and we can conclude that as entropy for an object increases the chances of it being inertinite decreases. Therefore, inertinite is smoother in appearance to vitrinite; a fact we have already discussed in a previous chapter where it was stated that the internal structure of vitrinite can be seen, mainly towards the periphery of the object. Two prototypes from each class are displayed in table 5.5. These objects were selected as the closest to the respective class means using the top 6 (standardised) features.

Classification accuracy using the top 6 features for each classifier is shown in table 5.1. Again the logistic classifier gains the highest average accuracy above all other classifiers

Table 5.4: Weights associated with features for the discriminant function of an inertinite object found using logistic regression.

Feature	Weights β_j
Mean red	-3.0330
Mean blue	1.9050
Entropy	-2.0160
Inner radius	-0.5118
Diameter	-0.3873
Rectangularity	-0.3388

Table 5.5: Inertinite and vitrinite prototypes including their top 6 feature values (standardised across all 601 objects).

Inertinite		Vitrinite	
			
Mean red:	-1.0869	Mean red:	-0.2482
Mean blue:	0.4287	Mean blue:	-0.4274
Entropy:	-0.6890	Entropy:	0.4284
Inner radius:	0.7438	Inner radius:	-0.2319
Diameter:	0.4179	Diameter:	0.0101
Rectangularity:	0.1626	Rectangularity:	-0.4043

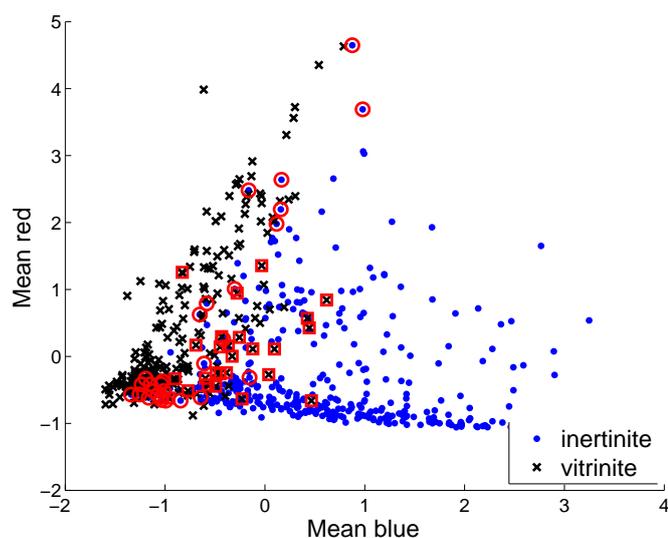


Figure 5.2: A plot showing the results of the logistic classifier when visualised using the top 2 features. Miss-classifications are highlighted with red circles and squares.

after 10 times 10-fold cross-validation. The paired t -test shows that it is significantly better than the majority of the other classifiers. We have chosen to use the logistic classifier in our system for labelling kerogen objects due to its high accuracy and simplicity. It is interesting to note that the resubstitution error of the logistic classifier trained on the top 6 features is 91.01%. This is not much different to the estimated accuracy using 10 times 10-fold cross-validation.

The results of the logistic classifier trained and tested on all 601 kerogen objects using the top 6 features is illustrated in figure 5.2. Each object is plotted as a point projected onto the ‘mean red’, ‘mean blue’ axes. If the logistic classifier has identified an object as inertinite then it is plotted as a blue dot, otherwise it is classified as vitrinite and plotted as a black cross. Miss-classifications of inertinite are highlighted by red circles and miss-classifications of vitrinite are highlighted by red squares.

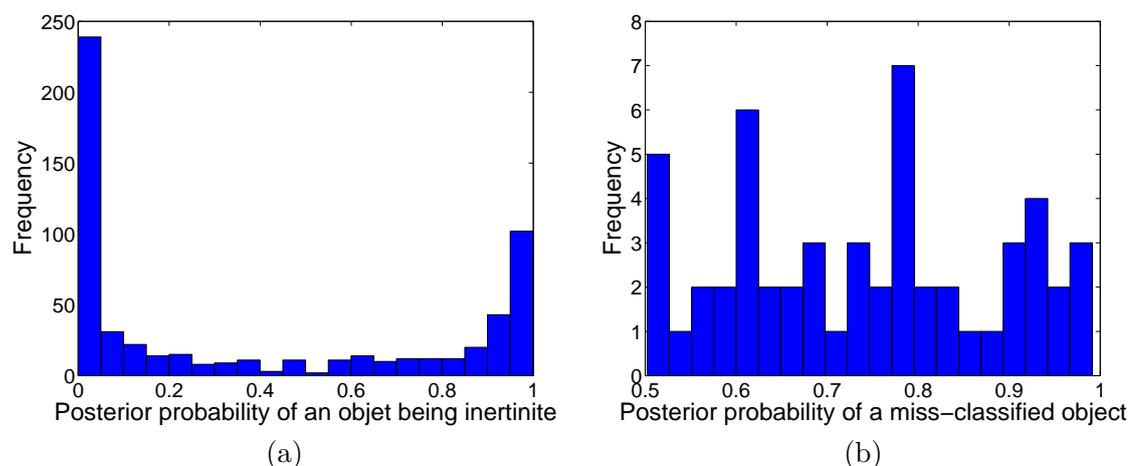


Figure 5.3: (a) Histogram displaying the posterior probabilities of a kerogen object being labelled as inertinite. (b) Histogram displaying the posterior probability of labelling a miss-classified object.

5.9 Further analysis

The posterior probabilities for labelling an object as inertinite using the logistic classifier are shown as a histogram in figure 5.3(a). The shape of the histogram indicates that the majority of objects are classified with a high certainty.

Allowing the classifier to opt out of making a decision if the posterior probability for an assigned class is lower than a certainty threshold may help decrease the number of miss-classifications. The posterior probabilities of the 54 miss-classified objects is shown as a histogram in figure 5.3(b). Notice that the probabilities span the whole interval between 0.5 and 1. This indicates that a certainty threshold is unlikely to improve the system's accuracy.

Out of the 54 miss-classified objects, 28 were miss-classified as inertinite. A second expert confirmed that 9 objects out of this 28 were correctly labelled by our classifier. The second expert also agreed that 13 of the 26 objects miss-classified as vitrinite were in fact correctly labelled, but an additional 7 objects should be labelled as class "other". In total

Table 5.6: Table showing classifier accuracy on objects where both experts agreed on labels.

Agreed label	Vitrinite	Inertinite
Classifier Accuracy	91.1%	95.2%

the second expert agreed with the classifiers supposedly wrong decision 22 times out of 54 (41%).

From this we can conclude that the ground truth obtained is *expert-specific*. It was highlighted by Athersuch et al. (1994) that fossil identification is often a personal science. A classifier trained from the opinion of one expert can only ever be as good as that expert. Therefore, when designing a system for fossil identification a consensus between experts should be considered to form a ground truth. For our system, the fact that the two experts disagree means that the definition between vitrinite and inertinite is not clear-cut (under transmitted light microscopy) and the classifier should not be penalised for making decisions between borderline cases.

The accuracy of the trained classifier against objects where both experts agreed on labels is listed in table 5.6. The high accuracies give us confidence that classification with the logistic classifier will be agreed upon by more than one expert.

5.10 Classification stability

The classification label automatically assigned to a kerogen object does not depend solely upon the classification stage of the system. It is also important to assess the sensitivity of classification accuracy to changes in previous stages. In this part of the chapter we intend to analyse the accuracy of the logistic classifier when alterations are made within the kerogen segmentation stage (Charles et al., 2009).

The CSS algorithm (see chapter 4) depends upon two parameters, s and d . Objects

below a certain size are removed from the system before entering the classification stage by using the threshold parameter s . If we wish to allow changes in the microscope resolution then s is scaled accordingly, otherwise s remains constant. Hence, s is assumed to be a fixed “component” of the system. The only remaining variable is the overlap value d , which is used to split objects based upon their degree of overlap.

Before we begin to assess the stability of classification accuracy with regard to alterations in d , it is necessary for us to first investigate the mechanics behind segmenting kerogen objects using CSS. This requires a deeper understanding of the formation of centres.

5.10.1 Hierarchical structure of centres

CSS outputs a list C containing all possible centres of objects and a list V containing their corresponding merging heights. After some objects have been removed according to the value of s another filter is applied which removes centres that have a degree of overlap greater than d . The formation of centres with respect to their overlap value is best visualised in the form of a hierarchical tree.

Merging centres

During the CSS algorithm a decreasing sequence of thresholds m_i are applied to the distance function D . Threshold m_i generates a black and white image B_i . Let K_i be the set of connected components in B_i . Lowering the threshold causes new connected components to form i.e. centres, and other connected components to merge. A new definition describes the behaviour of centres as their corresponding connected components begin to touch/merge:

Definition 8. Two centres \mathbf{p} and \mathbf{q} merge if there exists a pair of consecutive thresholds m_{i-1} and m_i such that

- 1). At threshold m_{i-1} , \mathbf{p} and \mathbf{q} are contained within two different connected components in K_{i-1} .
- 2). $D(\mathbf{p})$ is the largest distance function value within the connected component in K_{i-1} containing \mathbf{p} .
- 3). $D(\mathbf{q})$ is the largest distance function value within the connected component in K_{i-1} containing \mathbf{q} .
- 4). At threshold m_i , \mathbf{p} and \mathbf{q} appear in the same connected component of K_i .

To illustrate the concept of merging centres, consider the binary image in figure 5.4 (a). The CSS algorithm will first store centre 1, the corresponding connected component will grow until centre 2 appears. Next centres 3 and 4 will appear as the third and fourth connected components. As the threshold on the distance function m_i is lowered, the connected components with centres 2 and 3 join into a single connected component. This means that centres 2 and 3 merge (definition 8). At this instant centre 3 will receive its degree of overlap, $d_3 = 0.4453$. Finally, the connected component containing centres 2 and 3 will join that of centre 1, making centres 1 and 2 merge and giving centre 2 a degree of overlap $d_2 = 0.1150$. Note that centre 3 does not merge with centre 1 because it violates one of conditions 2 or 3 of the definition. For centres 1 and 4, $d_1 = 0$ and $d_4 = 0$. To summarise: centres 2 and 3 merge then centres 1 and 2 merge.

Hierarchical tree of centres

A hierarchical tree for each centre with zero overlap can be formed that describes the behaviour of merging centres with respect to their overlap value. Each layer of the tree represents a degree of overlap, where higher layers mean larger overlap values. First, we

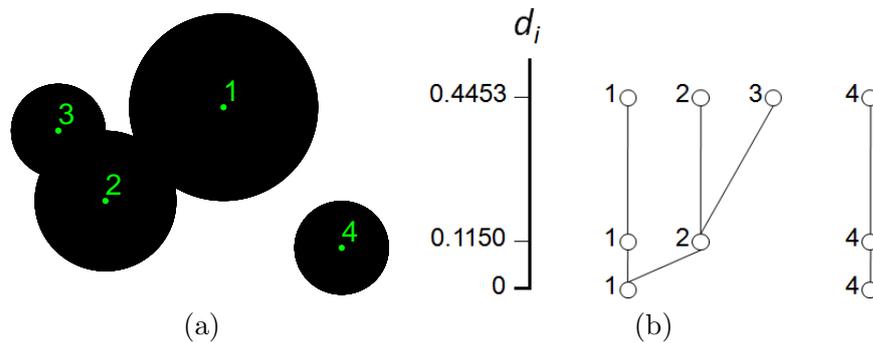


Figure 5.4: (a) An artificial image with 4 overlapping circular objects. (b) Two hierarchical trees of centres for the two connected components in (a). Nodes have been annotated with the number of each centre

take $K + 1$ samples of the overlap range $[0, 1)$ which must include the value 0. Let the sequence of sorted sample values be denoted by $d_0 = 0, d_1, \dots, d_n$. The centres with $d_0 = 0$ are taken to be root nodes of trees. All trees will be built to the same height, d_n . A layer at height d_i contains all centres, that also act as nodes, from the set C_i , where

$$C_i = \{\mathbf{p} \in C \mid \text{overlap value of } \mathbf{p}, O(\mathbf{p}) \leq d_i\}. \quad (5.5)$$

We note that $C_i \subseteq C_{i+1} \subseteq C$, where C is the set of all centres (found before applying the overlap threshold). A branch between two nodes $\mathbf{p} \in C_{i-1}$ and $\mathbf{q} \in C_i$ is drawn if \mathbf{p} is a *parent* of \mathbf{q} . This leads us to define the term *parent*:

Definition 9. Node $\mathbf{p} \in C_{i-1}$ is a **parent** of node $\mathbf{q} \in C_i$ if either

- 1.) $\mathbf{p} = \mathbf{q}$, or
- 2.) \mathbf{p} and \mathbf{q} merge and $\mathbf{q} \notin C_{i-1}$.

Some layers will contain exactly the same centres as in other layers. This depends upon the sampled values $d_0 = 0, d_1, \dots, d_n$. Including repeated layers in the tree will add no new relevant information. A tree where every layer contains a different set of centres can be formed when $d_0 = 0, d_1, \dots, d_n$ corresponds to the exact overlap of centres in C , with

no repetitions. A tree produced in this way will guarantee no loss of information about the formation of centres.

The construction of such a tree is illustrated using our example image in figure 5.4 and is formed using 3 samples from the overlap range $[0, 1)$: $d_0 = 0$, $d_1 = 0.1150$, $d_2 = 0.4453$. These values are chosen because they correspond to the exact overlap value of the centres in C for our image. Branches between nodes in the tree are drawn according to definition 9.

The final tree is displayed in 5.4 and can be used to visualise the result of filtering centres at certain overlap values. The centres found as a result of thresholding overlap at d are all located in the layer at a height equal to d or the nearest layer below d . For example, if the overlap threshold d is set at 0.3, our image will be segmented into three objects, so that objects 2 and 3 will be taken as one. For overlap threshold $0.4453 \leq d \leq 1$ four distinct objects will be detected.

5.10.2 Label inheritance trees

Varying the overlap value of d alters the number of centres produced by CSS and hence the corresponding segmentation. Thresholding overlap at $d = 0$ will result in a centre for each connected component of the original binary image. These connected components will be split into an increasing number of objects as the value of d is raised. Similarly, connected components merge with one another as the value of d decreases.

To assess the accuracy of a classifier for different segmentations using CSS, all objects found for various values of d must have labels that are considered to be ground truth. In practice the task of labelling each of the objects is laborious and time consuming. This is especially so in the case of kerogen classification where the spare time of a human expert is limited, and the number of segmented objects at high values of d for a single

image can exceed 250. We propose estimating the ground truth label of objects extracted at various values of d using the hierarchical tree of centres.

To carry out a classification experiment for different values of d one requires a mapping $f_i : C_i \rightarrow L$, where L is the set of possible class labels. Provided there exists one single layer of the tree at height d_k that has been labelled by a human expert, then the mapping f_k is known. We estimate the mapping f_j for all nodes on layer C_j as follows:

if $j > k$

Let $q \in C_k$ be a node connected to $p \in C_j$. Then $f_j(p) = f_k(q)$.

if $j < k$

We form a set $Q = \{q \in C_k \mid q \text{ is connected to } p \in C_j\}$. The nodes in Q represent objects. We calculate the areas of these objects and attach them to the corresponding nodes in Q . Then $f_j(p) = \text{weighted vote of set } \{f_k(q) \mid q \in Q\}$, where the areas of the objects are used as weights.

For example, suppose that an object labelled as vitrinite at an overlap threshold $d = 0.5$ is split up into smaller pieces by increasing the overlap threshold. According to the adopted procedure, each small piece will be also labelled as vitrinite. On the other hand, consider two objects - a large piece of inertinite and a small piece of vitrinite with a high overlap degree. At some value of the overlap parameter $d < 0.5$ these two objects will be merged into one. Since the inertinite piece occupies more area than the vitrinite piece, the weighted vote will label the combined object as inertinite.

To illustrate the use of a label inheritance tree we utilise the binary image in figure 5.5(a), displaying three overlapping pieces of kerogen. These three pieces can be segmented successfully using CSS with the overlap threshold set at $d = 0.5$. Next, a human expert manually labels them as either vitrinite or inertinite. For illustration purposes we

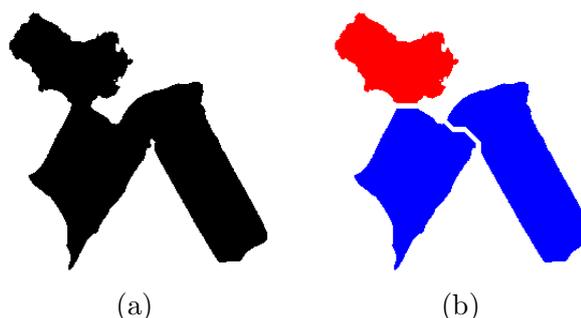


Figure 5.5: (a) Binary image displaying three overlapping pieces of kerogen. (b) Kerogen successfully segmented using CSS at $d = 0.5$ and then labelled by human expert. Red and blue represents a vitrinite and inertinite object, respectively.

have shown the segmentation and labelling in figure 5.5(b) where the object labelled as vitrinite is in red and the two inertinite objects are blue (this is the ground truth labelling).

A hierarchical tree of centres is constructed for the binary image in figure 5.5(a) with a sampling of $d_0 = 0, d_1 = 0.1, d_2 = 0.2, \dots, d_{10} = 0.9, d_{11} = 1$. For clarity we reduce the number of nodes in the tree by obtaining centres on a scaled down version of this image. These centres are then mapped back to the original image. The tree is shown in figure 5.6. The layer of the tree at height $d_6 = 0.5$ has been expertly labelled and highlighted in a yellow box, we call this the *known* layer. The nodes have been coloured red if vitrinite and blue if inertinite. The labels for objects in other layers of the tree have been inferred from the known layer as explained previously. Figure 5.7 shows the segmentations associated with layers $d_0 = 0, d_1 = 0.1, d_4 = 0.3$ and $d_{10} = 0.9$. Each segmentation has been coloured according to the inherited label.

Various trees can be constructed by altering the samplings $d_0 = 0, d_1, \dots, d_n$. Suppose each one of these trees has an identical known layer, containing the same set of centres and labels. It is interesting to note that any non-labelled object, common in each tree, will inherit exactly the same label regardless of what tree is used.

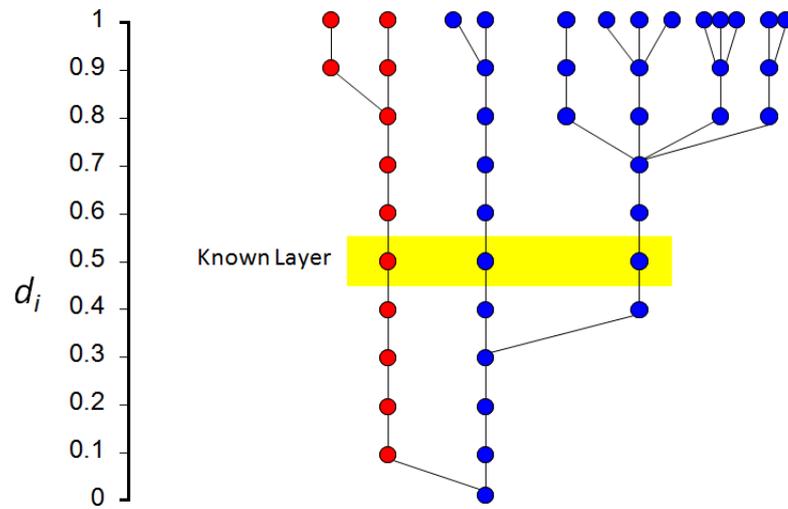


Figure 5.6: Label inheritance tree for binary image in figure 5.5(a). Red nodes represent vitrinite and blue nodes represent inertinite.

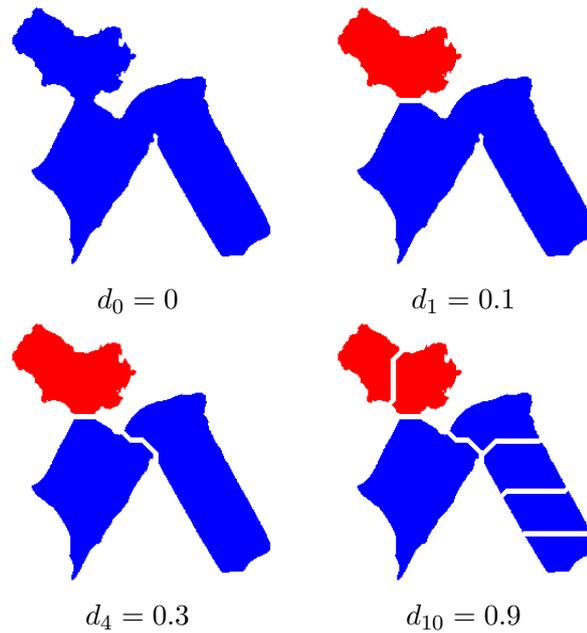


Figure 5.7: Segmentation of overlapping pieces of kerogen corresponding to layers d_0 , d_1 , d_4 and d_{10} of the label inheritance tree in figure 5.6. The colour of the objects corresponds to the label inherited from a segmentation at $d_6 = 0.5$. Red denotes vitrinite and blue denotes inertinite.

5.11 Stability experiment

Before measuring classification stability, let us recap on the previous classification experiment. Kerogen was automatically segmented from seven microscopy slides of size 2272 by 1704 pixels using CSS with parameters $s = 4$ and $d = 0.5$. The 609 kerogen objects obtained were hand labelled by human expert as discussed in section 5.3. From these objects 8 were classified as “other” and removed from the dataset. A logistic classifier was trained, tested and found to achieve accuracy of 89.07% when using all features.

The stability experiment measures the sensitivity in the accuracy of the logistic classifier for various segmentations produced using CSS with different values of overlap d . To train the classifier for each segmentation, ground truth labels are required. These labels were estimated from the expertly labelled objects, segmented at $d = 0.5$, using the label inheritance tree (see section 5.10.2).

A label inheritance tree was formed using 11 layers by sampling the overlap range at $d_0 = 0, d_1 = 0.1, d_2 = 0.2, \dots, d_{10} = 0.9, d_{11} = 1$. The *known layer* at $d_6 = 0.5$ consists of 609 centres included the objects labelled as “other”. The 10 other layers were given estimated labels according to the procedure of label inheritance. A dataset for each overlap value $d = d_i$, corresponding to each layer of the tree, was formed consisting of *inertinite*, *vitrite* and *other* objects. Each object is represented using all 32 features.

The logistic classifier was trained on the 601 kerogen objects extracted using CSS at $d = 0.5$ (excluding the objects expertly labelled as “other”). Accuracy of the classifier for each of the 11 new datasets, produced at different overlap values $d = d_i$ is obtained through 10-fold cross-validation and repeated 10 times. Objects labelled as “other” in the label inheritance tree were counted as an error in the classification. The desired result would be a high accuracy across all overlap values $d = d_i$. A classifier exhibiting this type

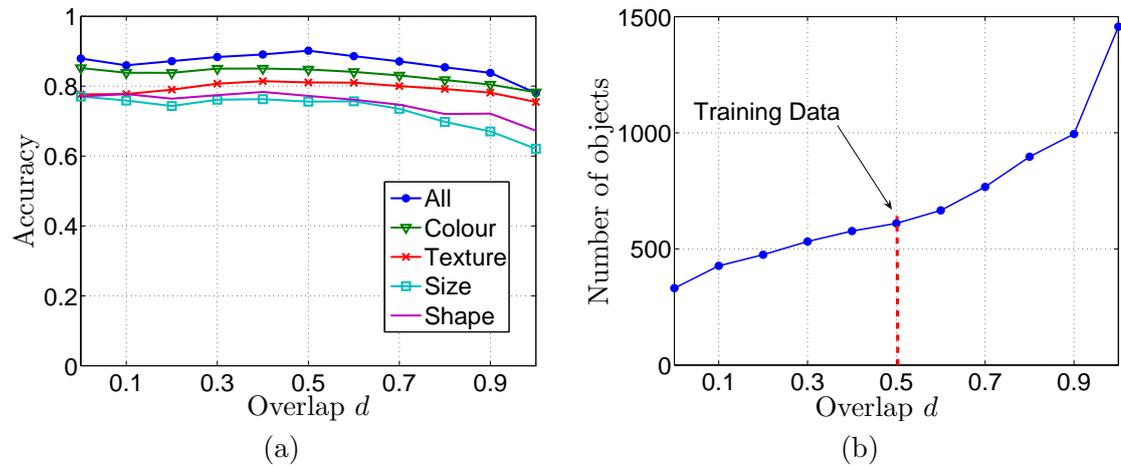


Figure 5.8: (a) Stability of the logistic classifier on various feature sets. (b) Dataset sizes at different levels of the overlap threshold d .

of behaviour would be deemed robust against the output of the CSS algorithm.

Figure 5.8(a) shows the accuracy of the logistic classifier using all features as a function of the overlap threshold. We also ran experiments for other groups of features: colour, size, texture and shape. The accuracy-overlap graphs for these groups are plotted in figure 5.8(a). The means and standard deviations across the values of d are shown in table 5.7. The standard deviation is used as a measure of stability, the lower this value the more stable the classifier is to changes in d . Here we notice that all sets of features behave fairly similarly, indicating high stability of the system across the overlap threshold.

When using all features, the observed average accuracy is 86.5%. This is slightly lower than the accuracy (89.07%) obtained when using only a single dataset of kerogen objects, extracted with overlap threshold $d = 0.5$. However, this is still acceptable for the purposes of kerogen classification.

The consistently high accuracy of the classifier is a reflection of the fact that some objects are “stable”, i.e. these objects will be present in the majority of datasets for each value of d . Stable objects will therefore acquire the same labels as in the training set (known

Table 5.7: Mean and standard deviation of the classification accuracy

Feature set	Mean	Standard deviation
All	86.50%	3.33%
Colour	83.20%	2.22%
Texture	79.20%	1.85%
Size	73.00%	4.75%
Shape	75.13%	3.37%

layer). In other words, the testing sets contain a large proportion of the training objects. The recognition rate does not suffer a significant drop even for large changes in the overlap threshold d , signifying good quality of the CSS algorithm and that of the whole system.

Figure 5.8(b) shows the number of objects in each of the 11 data sets. Viewed together with figure 5.8(a), indicates that the recognition rate deteriorates for larger number of objects but is levelled as the objects are aggregated into bigger ones (smaller d).

Table 5.7 shows that colour features strike the best balance between accuracy and stability compared to the other three groups (texture, size and shape). There are two reasons for this. Firstly, it was the colour features *mean red* and *mean blue* that ranked the highest in our feature selection experiment (see section 5.8). So we can expect that their overall performance will be better than that of the other feature groups. Secondly, colour features are robust against an object being broken into smaller pieces, whereas shape and size features would not be. This explains the decline of accuracy for the shape and size features toward the end of the graph in figure 5.8(a) and the relatively unchanged accuracy of colour and texture features. In fact, texture features exhibit the best stability across different values of d . However it is illustrated in figure 5.8(a) that regardless of d , using all the features ensures better accuracy than any of the feature groups alone. One possible avenue for improvement would be to use feature selection methods to increase overall accuracy and improve stability. For example, perhaps the combination of colour and

texture features will produce a higher and more stable accuracy than using all features. Furthermore, reducing the set of features will result in a decrease in execution time.

5.12 Summary

Ten well known classifiers were trained and tested on a dataset of kerogen objects obtained using CSS with overlap threshold $d = 0.5$. The classifiers were compared using a paired t -test and it was discovered that the logistic classifier fared best.

Using all 32 features the logistic classifier has accuracy of 89.07% for labelling kerogen objects as either inertinite or vitrinite. An improved accuracy was obtained through a greedy stepwise feature selection procedure, where six of the best features were chosen. This increased the accuracy of the classifier to 90.62%. Through analysing the weights assigned to each feature by the trained logistic classifier, a descriptive discrimination between inertinite and vitrinite was formed under transmitted light microscopy.

It is important to assess the stability of the logistic classifier to alterations made within the segmentation stage when one applies CSS. This was accomplished by further analysis of the CSS algorithm, through means of a hierarchical tree of centres. Provided a single expertly labelled segmentation is obtained, these trees can be used to estimate the labels of objects at various values of overlap threshold d . We call such trees *label inheritance trees*.

Label inheritance trees were used to label kerogen objects for 11 different values of d forming 11 different datasets. A trained logistic classifier tested on each one of these datasets was found to be stable with respect to the overlap threshold d . This is an encouraging finding and gives us ground and confidence to develop the system into a commercial product.

Nomenclature 4

C Set of centres.

C_i Set of centres at height d_i within the hierarchical tree of centres.

D Euclidean distance function/transform.

L Set of possible class labels.

$O(\mathbf{q})$ The overlap assigned to a centre \mathbf{q} .

V List of merging heights found using CSS.

d The overlap threshold used within the CSS algorithm.

d_i A sampled overlap value from the range $[0, 1)$.

f_i A mapping from the set C_i to the set L .

g_i Discriminant function.

s The minimum allowed size of an object.

Chapter 6

Complete palynomorph recognition

Having developed a system for automatic kerogen classification, attention now focuses on the recognition of palynomorphs. Our ultimate goal is to individually segment these types of organic microfossil from the image and label them into their respective classes. The first step in accomplishing this is to isolate palynomorphs that are suitable for input into a classification system (Charles, 2009).

Palynomorphs are a pale to brown organic microfossil exhibiting a sharp distinct outline with maybe some internal structure. They are noticeably different to amorphous material which does not possess a specific shape, structure or obvious outline. Various types of palynomorph can be seen in section 2.1, figure 2.1.

The varieties of palynomorph we are particularly interested in are those which present an elliptic/spherical morphology. These types of palynomorph can come from a number of different classes but particularly spores, pollen and acritarchs, a term first introduced by Evitt (1963). There are a large amount of these types of palynomorph preserved in the geological record, proving extremely useful for quantitative biostratigraphic and palaeobiological studies. Figure 6.1 shows example palynomorphs with elliptic shape.

Some of the numerous difficulties associated with detecting and segmenting palynomorphs automatically have been highlighted in figure 6.2(a). Firstly, the process used to



Figure 6.1: Examples of complete elliptic/spherical palynomorphs

arrange the palynomorphs on the microscope slide is haphazard. This causes clumping and overlapping of the organic material which increases the complexity of distinguishing between single palynomorphs, even by human eye. Secondly, their structure is deformable and palynomorphs can be found torn, squashed and folded around one another (see figure 6.2(d)). Thirdly, some palynomorphs are semi-transparent. When semi-transparent palynomorphs overlap, the outer edges of each overlapping palynomorph can be seen, even if they are underneath other palynomorphs (see figure 6.2(b-c)). This can cause problems when trying to extract individual palynomorphs by fitting ellipses to edge information. For example, a Randomized Hough Transform as proposed by McLaughlin (1998) can be applied to the edge data in order to detect ellipses. However, it was found that this type of approach was extremely slow and inaccurate for our images. Furthermore, due to semi-transparency, the colour of the overlapping palynomorphs is non-homogeneous. Alterations to the colour will occur at the areas of overlap, increasing the intricacy in locating a complete palynomorph (see figure 6.2(b)).

Current automatic classification systems work well on complete, isolated microfossils. Microfossils that are physically or visually distorted will be very difficult to classify by

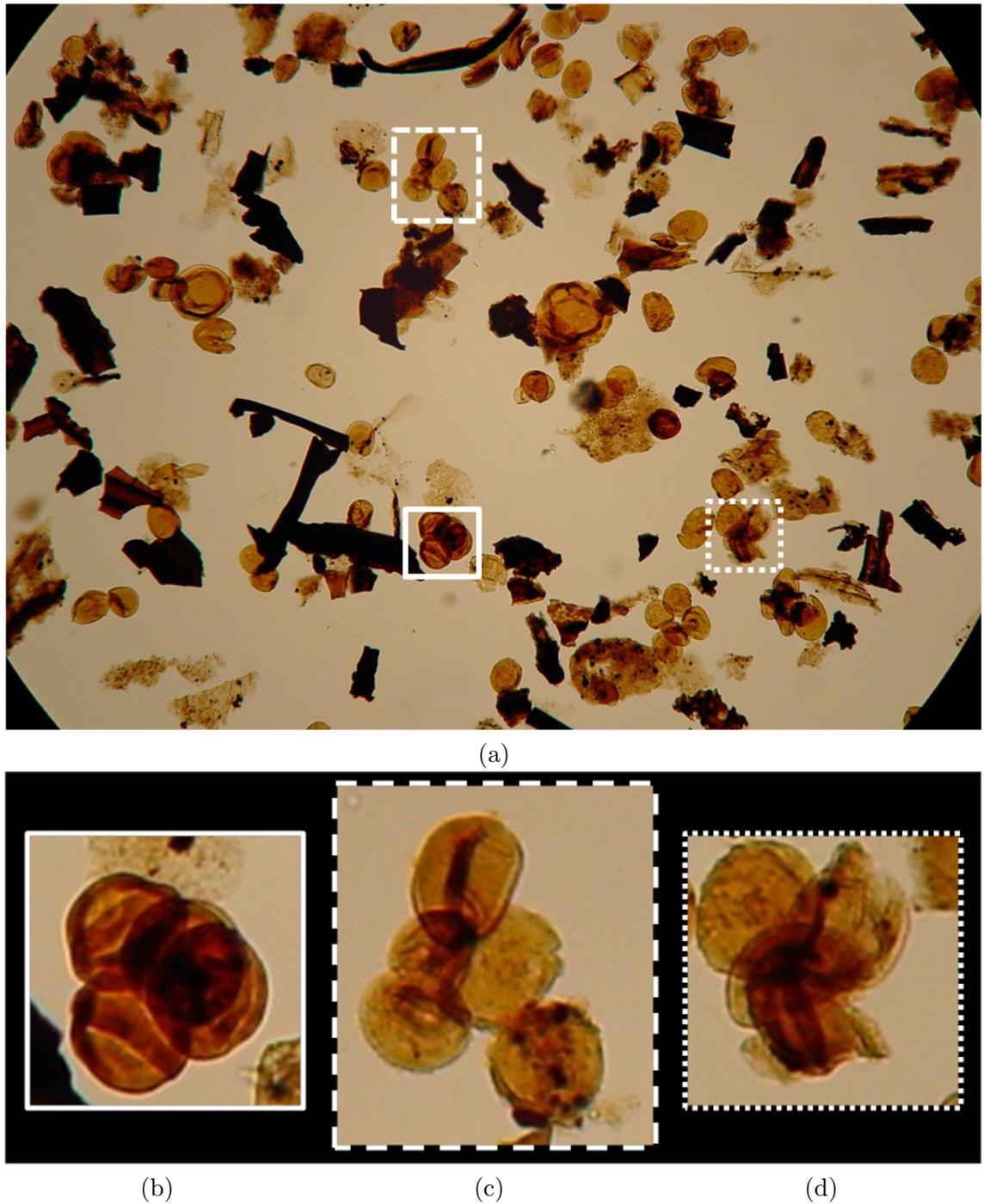


Figure 6.2: (a) Typical example of microscopy image containing palynofacies. (b) Colour changes occur due to overlapping of the palynomorphs. (c) The semi-transparent nature of palynomorphs causes edges of palynomorphs occluded by other palynomorphs to be seen. (d) Palynomorphs squashed, torn and folded

such systems. In our case palynomorphs are heavily overlapping and have folded over each other or occluded one another. An expert system for recognising microfossils under these types of conditions does not yet exist. Therefore, ideally a segmentation procedure will isolate non-folded, non-torn and non-squashed palynomorphs. Unfortunately the problems associated with palynomorph segmentation listed above, hinder an unsupervised segmentation procedure to such an extent that improper segmentation is inevitable.

Therefore, we propose a method subsequent to microfossil segmentation which will automatically detect segmented regions that contain a single, complete, elliptic/spherical palynomorph.

6.1 Microfossil segmentation

Microfossils found on a slide containing palynofacies can be broadly classified into 3 groups: Kerogen, palynomorphs and amorphous material. It has been shown that images of single particles can be recognised automatically with an accuracy of 87% (Weller et al., 2005). However for a slide containing many particles it is necessary to first locate and extract them individually.

The background of the slide is removed using the Crossing Stripe Parabolas method (Charles et al., 2008b), detailed in section 3.4. This time the threshold was set so that the black regions in the resultant binary image represent the kerogen, palynomorph and amorphous material and white regions are classed as background. An example of background segmentation for the image in figure 6.2(a) is shown in figure 6.3(a). The next step is to split up the black regions using a suitable segmentation algorithm in an attempt to isolate palynomorphs.

We have chosen to use the Centre Supported Segmentation (CSS) (Charles et al., 2008a) as our method of segmentation. This technique works well when segmenting kerogen

pieces and can be applied to any binary image separated into foreground and background. In previous chapters we have shown this method to be insensitive to small changes in its parameters, simple to implement and relatively quick to execute.

Although ellipse detection could be applied to an edge image of the slide, it was found to be slow and inaccurate. This was due to a) most of the palynomorphs on a slide are not perfect ellipses and b) noise in the edge image. On the other hand CSS is robust against noise and changes in object boundaries and can be controlled to only segment regions which overlap up to a certain limit. Heavily overlapping regions will not be segmented and this reduces the number of segmented regions containing visually distorted microfossils.

To illustrate the results of CSS used in this manner, we applied CSS to the binary image in figure 6.3(a) containing all microfossils. CSS was used with settings $s = 4$ and $d = 0.5$. Separation lines are found through marker controlled watershed segmentation, with centres acting as markers. Figure 6.3(b) shows the final segmentation of the microfossils by overlaying these lines in white on top of the binary image. In order to visualise this segmentation in context of the original microfossils colour information from the original image can be added back to the black regions of the binary image. The results of this can be seen in figure 6.4. The types of segmented objects fall into three categories: complete palynomorphs, non-palynomorphs or clumps of heavily overlapping microfossils.

6.2 Detecting complete palynomorphs

Once the foreground particles in the image have been segmented using CSS, regions containing a single complete elliptic palynomorph need to be identified. We propose using a trained classifier to distinguish between a region containing a single palynomorph and one containing kerogen, amorphous material or heavily overlapping microfossils clumped together (see figure 6.4). The novelty here is in automatically removing from the image

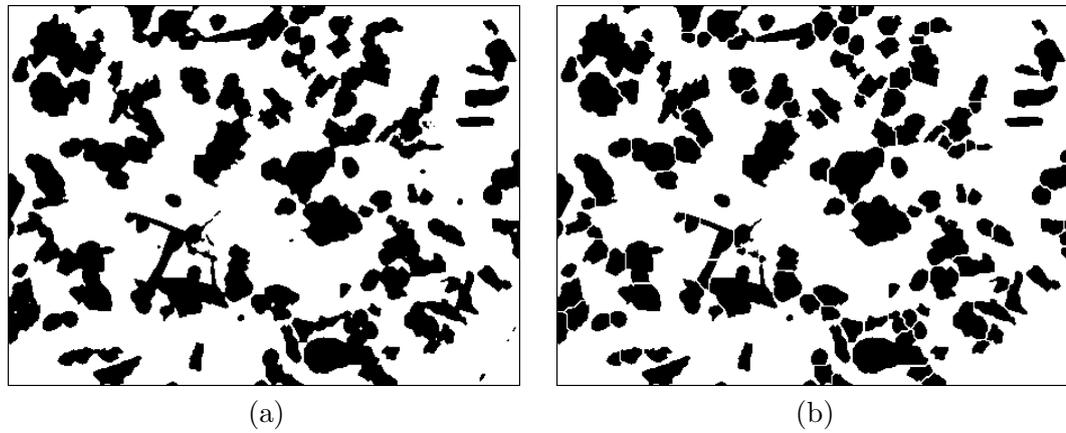


Figure 6.3: Demonstrating the pre-processing steps before complete palynomorph classification. (a) Foreground/background segmentation using Crossing Stripe Parabolas method. (b) Segmentation of microfossils using CSS.

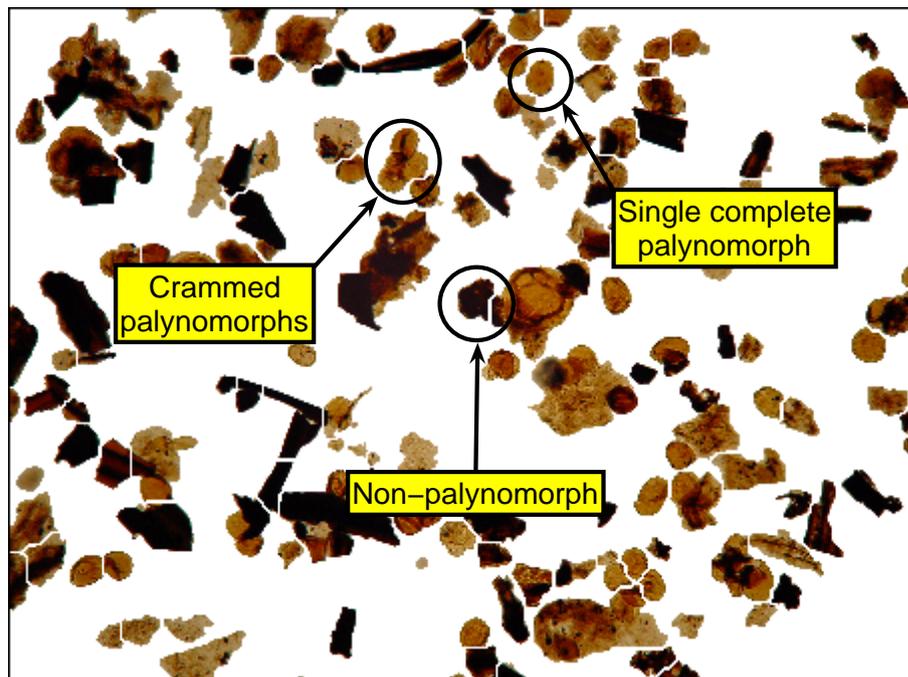


Figure 6.4: Final segmentation in context of the colour microfossils. The image has been annotated to show the types of segmented regions

complete palynomorphs which can be further classified by machine, leaving recognition of more complex regions to a human expert.

6.2.1 Classifiers

We have chosen to test the same 10 well known classifiers used in chapter 5 which have been shown to achieve a good standard across a diverse range of datasets: Logistic (Hastie et al., 2001), bagging (Breiman, 1996), support vector machines (SVM) (Cristianini and Shawe-Taylor, 2000), multilayer perceptron (Bishop, 1995), random forest (Breiman, 2001), logitboost (Friedman et al., 2000), adaboost (Freund and Schapire, 1997), decision tree (Breiman, 1984), nearest neighbour (Duda et al., 2001) and naive bayes (Hand and Yu, 2001). We used the Weka ¹ (Witten and Frank, 2005) implementation of the 10 classifiers using their default parameter settings.

6.2.2 Feature extraction and selection

Initially the 32 features used for kerogen classification found in section 5.3 are also applied here to represent an object. However, by using a subset of the 32 features one may improve classification accuracy and guard against possible overtraining. For that reason, we apply a greedy stepwise approach to feature selection within a 10-fold cross-validation.

Rather than adding features at each stage in the selection process, as in section 5.8, we begin by using the full set of 32 features and removing a feature if it reduces classification accuracy. Each feature is then ranked according to when it was removed. For example, if feature A is removed first then it will be ranked 1 and if feature B is removed last it will be ranked 32. An average of the 10 ranks for each feature, obtained from the 10 folds, is used as a measure of feature importance.

¹Weka is a free software environment for machine learning and data mining. <http://www.cs.waikato.ac.nz/ml/weka/>

6.2.3 Training data

Training data is obtained through the procedure of microfossil segmentation (detailed in section 6.1) and applied to several images containing palynofacies. Segmented objects are represented by the 32 features from table 5.3 in chapter 2. A ground truth label for each segmented object can then be constructed. An object is labelled as “palynomorph” if it consists of a single palynomorph with an elliptic shape, otherwise it is labelled as “other”. To label microfossils in this way a human expert sits in front of a monitor and manually selects the objects segmented by CSS which they deem to be complete elliptic palynomorphs.

6.2.4 Class imbalance

In some cases the dataset contains many more samples of one class than the other. The trivial (largest prior) classifier labels all samples according to the most popular class in the training set. Although reasonable accuracies can be achieved, such a classifier would be useless. Problems like this are termed *imbalanced*. Three main approaches have been employed to solve imbalanced problems (Barandela et al., 2003): a) one can assign a cost to classification errors. b) the discrimination process can be internally biased to account for the class imbalance. c) one can sample from the training set to balance the class distribution by either over-sampling from the minor class or under sampling from the major class. We chose to use an alternative approach of sampling from both classes with-replacement while maintaining a balanced class distribution.

6.2.5 Receiver-Operator characteristics (ROC) analysis

The concepts of sensitivity and specificity are used in medical tests as measures of performance but can equally be used for any binary classification. Suppose a classifier was to test an object, then there are four possible outcomes:

1. The object is a palynomorph and the classifier labels it as “palynomorph” - a true positive (TP).
2. The object is a palynomorph and the classifier labels it as “other” - a false negative (FN).
3. The object is “other” and the classifier labels it as “palynomorph” - a false positive (FP).
4. The object is “other” and the classifier labels it as “other” - a true negative (TN).

Sensitivity is the proportion of regions classified as “palynomorph” over all palynomorph objects $\frac{TP}{(TP+FN)}$. Thus a high sensitivity means very few palynomorph objects are undetected. Specificity is the proportion of objects classified as “other” over all “other” objects $\frac{TN}{(FP+TN)}$. Thus a high specificity means very few “other” objects are labelled as palynomorph.

Sometimes it is possible to estimate the posterior probability of an object being labelled as “palynomorph” from the output of a trained classifier. If this can be done then a *certainty threshold* (CT) may be applied to the estimated posterior probability to increase or lower the certainty of classifying an object as “palynomorph”. A ROC space is used to optimise the specificity and sensitivity of a classifier by adjusting the CT.

A ROC space is formed by two axes. The x axis is (1-specificity) and the y axis is the sensitivity. A perfect classifier will be at point (0, 1) in this space. Random classification lies on the diagonal line running from the point (0, 0) to the point (1, 1). Points above this line are better than a random guess. For each CT we can calculate the sensitivity and specificity of the results and plot a point in this space. In this way a curve is formed known as a ROC curve (Fawcett, 2004). The best CT is found by locating the point on the ROC curve closest to (0, 1). Additionally, classifier performance can be measured as the total

area under this curve (AUC), the closer this measure is to 1 the better the classifier.

6.2.6 Classification using top 10 classifiers

Our dataset was formed by segmenting seven images containing palynofacies as detailed in section 6.1 using CSS with parameters $s = 4$ and $d = 0.5$. This resulted in 1139 objects, consisting of 142 objects with a ground truth label of “palynomorph” and 997 with a ground truth label of “other”.

Accuracy is calculated by performing a 10-fold cross validation 10 times. Due to class imbalance, rather than training the classifier directly on each fold, a bootstrap sample of 90% is drawn so that the classes have approximately 50/50 representation. By repeating the bootstrap sampling 10 times a bias towards the training sample is reduced. Classification accuracy and AUC is therefore an average of 1000 testing sets of size 113 objects each. A 95% confidence interval (CI) is retrieved by finding the 26th and 975th largest accuracy.

The accuracies of each classifier together with their CI’s are shown in table 6.1, AUC’s are displayed in table 6.2. The CI’s are heavily overlapping indicating no clear winner between the various classifiers, but AUC values indicate the logistic classifier out-performs the rest. For clarification, ROC curves for each classifier are illustrated in figure 6.5 and figure 6.6. The curves are shown in order of AUC.

When detecting complete elliptic palynomorphs we recommend the logistic classifier due to its simplicity, high classification accuracy, high AUC and speed of training/testing.

6.3 Classification using the logistic classifier

An increase in performance was found when a greedy stepwise feature selection method was conducted (as in section 6.2.2). The top 10 ranked features were selected and

Table 6.1: Accuracy and 95% CI's of the ten classifiers using a 10 times 10-fold cross-validation.

Classifier	Accuracy (%)	95% CI (%)
Logistic	88.31	(82.45, 92.11)
Bagging	90.77	(85.09, 94.74)
SVM	85.33	(78.94, 92.11)
MLP	88.24	(81.58, 93.86)
Random Forest	92.89	(88.59, 93.86)
LogitBoost	88.04	(81.58, 93.86)
AdaBoost	85.29	(77.19, 92.98)
Decision Tree	88.24	(81.58, 92.11)
Nearest Neighbour	88.49	(83.33, 92.11)
Naive Bayes	75.74	(67.54, 84.96)

Table 6.2: AUC's of the ten classifiers using a 10 times 10-fold cross-validation.

Classifier	AUC
Logistic	0.946
Bagging	0.934
SVM	0.938
MLP	0.931
Random Forest	0.933
LogitBoost	0.937
AdaBoost	0.926
Decision Tree	0.822
Nearest Neighbour	0.845
Naive Bayes	0.928

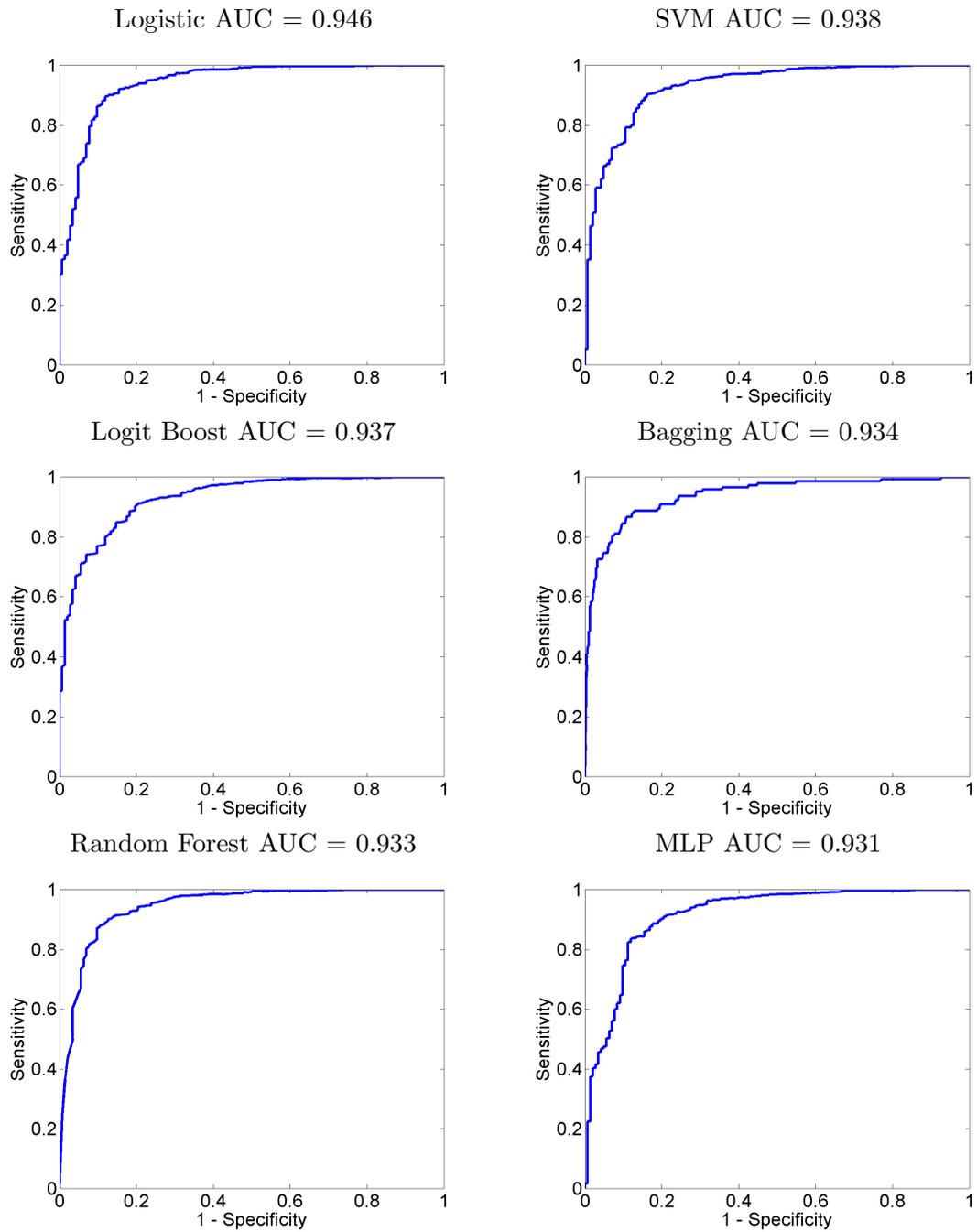


Figure 6.5: ROC curves for first 6 classifiers in order of AUC value

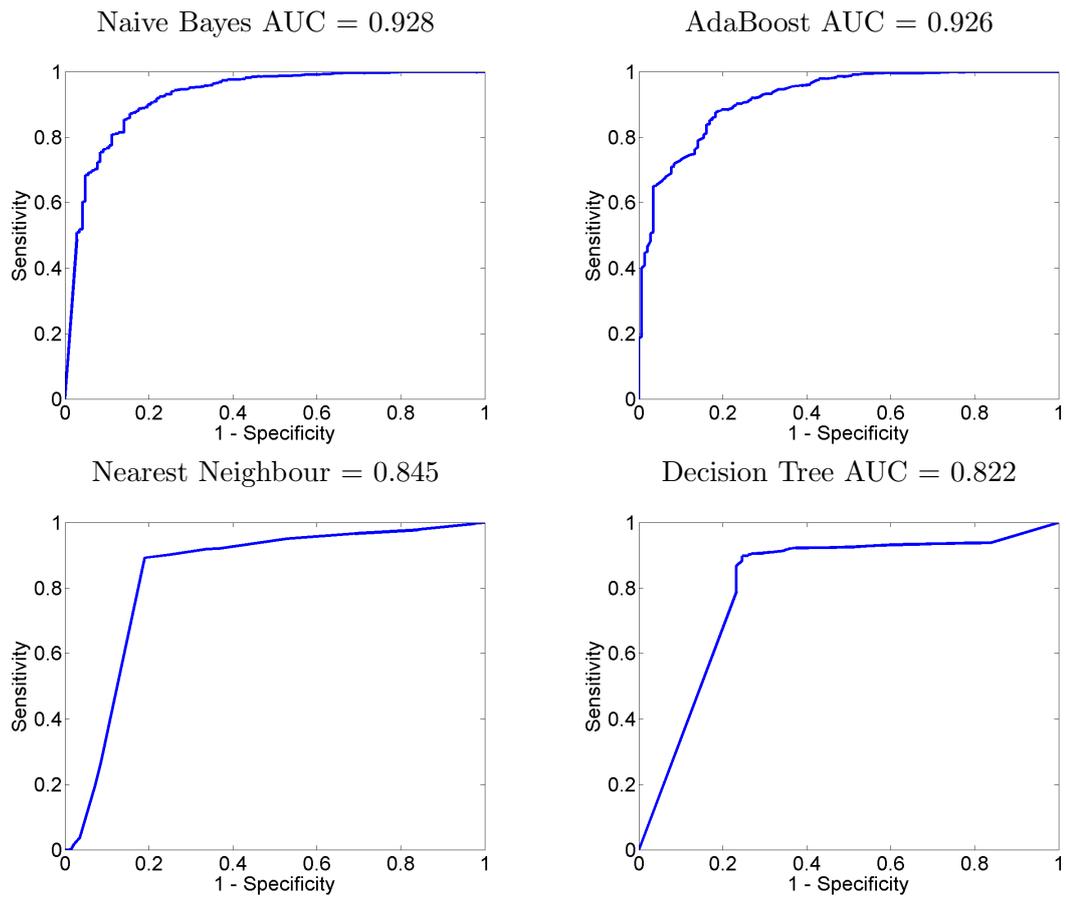


Figure 6.6: ROC curves for last 4 classifiers in order of AUC value

Table 6.3: Top 10 features ranked by the greedy stepwise selection procedure using the logistic classifier (average of 10-fold cross-validation).

Feature	Rank \pm std
distance	30.20 \pm 0.87
covariance	24.70 \pm 6.34
eccentricity	24.40 \pm 5.62
outter radius	24.20 \pm 7.59
mean red	24.00 \pm 6.28
mean blue	23.60 \pm 4.72
mean green	22.00 \pm 7.60
rectangularity	21.60 \pm 6.55
anisotropy	21.50 \pm 8.64
variance y	21.40 \pm 10.50

can be found listed in table 6.3 along with their average ranks.

The logistic classifier was assessed using 10-fold cross-validation 10 times on our dataset of 1139 objects. Class imbalance was accounted for by bootstrapping the training folds at 90% of their original size while maintaining a balanced class distribution. The bootstrapping was also repeated 10 times. The logistic classifier was trained using the top 10 ranked features and the accuracy, specificity and sensitivity were calculated as an average of 1000 results. The CT was increased from 0 to 1 in steps of 0.001, yielding 1001 points in ROC space. Joining the points forms a ROC curve shown in Figure 6.7. The best CT was found at 0.54 with a specificity of 88%, sensitivity of 87%, accuracy of 88% and an AUC of 0.944. Although the feature selection procedure has had no effect on the accuracy of classification or AUC value, it has reduced computation time and limited the possibility of overtraining.

6.4 Logistic classification examples

For demonstration purposes we will train the classifier on six of our images. The seventh image will be used to test the classifier and display the objects labelled as a palynomorph. The training set is re-sampled to attain a balanced class distribution. The

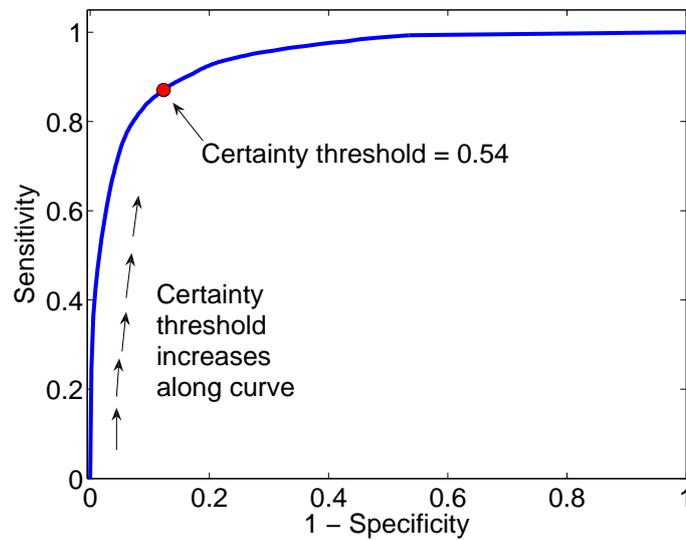


Figure 6.7: ROC curve of logistic classifier with respect to changes in the CT

logistic classifier is trained and then tested on the segmented image shown in Figure 6.4 using the 10 features listed in table 6.3. The ROC analysis is performed and the best CT was found to be 0.53. The accuracy is 91.1% with sensitivity 93.8% and specificity 90.4%. The regions correctly labelled as palynomorph by the logistic classifier are shown in figure 6.8 highlighted with a white border. False positives are highlighted with a green border and false negatives are highlighted with a blue border. For clarity, all other regions of the image have been faded.

In the same way we trained the logistic classifier three different times on various subsets of the original 7 images. Using a subset size of 6 meant that the remaining image in each case could be used to test the classifier. Segmentation results for each of the three test images along with accuracy, sensitivity and specificity can be seen in figures 6.9, 6.10 and 6.11.

This result is promising and shows that a relatively simple classifier can be used to detect complete elliptic palynomorphs with high accuracy. Most errors occurred in labelling

amorphous material as “palynomorph”. Because the amorphous material has a rough appearance, the number of false positives could be reduced by adding more specific texture features. Further improvements to the classification system could be to train statistical shape models of true complete elliptic palynomorphs (Cootes et al., 1992). These models can produce a feasible space of shapes which could in turn be used to provide another confidence level in labelling an unknown object.

Accuracy = 91.1%, Sensitivity = 93.8%, Specificity = 90.4%

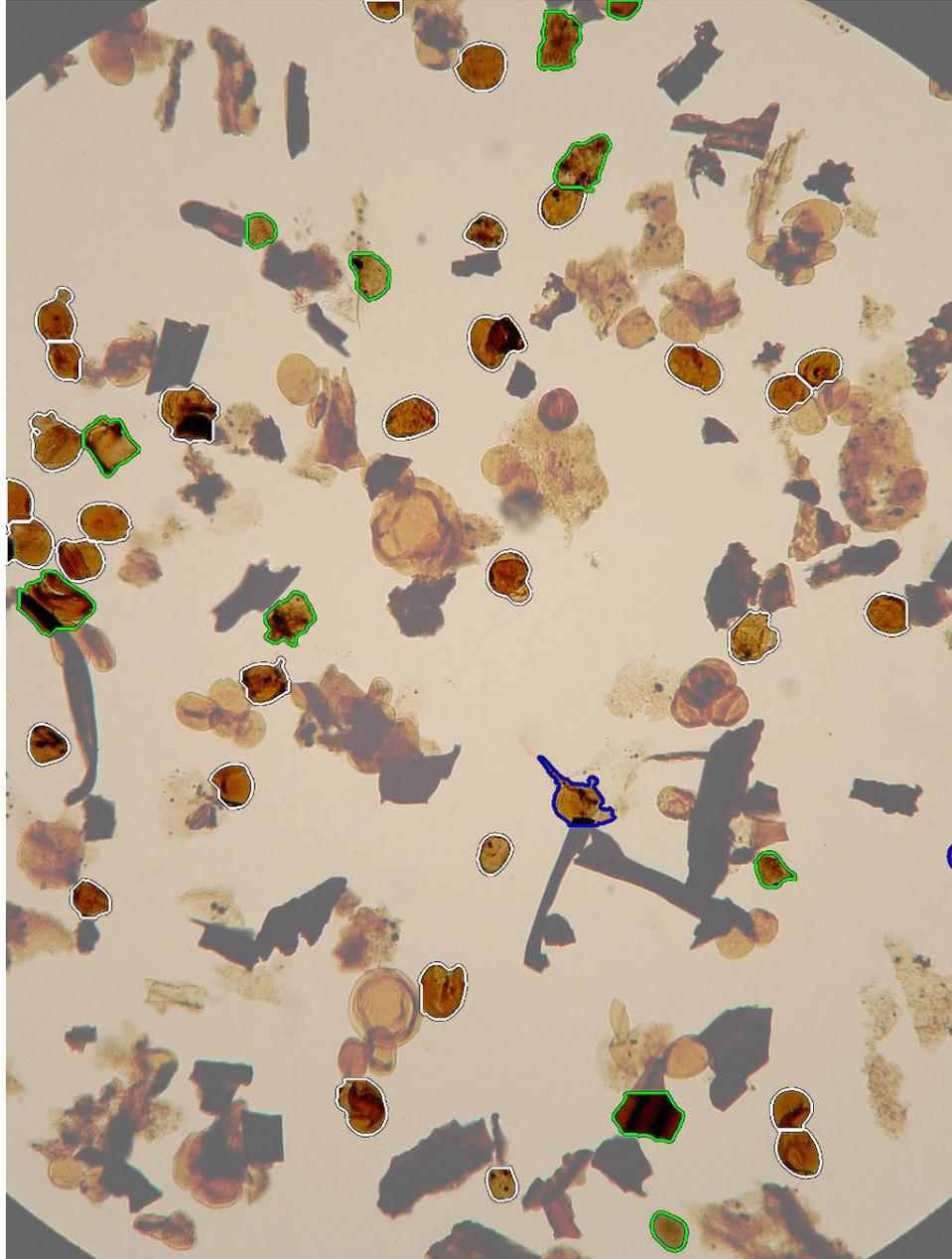


Figure 6.8: Regions highlight with a white border have been correctly classified by the logistic classifier as complete elliptic palynomorphs. Regions highlighted with a green and blue border are false positives and false negatives respectively.

Accuracy = 89.3%, Sensitivity = 90.9%, Specificity = 88.8%

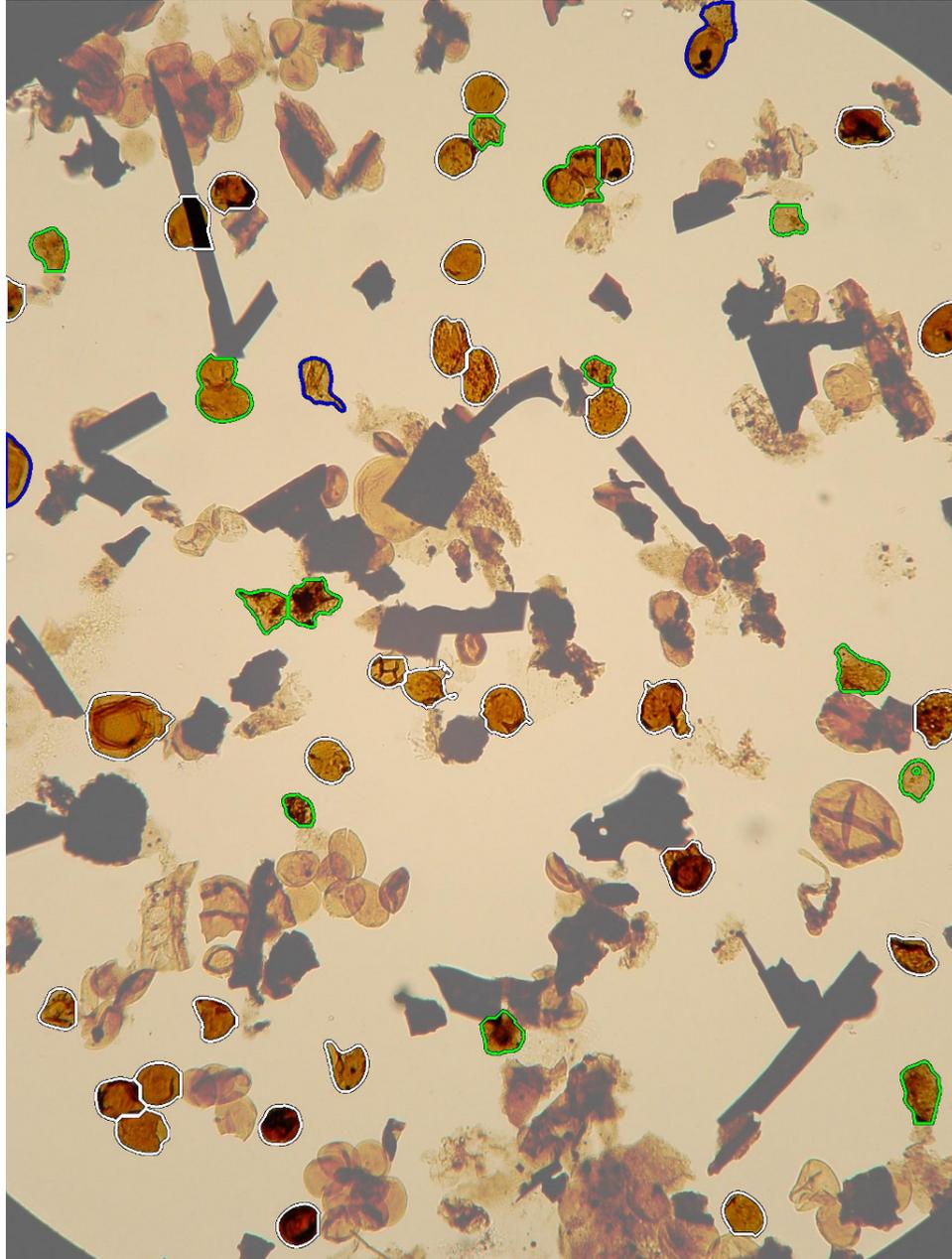


Figure 6.9: Regions highlight with a white border have been correctly classified by the logistic classifier as complete elliptic palynomorphs. Regions highlighted with a green and blue border are false positives and false negatives respectively.

Accuracy = 89.1%, Sensitivity = 95.2%, Specificity = 88.0%

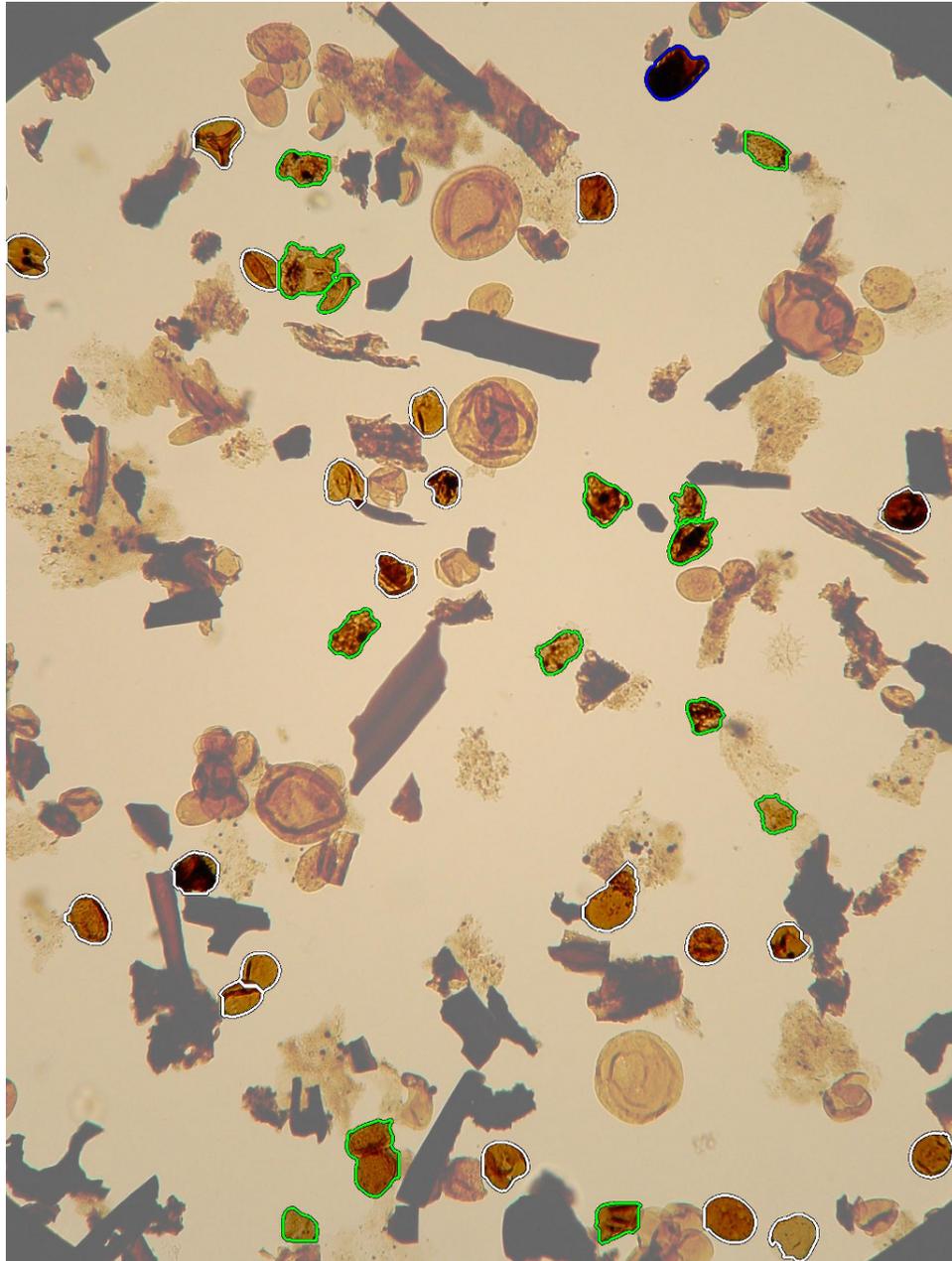


Figure 6.10: Regions highlight with a white border have been correctly classified by the logistic classifier as complete elliptic palynomorphs. Regions highlighted with a green and blue border are false positives and false negatives respectively.

Accuracy = 92.6%, Sensitivity = 42.1%, Specificity = 98.3%

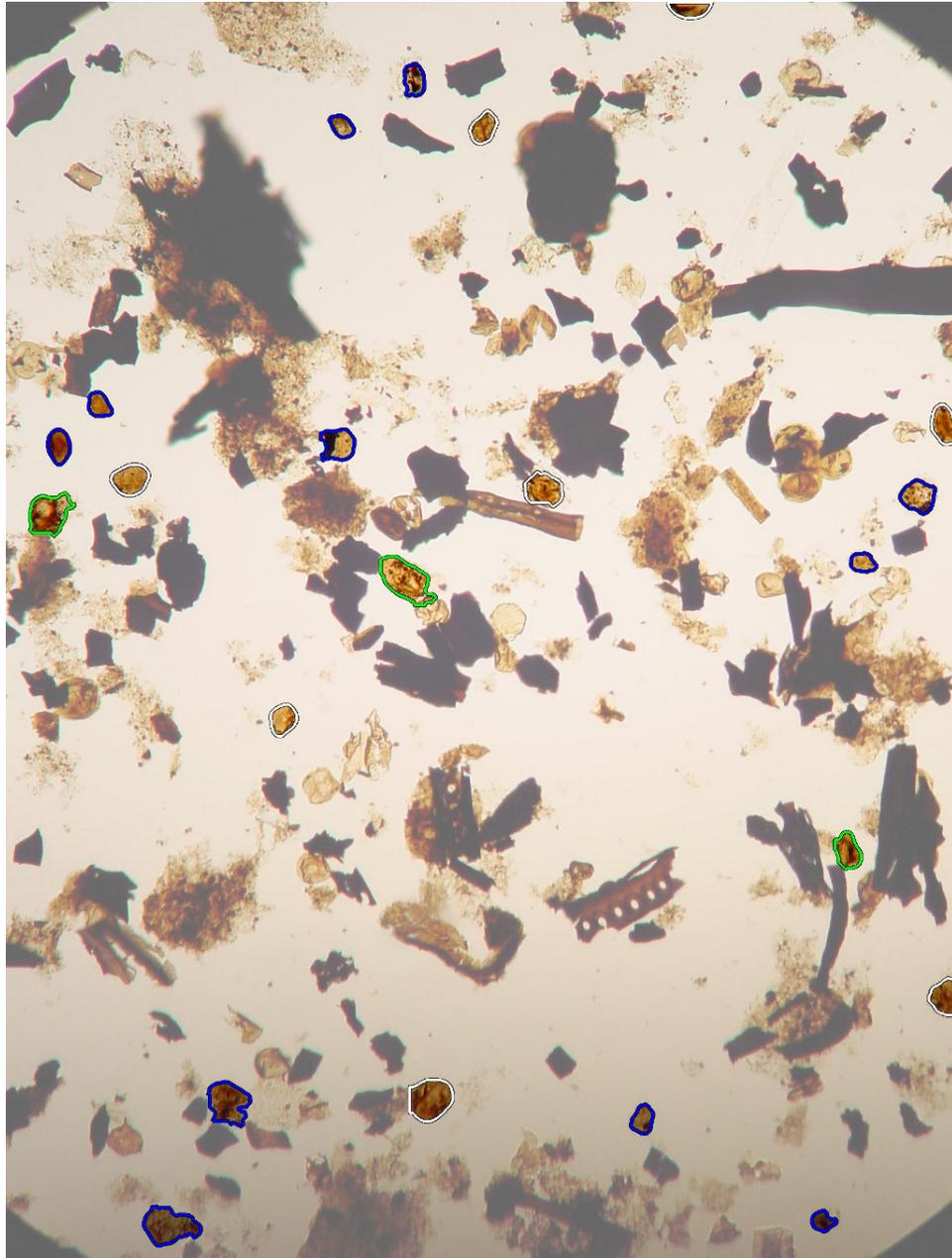


Figure 6.11: Regions highlight with a white border have been correctly classified by the logistic classifier as complete elliptic palynomorphs. Regions highlighted with a green and blue border are false positives and false negatives respectively.

6.5 Summary

In this chapter we detail a method for extracting, complete elliptically-shaped palynomorphs from images of slides containing palynofacies. The background is automatically segmented and then individual objects are located using the CSS method (see chapter 4.3).

Most of the foreground material is heavily overlapping or physically distorted. Therefore, segmented regions may contain clumps of microfossils, squashed folded or torn palynomorphs or single complete elliptic palynomorphs. We propose to use the logistic classifier to distinguish between single complete elliptic palynomorphs and “other” types of segmented objects. To select the classifier we carried out 10 times 10-fold cross-validation with 10 further splits in each fold. The logistic classifiers showed specificity of 88%, sensitivity of 87%, accuracy of 88% and an AUC of 0.944.

Nomenclature 5

d The overlap threshold used within the CSS algorithm.

s The minimum allowed size of an object.

Conclusion

The petroleum industry has keen interest in microfossils found buried deep underground. Such microfossils are utilised when forming detailed surveys regarding the depositional environment and consequently the likelihood of oil or gas prone sites.

Microfossils are retrieved in the form of rock cuttings produced when drilling deep into the earth. These cuttings are brought back up the drill-string with the aid of an oil lubricant and then washed and sieved. Organic microfossils can be seen when the processed cuttings are observed under a microscope. This type of sample will contain an assemblage of organic microfossils known as *palynofacies* which can be broadly classified into three groups: kerogen, palynomorph and amorphous. By analysing the properties of microfossil from each of these groups a detailed interpretation of the environment can be formed.

7.1 Overview

Currently, samples containing palynofacies are painstakingly analysed by human experts. The properties from a sample of microfossils found on a microscope slide are recorded in order to build a representation of its contents. Our aim was to partly automate and improve on this method of data retrieval by automatically analysing *all* microfossils of interest on a single slide.

To accomplish this goal we first concentrated on automatically identifying kerogen material and further classifying this type of organic microfossil into vitrinite and inertinite.

The system for automatic kerogen classification was built consisting of 5 steps: Image acquisition, background removal, microfossil segmentation, feature extraction and classification.

The background removal stage addressed the issue of correcting for non-uniform lighting across the image of a microscope slide using the Crossing Stripes Parabola (CSP) method. Subsequently, microfossil segmentation was conducted on the resultant binary image using the Centre Supported Segmentation (CSS) method.

Having split the kerogen into individual objects, a numerical representation of the kerogen pieces was constructed through feature extraction. Ten well known classifiers were tested against one another and the logistic classifier proved best, with accuracy 89.07%. An improved accuracy of 90.62% was found when a reduced feature set was used.

Classification stability was analysed between the classification stage and kerogen segmentation stage. The logistic classifier was found to be stable with respect to the CSS algorithm, giving us ground and confidence to continue developing the system into a commercial product.

Current microfossil recognition systems cannot easily recognise distorted, folded or torn palynomorphs. Hence, our proposed method was designed to automatically segment palynomorphs suitable for further classification. Again this procedure utilised CSP, CSS and a logistic classifier for recognising complete elliptic palynomorphs with a specificity of 88%, sensitivity of 87%, accuracy of 88%.

The hypothesis states that a robust system for accurate and fast fossil identification is possible. Results shown here support this hypothesis and demonstrate that automatic detection of individual microfossils is achievable with high accuracy while also being robust against varying conditions. In addition, the high accuracy of classification and insensitivity to parameter choices has led the system to be deployed as a commercial product and used within industry. Furthermore, our intentions were to develop new image analysis techniques

that could be used within other domains. We successfully developed new algorithms known as CSS and CSP. The next section on future work discusses the plausibility of implementing such algorithms into other systems.

7.2 Future work

An image with uneven illumination caused by vignetting or strong central light source can be corrected by CSP provided background intensities vary smoothly across the image. These types of distortions occur within microscopy and astrophotography images. Although the CSP algorithm assumes a lighter background compared to the foreground, input images with darker background can be easily accounted for with only minor modification to the algorithm. The algorithm can therefore be enhanced by adding a method to detect relative background intensity with respect to the foreground. One such method is proposed by Lindblad and Bengtsson (2001). By utilising this type of detection the CSP algorithm could be applied automatically to most microscopy images, albeit a dark foreground on a light background or vice versa. Astrophotography images of the night sky show a very similar distortion to the background illumination found within microscopy images. We hypothesise that a relatively dark uneven background against the light of the stars could be corrected by applying this modified CSP algorithm.

Extending CSP to allow for 3D images, e.g., magnetic resonance images (MRI), opens the doorway to a whole range of other domains where CSP could be applied. As an example, the method of background fitting in CSP could be used to remove background noise in MRI. A study on possible applications for CSP could lead to the development of an even more general algorithm in correcting uneven background data from other sources.

The separation of microfossils was conducted solely using CSS. Would an improvement in segmentation accuracy be achievable by combining other segmentation algorithms

with CSS? For example, a corner detection algorithm could provide a confidence level when splitting two overlapping objects found using CSS. Improvement of classification accuracy has been achieved through classifier ensembles. It can be expected that an ensemble of segmentation algorithms may produce better segmentation than any individual algorithm.

Complete elliptic palynomorphs are extracted with an application of CSS followed by classification using a logistic classifier. Advancement to this technique could be to provide the segmentation algorithm with feedback from the classifier. Using classifier knowledge, the segmentation algorithm could be automatically adjusted to result in an improved segmentation, i.e., more complete elliptic palynomorphs are found. This iterative approach to segmentation could then be repeated until certain conditions are met.

Not only could the CSS algorithm be applied in other areas of microscopy, such as cell counting, but it also lends itself to any image containing convex overlapping/touching objects. Extending the algorithm to incorporate 3D or even 4D data permits CSS to be functional within 3D microscopy or MRI. For example, a 3D version of CSS could be applied to segment touching pollen spores taken through a 3D microscopy or used within MRI to segment tumours from healthy tissue.

In the case of microfossil classification, labels from experts can vary from one expert to another. Therefore, before deploying a commercial product, further experts could be consulted to obtain their opinions for the kerogen labels. The concordance between them can then be evaluated to produce a more robust (consensus-based) system.

It is also necessary for further research to be conducted in order to successfully extract highly overlapping and occluded palynomorphs and amorphous material. Later, classification systems can be built for recognising these types of visually distorted microfossils.

7.3 Publications related to the thesis

Charles, J., Kuncheva, L., Wells, B., Lim, I., September 2006. An evaluation measure of image segmentation based on object centres. LNCS Image analysis and recognition 4141, 283-294.

Charles, J., Kuncheva, L., Wells, B., Lim, I., June 2008a. Object segmentation within microscope images of palynofacies. Computers & Geosciences 34 (6), 688-698.

Charles, J., Kuncheva, L., Wells, B., Lim, I., 2008b. Background segmentation in microscope images. In Proc 3rd International Conference on Computer Vision Theory and Applications VISAPP08, Madeira, Portugal.

Kuncheva, L., Charles, J., Miles, N., Collins, A., Wells, B., Lim, I., 2008. Automated kerogen classification in microscope images of dispersed kerogen preparation. Mathematical Geology 40 (6), 639-652.

Charles, J., Kuncheva, L., Wells, B., Lim, I., 2009. Stability of kerogen classification with regard to image segmentation. Mathematical Geology 41 (4), 475, DOI:10.1007/s11004-009-9219-3.

Charles, J., 2009. Automatic recognition of complete palynomorphs in digital images. Machine Vision and Applications, DOI: 10.1007/s00138-009-0200-4.

Glossary

amorphous

Organic microfossils that do not possess a specific shape, structure or obvious outline.

clustering

Unsupervised learning method of grouping objects of a similar kind.

feature

A measurable property of an object.

inertinite

Maceral derived from the tissues of higher plants and gelified amorphous material..

kerogen

An organic chemical compound found in sedimentary rock that releases oil or gas upon heating.

maceral

A component of coal; analogous to the term mineral.

microfossil

A fossil generally no larger than 4mm and commonly smaller than 1mm.

palynofacies

The acid-resistant organic microfossils found in sediments and sedimentary rocks.

palynomorph

The preserved cell, tissue or organ component of sedimentary organic microfossils derived from an organism.

segmentation

The process of splitting up an image into meaningful regions.

thresholding

A simple form of image segmentation where pixels are labelled as object pixels if their value is lower than a predefined threshold.

vitrite

Maceral derived from woody tissues of roots, stems, barks and leaves composed of cellulose and lignin.

Bibliography

- Al-Ameri, T., Batten, D., 1997. Palynomorph and palynofacies indications of age, depositional environments and source potential for hydrocarbons: Lower cretaceous zubair formation, southern irag. *Cretaceous Research* 18, 789–797.
- Athersuch, J., Banner, F., Higgins, A., Howarth, R., Swaby, P., 1994. The application of expert systems to the identification and use of microfossils in the petroleum industry 26 (4), 483–489.
- Barandela, R., Sanchez, J., Garcia, V., Rangel, E., 2003. Strategies for learning in class imbalance problems. *Pattern Recognition* 36, 849–851.
- Bartels, R., Beatty, J., Barsky, B., 1987. An introduction to Splines for use in Computer Graphics and Geometric Modeling. Morgan Kaufmann Publishers, Inc.
- Beauchemin, M., Thomson, K. P. B., 1997. The evaluation of segmentation results and the overlapping area matrix. *Remote Sensing* 18 (18), 3895–3899.
- Ben-Hur, A., Elisseeff, A., Guyon, I., 2002. A stability based method for discovering structure in clustered data. In: *Proc. Pacific Symposium on Biocomputing*. pp. 6–17.
- Beucher, S., 1992. The watershed transformation applied to image segmentation. *Scanning Microsc* 6, 299–314.
- Bishop, C., 1995. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford.
- Bishop, C., 2006. *Pattern Recognition and Machine Learning*. Springer, New York.
- Blayvas, I., Bruckstein, A., Kimmel, R., January 2006. Efficient computation of adaptive threshold surfaces for image binarization. *Pattern Recognition* 39 (1), 89–101.
- Bollmann, J., Quinn, P. S., Vela, M., Brabec, B., Brechner, S., Cortes, M. Y., Hilbrecht, H., Schmidt, D. N., Schiebel, R., Thierstein, H. R., 2004. Automated particle analysis: Calcareous microfossils. *Image Analysis, Sediments and Paleoenvironments* 7, 229–252.
- Bonton, P., Boucher, A., Thonnat, M., Tomczak, R., Hidalgo, P., Belmonte, J., Galan, C., 2002. Colour image in 2d and 3d microscopy for automation of pollen rate measurement. *Image analysis and stereology*.
- Borsotti, M., Campadelli, P., Schettini, R., 1998. Quantitative evaluation of color image segmentation results. *Pattern Recognition Letters* 19, 741–747.

- Boucher, A., Hidalgo, P., Thonnat, M., Belmonte, J., Galan, C., Bonton, P., Tomczak, R., 2002. Development of a semi-automatic system for pollen recognition. *International Journal of Aerobiology* 18 (3), 195–201.
- Breiman, L., 1984. *Classification and Regression Trees*. Wadsworth International, Belmont.
- Breiman, L., 1996. Bagging predictors. *Machine Learning* 26 (2), 123–140.
- Breiman, L., 2001. Random forests. *Machine Learning* 45, 5–32.
- Breu, H., Gil, J., Kirkpatrick, D., Werman, M., 1995. Linear time euclidean distance transform algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 17 (5), 529–533.
- Brinkmann, B. H., Maduca, A., Robb, R. A., April 1998. Optimized homomorphic unsharp masking for mr grayscale inhomogeneity correction. *IEEE Transactions on Medical Imaging* 17, 161–171.
- Bruckner, S., 2000. Estimation of the background in powder diffraction patterns through a robust smoothing procedure. *Applied Crystallography* 33, 977–979.
- Buckley, L., 2004. Organic facies of the cretaceous colorado group, western canada sedimentary basin. Univeristy of Newcaslte-upon-Tyne.
- Burmham, A., Sweeny, J., 1989. A chemical kinetic model of vitrinite reflectance maturation. *Geochimica et Cosmochimica Acta* 53, 2649–2657.
- Canny, J., 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (6), 679–714.
- Cardoso, J. S., Corte-Real, L., November 2005. Toward a generic evaluation of image segmentation. *IEEE Transactions on Image Processing* 14 (10), 1773–1782.
- Casasent, D., Talukder, A., Cox, W., Chang, H., Weber, D., 1996. Detection and segmentation of multiple touching product inspection items. In: *Proceedings of SPIE*. Vol. 2907. p. 205.
- Chan, F., Lam, F., Zhu, H., March 1998. Adaptive thresholding by variational method. *IEEE Transactions on Image Processing* 7 (3), 468–473.
- Charles, J., 2009. Automatic recognition of complete palynomorphs in digital images. *Machine Vision and Applications* DOI:10.1007/s00138-009-0200-4.
- Charles, J., Kuncheva, L., Wells, B., Lim, I., September 2006. An evaluation measure of image segmentation based on object centres. *LNCS Image analysis and recognition* 4141, 283–294.
- Charles, J., Kuncheva, L., Wells, B., Lim, I., June 2008a. Object segmentation within microscope images of palynofacies. *Computers & Geosciences* 34 (6), 688–698.

- Charles, J., Kuncheva, L., Wells, B., Lim, I., 2008b. Background segmentation in microscope images. In Proc 3rd International Conference on Computer Vision Theory and Applications VISAPP08, Madeira, Portugal.
- Charles, J., Kuncheva, L., Wells, B., Lim, I., 2009. Stability of kerogen classification with regard to image segmentation. *Mathematical Geology* 41 (4), 475, DOI:10.1007/s11004-009-9219-3.
- Cinque, L., Foresti, G., Lombardi, L., September 2004. A clustering fuzzy approach for image segmentation. *Pattern Recognition* 37 (9), 1797–1807.
- Clocksink, W., 2003. Automatic segmentation of overlapping nuclei with high background variation using robust estimation and flexible contour models. 12th International Conference on Image Analysis and Processing 0, 682.
- Combaz, A., 1964. Les palynofaciés. *Revue de Micropaléontologie* 7 (3), 205–218.
- Cootes, T., Taylor, C., Cooper, D., Graham, J., 1992. Training models of shape from sets of examples. In: Proc. British Machine Vision Conference. Vol. 557. pp. 9–18.
- Cristianini, N., Shawe-Taylor, J., 2000. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK.
- Demšar, J., 2006. Statistical comparison of classifiers over multiple data sets. *Journal of Machine Learning Research* 7, 1–30.
- Du Buf, H., Bayer, M., 2002. *Automatic Diatom Identification*. World Scientific Publishing Company.
- Duda, R., Hart, P., Stork, D., 2001. *Pattern Classification*. John Wiley & Sons, Ny, second edition.
- Eikvil, L., Taxt, T., Moen, K., 1991. A fast adaptive method for binarization of document images. Proceedings of the First International Conference on Document Analysis and Recognition, Saint-Malo, France, 435–443.
- Epstein, A., Epstein, J., Harris, L., 1977. Conodont colour alteration - an index to organic metamorphism. *U.S. Geol. Surv* 995, 1–27.
- Evitt, W., 1963. A discussion and proposals concerning fossil dinoflagellates, hystrichospheres and acritarchs. *Proceedings of National Academy of Sciences* 49, 158–164.
- F., W. M., W., M. C., L., B., October 1999. A comparison of radial basis function and back-propagation neural networks for identification of marine phytoplankton from multivariate flow cytometry data. *Applied and Environmental Microbiology* 65 (10), 4404–4410.
- Fawcett, T., 2004. Roc graphs: Notes and practical considerations for researchers. *Machine Learning* 31.

- Flesche, H., Nielsen, A., Larsen, R., 2000. Supervised mineral classification with semiautomatic training and validation set generation in scanning electron microscope energy dispersive spectroscopy images of thin sections. *Mathematical Geology* 32 (3).
- France, I., Duller, A., Duller, G., Lamb, H., February 2000. A new approach to automated pollen analysis. *Quaternary Science Reviews* 19 (6), 537–546.
- Freund, Y., Schapire, R., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Science* 55 (1), 119–139.
- Friedman, J., Hastie, T., Tibsharani, R., 2000. Additive logistic regression: a statistical view of boosting. *Annals of Statistics* 28 (2), 337–374.
- Gevers, T., Stokman, H., 2004. Robust histogram construction from color invariants for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26 (1), 113–118.
- Gonzalez, R., Wintz, P., 2002. *Digital image processing*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
- Gonzalez, R. C., Woods, R. E., Eddins, S. L., 2004. *Digital Image Processing Using Matlab*. Pearson Education, Inc.
- Hand, D., Yu, K., 2001. Idiot's Bayes - not so stupid after all? *International Statistical Review* 69, 385–398.
- Haralick, R., Shanmugam, K., Dinstein, I., November 1973. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics* 3 (6), 610–621.
- Hastie, T., Tibsharani, R., Friedman, J., 2001. *Elements of Statistical Learning*. Springer, New York.
- Hoelstad, T., Pisecki, S., Stemmerik, L., 1994. Shape and size of lacustrine deposited melanogen (opaque organic matter), upper carboniferous, east greenland. *Gronlands-Geologiske Undersogelse* 164, 19–28.
- Honkanen, M., Saarenrinne, P., Stoor, T., Niinimaki, J., 2005. Recognition of highly overlapping ellipse-like bubble images. *Measurement Science and Technology* 16 (9), 1760.
- Hoover, A., Jean-Baptiste, G., Jiang, X., Flynn, P., Bunke, H., Goldgof, D., Bower, K., Egert, D., Filtzbiggbon, A., Fisher, R., 1996. An eperimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18, 673–689.
- Hubert, L., Arabie, P., 1985. Comparing partitions. *Journal of Classification* 2, 193–218.
- International Committee for Coal and Organic Petrology (ICCP), 1998. The new vitrinite classification (iccp system 1994). *Fuel* 77 (5), 349–358.

- Iwamoto, S., Checkley, D. M., Trivedi, M. M., August 2001. Reflics: Real-time flow imaging and classification system. *Machine Vision and Applications* 13 (1), 1–3.
- Jalba, A., Wilkinson, M., Roerdink, J., Bayer, M., Juggins, S., September 2005. Automatic diatom identification using contour analysis by morphological curvature scale spaces. *Machine Vision and Applications* 16 (4), 217–228.
- Jonker, R., Groben, R., Tarran, G., Medlin, L., Wlkins, M., Garcia, L., Zabala, L., Boddy, L., 2000. Automated identification and characterisation of microbial populations using flow cytometry: the aims project. *Scientia Marina* 64, 225–234.
- Kittler, J., Illingworth, J., 1986. Minimum error thresholding. *Pattern Recognition* 19 (1), 41–47.
- Kohonen, T., 1989. *Self-organization and Associative Memory*. Springer-Verlag, London, UK.
- Kuncheva, L., Charles, J., Miles, N., Collins, A., Wells, B., Lim, I., 2008. Automated kerogen classification in microscope images of dispersed kerogen preparation. *Mathematical Geology* 40 (6), 639–652.
- Landini, G., Othman, I., 2003. Estimation of tissue layer level by sequential morphological reconstruction. *Journal of Microscopy* 209 (2), 118–125.
- Leong, F., Brady, M., McGee, J., 2003. Correction of uneven illumination (vignetting) in digital microscopy images. *Journal of Clinical Pathology* 56 (8), 619–621.
- Lindblad, J., Bengtsson, E., 2001. Comparison of methods for estimation of intensity nonuniformities in 2d and 3d microscope images of fluorescence stained cells. *Proceedings of the 12th Scandinavian Conference on Image Analysis (SCIA)*, 264–271.
- Lindblad, J., Waehlby, C., Bengtsson, E., Zaltsman, A., 2003. Image analysis for automatic segmentation of cytoplasm and classification of rac1 activation. *Cytometry Part A* 57, 22–33.
- Liu, J., Yang, Y. H., 1994. Multiresolution color image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16 (7), 689–700.
- Malpica, N., de Solorzano, C., Vaquero, J., Santos, A., Vallcorba, I., Garcia-Sagredo, J., del Pozo, F., 1997. Applying watershed algorithms to the segmentation of clustered nuclei. *Cytometry* 28 (4), 289–297.
- McLaughlin, R., 1998. Randomized hough transform: Improved ellipse detection with comparison. *Pattern Recognition Letters* 19 (3-4), 299–305.
- Megiddo, N., 1983. Linear-time algorithms for linear programming in \mathbb{R}^3 and related problems. *Society for Industrial and Applied Mathematics* 12 (4).
- Meyer, F., 1978. Contrast feature extraction. *Quantitative Analysis of Microstructures in Material Science, Biology and Medicine Special issue of Practical Metallography*.

- Meyer, F., 1994. Topographic distance and watershed lines.
- Montage, 2007. Baseline background correction <http://www.steppingstage.com> [Accessed 21 Jan 2007].
- MVTec Software GmbH, 2008. Halcon Available from: www.mvtex.com/halcon [Accessed 30 Jun 2008].
- Nadeau, C., Bengio, Y., 2003. Inference for the generalization error. *Machine Learning* 52, 239–281.
- Nakagawa, Y., Rosenfeld, A., 1979. Some experiments on variable thresholding. *Pattern Recognition* 11 (13), 191–204.
- Osher, S., Sethian, J., Fedkiw, R., 2002. *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag.
- Otsu, N., March 1979. A threshold selection method from gray level histogram. *IEEE Trans. Systems, Man and Cybernetics* 9, 62–66.
- Palmer, D., 2005. *Earth Time: Exploring the Deep Past from Victorian England to the Grand Canyon*. John Wiley & Sons.
- Pan, X., Brady, M., Bowman, A., Crowther, C., Tomlin, R., 2004. Enhancement and feature extraction for images of incised and ink texts. *Image and Vision Computing* 22 (6), 443–451.
- Petrou, M., Bosdogianni, P., 1999. *Image Processing : The Fundamentals*. John Wiley & Sons, Inc. New York, NY, USA.
- Ping-Sung Liao, T., Chung, P., 2001. A fast algorithm for multilevel thresholding. *Journal of information science and engineering* 17, 713–727.
- Pla, F., 1996. Recognition of partial circular contours from segmented contours. *Computer Vision and Image Understanding* 63 (2), 334–343.
- Porter, R., Canagarajah, N., April 1996. A robust automatic clustering scheme for image segmentation using wavelets. *IEEE Transactions on Image Processing* 5 (4), 662–665.
- Pross, J., Pletsch, T., Shillington, D., Ligouis, B., Schellenberg, F., Kus, J., 2007. Thermal alteration of terrestrial palynomorphs in mid-cretaceous organic-rich mudstones intruded by an igneous sill. *International Journal of Coal Geology* 70, 277–291.
- Ramesh, N., Yoo, J., Sethi, I., 1995. Thresholding based on histogram approximation. *IEE Proc. Vision Image Signal Processing* 142 (5), 271–279.
- Rand, W., 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association* 66, 846–850.

- Riding, J. B., Kyffin-Hughes, J., 2004. A review of the laboratory preparation of palynomorphs with a description of an effective non-acid technique. *Revista Brasileira de Paleontologia* 7 (1), 13–44.
- Rocha, L., Velho, L., Carvalho, P., 2002. Image moments-based structuring and tracking of objects. *Computer Graphics and Image Processing*, 2002. Proceedings. XV Brazilian Symposium on, 99–105.
- Russ, J., 2006. *The Image Processing Handbook Fifth Edition*. CRC Press.
- Sahoo, P., Soltani, S., Wong, A., 1988. A survey of thresholding techniques. *Computer Vision Graphics and Image Processing* 41, 233–260.
- Sankur, B., Sezigin, M., 2004. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging* 13 (1), 146–165.
- Shafer, S., 1985. Using color to separate reflection components. *Color research and application* 10 (4), 210–218.
- Shahin, M., Symons, S., 2005. Seed sizing from images of non-singulated grain samples.
- Shatadal, P., Jayas, D., Bulley, N., 1995. Digital image analysis for software separation and classification of touching grains: I. disconnect algorithm. *Transactions of the ASAE* 38 (2), 645–649.
- Soille, P., 2003. *Morphological Image Analysis: Principles and Applications*. Springer-Verlag New York, Inc. Secaucus, NJ, USA.
- Steenstrup, S., 1981. A simple procedure for fitting a background to a certain class of measured spectra. *Applied Crystallography* 14, 226–229.
- Stopes, M., 1935. *Fuel* 14 (4).
- Sund, T., Eilertsen, K., 2003. An algorithm for fast adaptive image binarization with applications in radiotherapy imaging. *IEEE Transactions on Medical Imaging* 22 (1), 22–28.
- Thompson, C., Dembicki, H., 1986. Optical characteristics of amorphous kerogens and the hydrocarbon-generating potential of source rocks. *International Journal of Coal Geology* 6, 229–249.
- Tsai, D.-M., Hou, H., Su, H., 1999. Boundary-based corner detection using eigenvalues of covariance matrices. *Pattern Recognition Letters* 20, 31–40.
- Tyson, R., 1998. Automated transmitted light kerogen typing by image analysis. *Mededelingen Rijks Geologische Dienst* 45, 139–149.
- Tyson, R., Follows, B., June 2000. Palynofacies prediction of distance from sediment source: A case study from the upper cretaceous of the pyrenees. *Geological Society of America* 28 (6), 569–571.

- Universal Imaging Corporation, July 2004. Correcting for uneven background in an image using morphology filters.
- van den Berg, E., Meesters, A., Kenter, J., Schlager, W., 2002. Automated separation of touching grains in digital images of thin sections. *Computers & Geosciences* 28 (2), 179–190.
- Vekemans, B., Janssens, K., Vincze, L., Adams, F., Espen, P. V., 1995. Comparison of several background compensation methods useful for evaluation of energy-dispersive x-ray fluorescence spectra. *Spectrochimica Acta Part B: Atomic Spectroscopy* 50 (2), 149–169.
- Venables, W., Ripley, B., Venables, W., 1999. *Modern applied statistics with s-plus*.
- Vincent, L., 1993. Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. *Image Processing, IEEE Transactions on* 2 (2), 176–201.
- Visen, N., Shashidhar, N., Paliwal, J., Jayas, D., 2001. Identification and segmentation of occluding groups of grain kernels in a grain sample image. *Journal of Agricultural Engineering Research* 79 (2), 159–166.
- Wahlby, C., Sintorn, I., Erlandsson, F., Borgefors, G., Bengtsson, E., July 2004. Combining intensity, edge and shape informations for 2d and 3d segmentation of cell nuclei in tissue sections. *Journal of Microscopy* 215, 67–76.
- Wang, L., 1995. Automatic identification of rocks in thin sections using texture analysis. *Mathematical Geology* 27 (7), 847–865.
- Weller, A., Corcoran, J., Harris, A., Ware, J., December 2005. The semi-automated classification of sedimentary organic matter in palynological preparations. *Computers & Geosciences* 31 (10), 1213–1223.
- Weller, A., Harris, A., Ware, J., October 2006. Artificial neural networks as potential classification tools for dinoflagellate cyst images: A case using the self-organizing map clustering algorithm. *Review of Palaeobotany and Palynology* 141 (3-4), 287–302.
- Weller, A., Harris, A., Ware, J., 2007. Two Supervised Neural Networks for Classification of Sedimentary Organic Matter Images from Palynological Preparations. *Mathematical Geology* 39 (7), 657.
- Wells, B., 2008. Digital petrography data collection software <http://www.stepsstage.com> [Accessed 28 Jun 2008].
- Weszka, J. S., 1978. A survey of threshold selection techniques. *Computer Graphics and Image Processing* 7 (2), 259–265.
- Williams, A., Wiltshire, R., Thomas, M., 1998. Sand grain analysis - image processing textural algorithms and neural nets. *Computers & Geosciences* 24 (2), 111–118.
- Witten, H., Frank, E., 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2nd edition.

- Yanowitz, S., Bruckstein, A., 1989. A new method for image segmentation. *Computer Vision Graphics and Image Processing* 46, 82–95.
- Yu, W., Chung, Y., Soh, J., 2004. Vignetting distortion correction method for high quality digital imaging. *17th International Conference on Pattern Recognition* 3, 666–669.
- Zawada, D. G., October 2003. Image processing of underwater multispectral imagery. *IEEE Journal of Oceanic Engineering* 28 (4), 583–594.
- Zhang, G., Jayas, D., White, N., 2005. Separation of touching grain kernels in an image by ellipse fitting algorithm. *Biosystems Engineering* 92 (2), 135–142.
- Zhang, Y. J., 1996. A survey on evaluation methods for image segmentation. *Pattern Recognition* 29 (8), 1335–1346.
- Zickler, T., Mallick, S., Kriegman, D., Belhumeur, P., August 2008. Color subspaces as photometric invariants. *International Journal of Computer Vision* 79 (1), 13–30.
- Zijdenbos, A., Dawant, B., Margolin, R., October 1991. A surface-fitting approach to the correction of spatial intensity variations in MR images. *Proceedings of the Annual International Conference of the IEEE* 12, 87–88.

Appendix A

Logistic Classifier - A Matab implementation

```
function [w posterior_prob] = logistic_regression(T,D,w)
%WEIGHTS = LOGISTIC_REGRESSION(T,D,w)
%T - The target matrix, each row corresponds to each data point in D.
%   It is a binary vector with
%   all elements zero except for the element k which equals 1 (i.e. if
%   the row is in class k).
%D - The training dataset (should have all ones in the last column, this
%   acts as a bias)
%w - Initial weights (one set of weights has to be all zero, this
%   represents the reference class)
%This function will return the updated weights for the approximation to
%the posterior probabilities given by exp/sum(exp).

[n,m]= size(D);
K = size(T,2)-1;
epsilon = 1e-10; %converged if difference between consecutive error is
                %less than epsilon

error = 1;
maxiteration = 2000;
ridge = 1e-8; %add this onto the diagonal of the ddE so that a
              %singularity does not occur

%if weights not entered as input parameter then initialise them
%automatically
if nargin < 3
    t = rand(K+1,m)>0.5;
    w = 0.00001*rand(K+1,m);
    w(t) = -w(t);
    w(K+1,:) = 0;
```

```

end

count = 0; %initialise variable to count the number of iterations
Eold = inf; %initialise starting cross-entropy error;

while count < maxiteration & error > epsilon
    %Update counter
    count = count + 1;
    %Matrix containing posterior probabilities. Element (n,k) will
    %represent the posterior probabilities for data point n
    %belonging in class k.
    posterior_prob = exp(D*w') ./ repmat(sum(exp(D*w'))',1,K+1);

    %Likelihood function is defined as  $p(T | w)$  where T is an N x K
    %built from t, where k is the total number of classes.
    %Make sure we do not take a log of zero
    likelihood = prod(diag(T*posterior_prob'));

    %The cross-entropy error is defined as the negative natural log of
    %the likelihood.
    Enew = -log(likelihood);
    error = abs(Eold-Enew);    %calculate change in cross-entropy error

    %It is this error that we wish to minimize, to do this we use the
    %Newton-Raphson method to find a minimum. Before this we must
    %calculate the gradient vector of the error function with respect
    %to each set of weights. This results in a matrix dE each row k
    %corresponding to the gradient vector with respect to the set of
    %weights k i.e. row k in w
    diff = posterior_prob - T;
    dE = zeros(K,m);
    for k = 1:K
        nextrow = zeros(1,m);
        for i = 1:n
            nextrow = diff(i,k)'*D(i,:) + nextrow;
        end
        dE(k,:) = nextrow;
    end

    %A Hessian matrix (ddE) is formed for each set of weights that
    %we wish to update
    for k = 1:K
        ddE = zeros(m,m);

```

```
for i = 1:n
    ddE = (posterior_prob(i,k).*...
           (1 - posterior_prob(i,k))*D(i,:))'*D(i,:) + ddE;
end
ddE = ridge*eye(m) + ddE; %The ridge is added so that we my
                           %approximate the inverse of ddE
w(k,:) = w(k,:) - (inv(ddE)*dE(k,:))'; %The weights are updated
end
Eold = Enew;
end
```