Morphological Granulometry for Texture Analysis

Mahmuda Khatun

Requirements for the degree of
Doctor of Philosophy

Department of Mathematics and Statistics
University of Strathclyde

November 2012

Acknowledgements

All kinds of praises and thanks are for Allah (God) alone, the beneficent, the merciful. The author invokes Allah's choicest blessing and peace for Prophet Muhammad (Sm), the messenger of Allah, the bearer of glad tidings and Warner for mankind.

My sincere gratitude to my supervisors, Dr. Alison Gray and Professor Stephen Marshall for their constructive guidance throughout the entire period of my PhD study. I wish to articulate my appreciation to my husband Dr. Nazrul Islam, moreover, I am thankful to my parents and my daughter (Munami) for their moral support and patience to accomplish this study.

I am indebted to the university for providing me with the University of Strathclyde scholarship which made this study possible. I appreciate the pleasant environment in the department of Mathematics and Statistics and the services provided by the office staff members. I am very grateful to E. Hines, S. Borah and M. Bhuyan for access to, and use of, the black tea granule images and Timothy Kelman for providing the hyperspectral images of Chinese teas.

Parts of Chapters 4–6 and 8 were published in Proceedings of 19th European Signal Processing Conference, 2011, Barcelona, Spain, 759-763 and parts of Chapters 7 and 8 were published in Proceedings of 15th Irish Machine Vision and Image Processing Conference, Dublin, 2011, 70-75.

Contents

List of Figures

ABSTRACT

This thesis concerns the analysis of digital texture images, using techniques from mathematical morphology and regression modelling for the classification of texture images. It investigates the use of granulometric moments, arising from the morphological pattern spectrum, as texture descriptors to predict evolution time or class label of texture images which evolve over time or follow an intrinsic ordering of textures. A cubic polynomial regression is used to model each of several granulometric moments as a function of time or class. These models are combined in a novel way and used to predict time or class.

The methodology was developed on synthetic images of evolving textures generated for the purpose, and then applied to classify a sequence of images of corroding metal to a point on an evolution time scale. Performance of the new regression approach is compared to that of several well established classifiers, namely linear discriminant analysis, neural networks and support vector machines (SVMs). The method was also applied to images of Indian black tea granules, which are ordered according to granule size. Better classification was achieved for both sets of images compared to previously published results for these images.

The performance of grey level co-occurrence matrix (GLCM) features from the synthetic images and both sets of real images was compared to that of granulometric moments, and it was found that granulometric moments provide much improved classification compared to GLCM features for such shape-based texture images. The performance of wavelet-based features from the Indian black tea images was also evaluated and was poorer than expected.

SVMs were generally found to be superior to the other classifiers.

The later part of this thesis concerns classifying hyperspectral images of Chinese teas. Several methods were compared for selection of appropriate spectral bands from these images. Principal component analysis and entropy proved to be the best band selection criteria in this application. GLCM features, wavelet-based features and wavelet-based GLCM features outperformed granulometric moments computed from the same set of bands. Calculating texture features from an optimum set of spectral bands gave better classification performance compared to the use of RGB (red, green and blue) or HSV (hue, saturation and value) colour representations or grey scale versions of the images.

Overview

Image processing is a multi-disciplinary research area, with contributions from engineers, computer scientists and statisticians. Digital image analysis has many applications, for example automating cell recognition and counting in microscope images, tracking objects in radar images for military applications, or reducing noise in satellite images before classifying areas on the ground into different types of land use. Digital images are represented as an array of numbers, each representing black or white (in a binary image), a shade of grey (in a grey scale image), or a multivariate measurement (for a colour or multi-sensor image).

In this project we investigate the use of statistics and mathematical morphology for classifying shape-based texture images. Mathematical morphology is a type of set theory, involving the interaction of specified test sets or structuring elements with a larger set. The result is another set. Applying different test sets to an image has different effects, useful for different purposes. These methods may be applied to either black and white or grey scale images.

We use morphological granulometries based on a series of openings at different scales, which produce a statistical probability distribution. The moments of this distribution provide a summary of texture information present in the image. Applying this to the foreground of an image provides information on shape and size of objects present, while applying it to the image background yields information on the spacing of the objects.

Most approaches to texture classification consider texture classes that have no intrinsic ordering, e.g. grass, wood, ceramic. In this work we are concerned with classification of a texture image to a point on an ordered scale of texture, using textures which evolve over time or follow an intrinsic ordering. In some texture analysis applications, such as monitoring of the degree of corrosion of machine parts, knowledge of the point reached on a time scale from no corrosion to severe corrosion is a vital aspect of industrial technology monitoring, related to safety issues as well as functionality (Choi and Kim (2005)). It is important to be able to measure the severity of corrosion as this can cause component weakness and potentially catastrophic failure.

The granulometric moments can be used to decide what type of texture an image contains, using statistical classification procedures. Gray et al. (2006), Gray et al. (2005), McKenzie (2004) and McKenzie et al. (2003) have shown the usefulness of this approach, especially when small numbers of images are available, and it has been implemented to enable classification of an image to a point in time when the texture can be considered as an evolving process.

A statistical approach to image texture classification is proposed here which uses polynomial regression. We model each texture feature as a function of lapsed evolution time or class label directly, using training images for which both the evolution parameters and the time state or class label are known. Polynomial regression models (one for each moment) are built using average moments from training images. A combined model is formed and the evolution time or class label of a new image is predicted from its observed features.

The main difference between our methodology and the methodology developed in McKenzie (2004) is that the latter used multiple regression to relate moments with the underlying parameters used to generate their synthetic images. The evolution of the artificial images depended explicitly on some evolution parameters which were set up as a known function of time before generating the images. The synthetic images were then used to relate granulometric moments to evolution parameters and back-prediction was used to predict the lapsed evolution time of a new image, based on the artificial image model and the observed granulometric moments from the new image.

In our case, the method was developed and evaluated using synthetic images for which the parameters do not change over time. Relating moments to time directly makes more sense for our synthetic images, as the parameters used to generate the images do not relate directly or explicitly to time. We build the model separately for the real images of interest rather than using the synthetic images-based model to predict time. Also, in practice any underlying parameters considered in the earlier approach may not necessarily be the most appropriate ones to use for real images of a particular type, so the approach developed here should be more robust.

Our methodology was developed on computer-generated grey scale images and is applied to two sets of real images, which consist of corrosion and Indian black tea granules. Several well established classifiers, i.e. support vector machines, linear discriminant analysis and neural networks are also used, using the same features, and their performance is compared with that of the regression classification approach. We also consider different types of features, including grey level co-occurrence features and wavelet-based features, and compare their relative performance for discriminating textures. The last part of this work concerns classifying Chinese teas based on analysis of hyperspectral images, and the benefits of using hyperspectral images over grey scale and colour images for classification is investigated.

The thesis structure is as follows: Chapter 1 reviews elementary concepts of digital image processing, with some examples. Morphological techniques, such as dilation, erosion, opening, closing, and applications of morphological techniques both for binary and grey scale images, are discussed in Chapter 2. An overview of texture analysis and description of the conventional texture feature extraction methods and classification approaches are given in Chapter 3. This chapter also contains details of how we generate synthetic binary and grey scale texture images which evolve over time. A detailed presentation of granulometries as the main feature extraction approach used in this thesis is in Chapter 4. The relationships of granulometric features to lapsed evolution time for synthetic images are investigated there. Chapter 5 presents the new regression-based texture classifier and the classification results of the synthetic images. Chapter 6 concerns classification of real corrosion images according to their evolution time and the performance of all classifiers is presented. This methodology is applied to another set of real images, of Indian black tea granules, to classify them according to granule size, in Chapter 7. Grey level co-occurrence features and wavelet-based features are extracted from the synthetic images as well as from both sets of real images and their performance for classifying those images is compared to that of granulometric moments in Chapter 8. Our methodology is then applied to hyperspectral images of six different types of Chinese teas to classify them in Chapter 9. Different band selection techniques are also compared there. Finally, Chapter 10 provides overall conclusions of the work in the thesis.

Chapter 1

Introduction to Image Analysis

1.1 Introduction

This chapter gives an introduction to basic image analysis, including mathematical morphology. Different image analysis operations are illustrated on example images, and carried out in the Matlab software package.

Image analysis is the extraction of meaningful information from digital images by image processing techniques. A digital image is an electronic representation of an image, usually as a two-dimensional finite array of numbers f_{ij} (Gonzalez and Woods (2008)) where i and j are finite integers indicating the row and column, and f_{ij} is the intensity or grey level of the image at that point (also finite). Each location is known as a *pixel* (picture element). So function f_{ij} indicates the pixel value in row i and column j (or vice versa).

Image analysis typically consists of five distinct stages that follow each other logically. These stages are display, filtering, segmentation, mathematical morphology and measurement, discussed briefly below.

1.2 Display

Display is the most basic step in image analysis. Different types of displays (binary display, grey scale display, or multivariate display) are appropriate, depending on the nature of the image.

1.2.1 Binary display

A binary image is the simplest type of display. In this case each pixel takes value 0 (for black) or 1 (for white), or sometimes 0 represents black and 255 represents

white. The value 0 or 1 can be stored in a *bit* of computer memory. A *byte* is the fundamental unit of computer memory which consists of 8 bits.

1.2.2 Grey scale display

In a grey scale image display, pixel values are used to specify the brightness with which pixels are illuminated on a computer screen, or how bright they appear on printed paper. Often the pixel values represent physical properties of the image. Showing larger pixel values as brighter intensity is simply a way to enable the spatial structure in the image to be seen. Grey scale pixels usually take an integer value between 0 and 255. The number of grey levels that can be handled is usually expressed in terms of a number of bits. One byte can represent $2^8 = 256$ grey levels.

1.2.3 Colour display

Human eyes are the main processors of the colour spectrum. The human eye contains three types of *cones* for colour processing (Gonzalez and Woods (2008)). One type has maximum sensitivity to the blue region of the colour spectrum, another type to the green region, and a final type to the red region. Light consisting of a single wavelength in the red region of the spectrum will be detected most strongly by the red sensitive cells, and we see it as red. Similarly, green sensitive cells detect green and blue sensitive cells detect blue. Different combinations of these three wavelengths produce a different colour sensation. To produce any colour, the intensities of the red, green, and blue light need to be present in specific proportions. This is referred to as the RGB (red, green, blue) system. Many colours can be obtained on a display unit from combinations of the three basic colour components. If each component can be displayed in 256 different intensities, there are $256^3 = 16,777,216$ different possible colours for a pixel (Gonzalez and Woods (2008)).

Figure 1.1 shows a colour, grey scale and a binary version of an example image of peppers. The original image is a colour image (RGB), which was converted to grey scale using the Matlab function 'rgb2gray' (which converts RGB images to grey scale by forming a weighted sum of the R, G, and B components as $0.2989 * R + 0.5870 * G + 0.1140 * B$). The grey scale image was then converted to binary using function *Im2bw* with an associated threshold. Otsu's thresholding (Gonzalez and Woods (2008)) is used here, which chooses the threshold to

minimise the intra-class variance of the black and white pixels (in this case the threshold was 0.3961). In the output image all pixels in the input image with luminance or intensity above the threshold are replaced with the value 1 (white) and all other pixels with value 0 (black).

(a) Colour display (b) Grey scale display (c) Binary display

Figure 1.1: Colour, grey scale and binary display of a peppers image of size 350×243 (from the Matlab Help System).

1.2.4 Image enhancement

The objective of image enhancement is to improve the interpretability or perception of information in images for human viewers, or to provide 'better' input for other automated image processing techniques. Image enhancement techniques can be divided into two broad categories:

1. Spatial domain methods, which operate directly on pixels, and
2. Frequency domain methods, which operate on the Fourier transform of an image.

We describe *contrast stretching*, and *histogram equalisation*, as spatial domain techniques for image enhancement.

Contrast stretching: Contrast stretching, often called *normalisation*, is a simple image enhancement technique that attempts to improve contrast in an image by 'stretching' the range of intensity values to span a desired range, e.g. the full range of pixel values that the image type allows, using a linear scaling function. As a result the 'enhancement' is less harsh than in histogram equalisation (see below). A simple way of contrast stretching described by Jain (1989) is as follows:

The first step specifies limits a and b over which image intensity values will be extended (for standard 8-bit grey scale images, these limits are usually 0 and 255). The original image is examined to determine the intensity limits (c and d)

in the unmodified image. Then for each pixel, the original value I_{in} is mapped to output value I_{out} as

$$I_{out} = (I_{in} - c)\left(\frac{b-a}{d-c}\right) + a.$$

The main drawback is that a single outlying pixel with either a very high or very low value can severely affect the effectiveness of the operation. One way to overcome this is to set c and d to, say the 5^{th} and 95^{th} percentiles of the original intensities, respectively.

Histogram equalisation: Histogram equalisation is a more sophisticated process of image enhancement than contrast stretching, as it may employ non-linear and non-monotonic transfer functions to map between input and output pixel intensity values, whereas contrast stretching is restricted to a linear mapping. Histogram equalisation ensures that all display intensities are approximately equally represented, with the objective of obtaining a new enhanced image with a uniform histogram. This can be achieved by using the normalised cumulative histogram as the grey scale mapping function.

Glasbey and Horgan (1994) describe histogram equalisation as follows: Let I denote the intensities ranging from $0, 1, \ldots, I_{\max}$ and $p(f)$ be the proportion of the pixel values $\leq f$. If the pixel values are displayed with intensity $I = I_{\max}p(f)$, then the proportion of pixels with display intensity $\leq I$ will be $i/i_{\max}$, leading to a linear cumulative distribution of intensities, i.e. a uniform distribution. Because of the discrete nature of the image histogram, the transformed image will have an approximate rather than an exact uniform distribution.

Zooming and reduction: If an image is too large to fit on a screen or too small to see finer detail, then image *reduction* or *zooming* can be used to shrink or enlarge the image. Image zooming gives control of the size of most images displayed. The simplest form of zooming is *pixel replication*.

Alternatively it may be of interest to reduce the size of an image for easier display or so it can be processed in less computer time, or occupy less storage memory. Although reducing an image involves some loss of data, it may sometimes be essential for display. *Pixel sampling* is one way to do this. Some techniques of zooming and reduction are described in Gonzalez and Woods (2008).

1.3 Filtering

Image filtering is another enhancement process. Image filters are the most common image processing operations, and create a new image by processing the pixels of an existing image. Different filters are used for different purposes, such as to remove noise, to smooth out high-frequency fluctuations or remove periodic trends of a specific frequency. Edge detection filters are of fundamental importance in image processing as edges characterise boundaries of objects of an image. Edge detecting an image significantly reduces the amount of data and filters out unnecessary information, while preserving the important structural properties of the objects in an image. Edges represent rapid transitions of intensity in the image.

Filters are broadly classified as *linear filters* and *non-linear filters*, although they can also be classified as (a) smoothing and noise reduction filters, (b) sharpening filters, and (c) edge-detection filters.

1.3.1 Linear filters

A filter is called linear if its output is derived as a linear combination of the pixels in the original image. An optimised linear filter possesses computational simplicity but cannot smooth without simultaneously blurring the edges. Linear filters can be used either for smoothing images to reduce noise, i.e. reducing the amount of intensity variation between one pixel and the next, or for edge-detection. Although there are many smoothing filters, we discuss only the moving average filter and the Gaussian filter.

Let f_{ij}, $i, j = 1, 2, \ldots, n$, denote the image pixel values, and g_{ij} denote the output of a linear filter of size $(2m+1) \times (2m+1)$ with specified weights w_{kl} for $k, l = -m, \cdots, m$. Then the output image is derived as :

$$g_{ij} = \sum_{k=-m}^{m} \sum_{l=-m}^{m} w_{kl} f_{i+k,j+l}, \tag{1.1}$$

for $i, j = (m+1), \ldots, (n-m)$ (Gonzalez and Woods (2008)). So g_{ij} is a weighted combination of the original pixel values, over a window of size $(2m+1) \times (2m+1)$, centred on pixel (i, j). Most linear filters use a window with an odd number of rows and columns. An even-sized window can be used, but in that case there will be a half-pixel displacement between the input and output image. For smoothing, all the w_{ij} are non-negative and usually sum to 1.

Moving average filter

The moving average filter is the simplest method of smoothing images. Its output is simply the average of the pixel values in the neighbourhood of pixels centred at that pixel, usually a square neighbourhood, given by equation (1.1) with $w_{kl} = 1/(2m+1)^2$, for an $(2m+1) \times (2m+1)$ neighbourhood, so that for a 3×3 square neighbourhood the weight will be $1/9$ (and $m = 1$).

Gaussian filter

The Gaussian filter can also be defined by equation (1.1) with weights specified by the probability density function of a bivariate Gaussian (normal) distribution with variance σ^2, that is

$$w_{ij} = \frac{1}{2\pi\sigma^2} \exp\left\{\frac{-(i^2+j^2)}{2\sigma^2}\right\},$$

for $i, j = -[3\sigma], \ldots, [3\sigma]$ for some specified positive value of σ^2 and where $[3\sigma]$ represents the integer part of $[3\sigma]$. Beyond this range the w_{ij} are near zero.

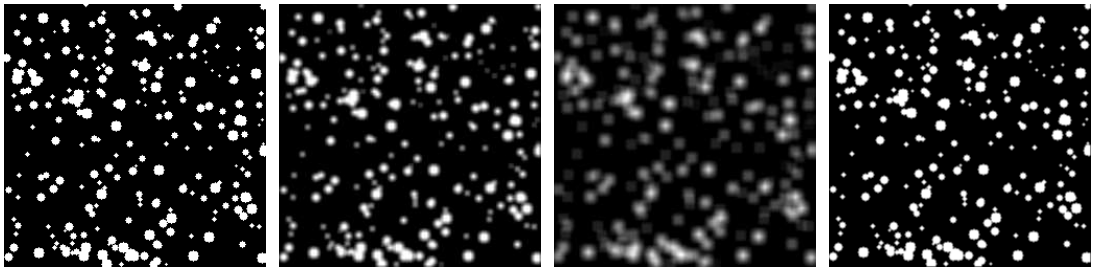
The Gaussian filter is a generalisation of the moving average filter. Four repeats of a moving average filter of size $(2m+1) \times (2m+1)$ approximates a single Gaussian filter with $\sigma^2 = \frac{4}{3}(m^2+m)$ (Wells (1986)). The Gaussian filter has the following advantages over the moving average filter:

1. Gaussian filters are separable and circularly symmetric (isotropic) but average filters are not. Average filters smooth further along diagonals then along rows and columns.

2. The weights in the Gaussian filter decay gradually to zero, while in the case of an average filter the weights have an abrupt cut-off which leaves discontinuities in the smoothed image.

Figure 1.2 shows the effect of two moving average filters of size 5 and 10, and a Gaussian filter of size 10 with $\sigma = 0.5$, on a binary image of disks of various radii. The effect is to smooth out the noise but also smooth out edges, so the result appears blurred. Comparing Figures 1.2(b) and (c) it is seen that the larger moving average filter blurs the image more than the smaller one. Comparing Figures 1.2(c) and (d), we see that the Gaussian filtering produces a better result as it is not as blurred. Such filters are called *low pass filters*, as the high frequency

components which give sharp changes in intensity are smoothed out, but lower ones 'pass' the filtering process.



(a) Binary image of size 256×256 (b) Effect of 5×5 average filter (c) Effect of 10×10 average filter (d) Effect of 10×10 Gaussian filter

Figure 1.2: Effect of moving average filters of different size and a Gaussian filter on a binary image of disks.

First order edge detector

A *high pass filter* on the other hand is used to sharpen an image or to detect edges. Linear edge detection filters use a combination of positive and negative weights to emphasise edges and the weights usually sum to zero. The first-derivative row and column filters detect edges in a given direction. The first-derivative row filter replaces each pixel value by the difference in pixel values in columns on either side of that location in the input image. The output will be larger in magnitude if the pixel values to the left and right of that pixel are quite different from each other. A set of weights, which operates over 3 rows, is given by

$$w = \frac{1}{6} \begin{Bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{Bmatrix}.$$

The output estimates the first row derivative, based on a Taylor series expansion of the image f_{xy} near (i, j). It can be shown that

$$\sum_{k=-1}^{1} \sum_{l=-1}^{1} w_{kl} f_{i+k,j+l} \approx \frac{\partial f_{ij}}{\partial x},$$

the first row derivative.

For the first derivative column filter the weights are just the transpose of the previous weights matrix, i.e.

$$w = \frac{1}{6}\begin{Bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{Bmatrix}.$$

Again this provides an approximation to the column derivative image. The first-derivative row filter produces non-zero values in response to vertical edges and the first-derivative column filter responds most to horizontal edges. However, these filters can emphasise noise as well as edges. One way to overcome this is to use a smoothing filter first. The effect of first-derivative row and column filters on Figure 1.2(a) are shown in Figures 1.3(b) and (c). The row filter highlights the image area where the intensity changes vertically and the column filter indicates changes in the horizontal direction.

Second order edge detector

Second order derivative filters are also known as Laplacian filters, which are isotropic and therefore responsive to edges in any direction. With the weights

$$w = \begin{Bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{Bmatrix}$$

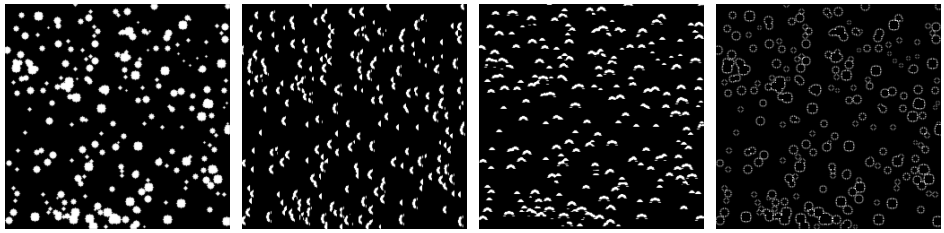
the output of a Laplacian filter approximates the Laplacian transform f of the input image (Glasbey and Horgan (1994)), i.e.

$$\sum_{k=-1}^{1}\sum_{l=-1}^{1} w_{kl} f_{i+k,j+l} \approx \frac{\partial^2 f_{ij}}{\partial x^2} + \frac{\partial^2 f_{ij}}{\partial y^2}. \tag{1.2}$$

Figure 1.3 shows the result of applying a Laplacian filter to Figure 1.2(a). Since the Laplacian filter emphasises noises as well as edges, a 5×5 Gaussian filter was applied to reduce noise first and then the Laplacian filter was applied ($\alpha = 0.8$) to the result to detect the edges (i.e. using a Laplacian of Gaussian filter). Comparing the output image with the original, the effect is to highlight areas in which intensity changes rapidly. The edges are finer than those produced by the first derivative filters and noise has been suppressed.

1.3.2 Nonlinear filters

A *nonlinear filter* is a filter whose output is a nonlinear function of the input. One practical reason to use nonlinear filters instead of linear filters is that linear filters



(a) Effect of 5×5 Gaussian filter (b) Effect of 3×3 row filter (c) Effect of 3×3 column filter (d) Effect of 3×3 Laplacian filter

Figure 1.3: Effect of row filter, column filter and Laplacian filter on a binary image of disks.

may be too sensitive to a small fraction of unusually large input observations (Macleod (1992)). Unlike linear filters, nonlinear filters can reduce noise and enhance edges at the same time.

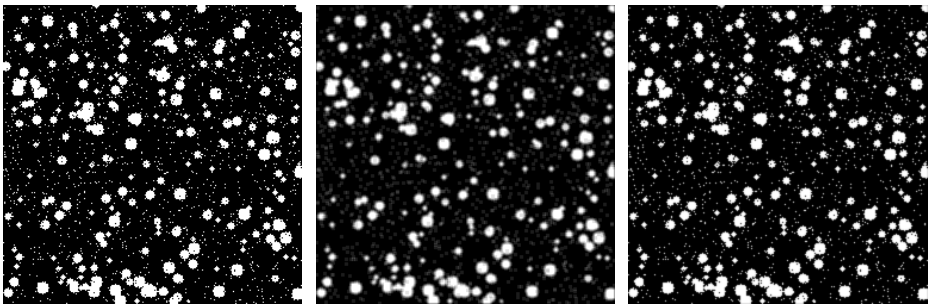
Nonlinear smoothing filters

A variety of non-linear smoothing filters have been developed. The *median filter* is one of the most widely used nonlinear smoothing filters. A median filter moves a window over an image and computes the output pixel value as the median value of all input pixels within the window. For an $n \times n$ image the output of the $(2m+1) \times (2m+1)$ median filter is

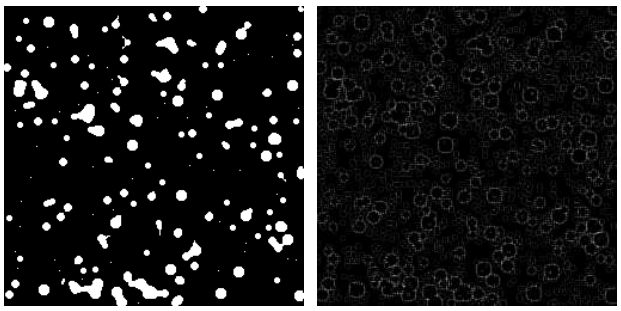
$$g_{ij} = \text{median}\{f_{i+k,j+l}\} \qquad \text{for} \quad i,j = (m+1), \cdots (n-m).$$

Median filtering is a simple and very effective noise removal filtering process which avoids blurring of edges. It is particularly good for removing shot noise (strong spike-like isolated values) (Dougherty and Astola (1994)).

The moving average filter, the Gaussian filter, and the median filter are all extensively used for noise reduction. To compare their effect with the Laplacian filter, we superimposed salt and pepper noise on Figure 1.2(a), then applied a 3×3 average filter, a 3×3 Gaussian filter with $\sigma = 0.5$ and a 5×5 median filter (Figure 1.4). Although the Gaussian filter performs better in image smoothing it does very little to reduce noise, and the median filter is much better for noise reduction. The Laplacian filter of size 3×3 was applied with $\alpha = 0.8$, and effectively detects the edges of the image objects.



(a) Noisy image of disks (b) Effect of 3×3 average filter (c) Effect of 3×3 Gaussian filter



(d) Effect of 5×5 median filter (e) Effect of 3×3 Laplacian filter

Figure 1.4: Effect of average, Gaussian, median, and Laplacian filters on a noisy binary image of disks.

Nonlinear edge-detection filters

Most of the edge-detection filters are based on the gradient method, which detects edges by looking for the maximum and minimum in the first derivative of the image (Kimmel and Bruckstein (2003)). There are various such filters, of which a few are described below:

Variance filter: The variance of the pixel values in a window will be larger if an edge is present than if it is not. Computing the variance or standard deviation in a window centred on each pixel will therefore help to highlight edges. There are several variations of the variance filter, such as the range filter and the Roberts filter.

Range filter: The output of a *range filter* is the difference between the maximum and minimum values of the pixel values in a window:

$$g_{ij} = \max\{f_{i+k,j+l} : k, l = -m, \cdots, m\} - \min\{f_{i+k,j+l} : k, l = -m, \cdots, m\}.$$

The range filter is an useful edge-detector.

Roberts' filter: The *Roberts' filter* uses the sum of absolute differences of diagonally opposite pixel values, i.e.

$$g_{ij} = |f_{ij} - f_{i+1,j+1}| + |f_{i+1,j} - f_{i,j+1}|,$$

so uses a 3×3 window. This filter is used as an edge enhancement operator, and especially determines edges at points where the gradient of the input image is largest.

Gradient filter

The gradient indicates the highest rate of change. The maximum gradient of the image f_{xy} near (i, j) is given by

$$\sqrt{\left(\frac{\partial f_{ij}}{\partial x}\right)^2 + \left(\frac{\partial f_{ij}}{\partial y}\right)^2},$$

so a *gradient* filter can be designed by replacing the partial derivatives above by their estimates. It is wise to use a smoothing filter before using a gradient filter to obtain a satisfactory edge-detection.

Prewitt's filter: *Prewitt's* edge detector filter is the standard un-weighted gradient filter used to detect edges based on applying a horizontal and vertical filter in sequence. Both filters are applied to the image and summed to form the final result. Prewitt's filter replaces the partial derivatives by their estimates,

$$\frac{\partial \hat{f}_{ij}}{\partial x} = \frac{1}{6}(f_{i-1,j+1} + f_{i,j+1} + f_{i+1,j+1} - f_{i-1,j-1} - f_{i,j-1} - f_{i+1,j-1})$$

and

$$\frac{\partial \hat{f}_{ij}}{\partial y} = \frac{1}{6}(f_{i+1,j-1} + f_{i+1,j} + f_{i+1,j+1} - f_{i-1,j-1} - f_{i-1,j} - f_{i-1,j-1})$$

(Glasbey and Horgan (1994)), using a 3×3 window size.

The filter is of the form:

$$g_{ij} = \sqrt{\left(\frac{\partial \hat{f}_{ij}}{\partial x}\right)^2 + \left(\frac{\partial \hat{f}_{ij}}{\partial y}\right)^2}, \tag{1.3}$$

and detects edges at all orientations.

Sobel's filter: *Sobel's filter* consists of two kernels which detect horizontal and vertical changes in an image. If both are applied to an image, the results can be used to compute the magnitude and direction of the edges in the image. It is a weighted version of Prewitt's filter, which is also based on the maximum gradient, but while estimating the partial derivatives more weight is given to the pixels nearest to (i, j), as follows:

$$\frac{\partial \hat{f}_{ij}}{\partial x} = \frac{1}{8}(f_{i-1,j+1} + 2f_{i,j+1} + f_{i+1,j+1} - f_{i-1,j-1} - 2f_{i,j-1} - f_{i+1,j-1})$$

and

$$\frac{\partial \hat{f}_{ij}}{\partial y} = \frac{1}{8}(f_{i+1,j-1} + 2f_{i+1,j} + f_{i+1,j+1} - f_{i-1,j-1} - 2f_{i-1,j} - f_{i+1,j-1}).$$

Again these are substituted in equation (1.3). Like Prewitt's filter, this is a filter of size 3×3. A detailed description is given in Glasbey and Horgan (1994).

Canny filter: Canny (1986) developed another edge detector filter which is also based on the gradient of the image intensity. It starts with linear filtering to compute the gradient of the image intensity distribution function and ends with thinning and thresholding to obtain a binary map of edges. The method uses two thresholds, to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges. Canny suggested using zero-crossings of second derivatives in the direction of steepest gradient to determine the position of the edges, i.e.

$$\cos^2 \varphi_{ij} \frac{\partial^2 f_{ij}}{\partial x^2} + \sin^2 \varphi_{ij} \frac{\partial^2 f_{ij}}{\partial y^2} + 2 \sin \varphi_{ij} \cos \varphi_{ij} \frac{\partial^2 f_{ij}}{\partial x \partial y}$$

where $\varphi_{i,j}$ is the direction of the maximum gradient and may be defined as

$$\varphi_{i,j} = \tan^{-1}\left(\frac{\partial \hat{f}_{ij}}{\partial y} / \frac{\partial \hat{f}_{ij}}{\partial x}\right)$$

In effect, the Canny filter thins the edges produced by the gradient filter.

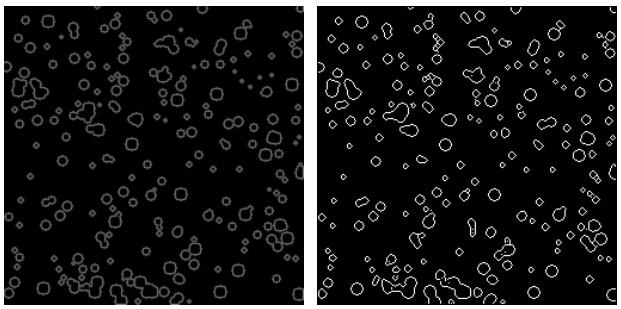
Figure 1.5 shows the result of applying Roberts' filter, Prewitt's filter, Sobel's filter, the Canny filter and a variance filter to the disks image. The Roberts' filter emphasises the edges better than the Canny filter and variance filter, but the variance filter detects individual object edges more successfully, although Canny is the best filter in general. The Prewitt's filter and Sobel filter have

a very similar effect, as they both are based on the maximum gradient. The Roberts filter seems to be superior to the other filters for edge detection in this case.

(a) Effect of 2×2 Roberts filter

(b) Effect of 2×2 Prewitt's filter

(c) Effect of 3×3 Sobel's filter



(d) Effect of 3×3 Canny filter

(e) Effect of 5×5 variance filter

Figure 1.5: Effect of Roberts, Prewitt's, Sobel's, Canny and variance filters on a binary noisy image of disks.

Morphological filter: Morphological filters provide a whole class of nonlinear filters, used for various purposes. They are mentioned in Section 1.5 as they are also be used at a later stage in the processing of an image, and some of these are also described and used extensively later in the thesis.

1.4 Segmentation

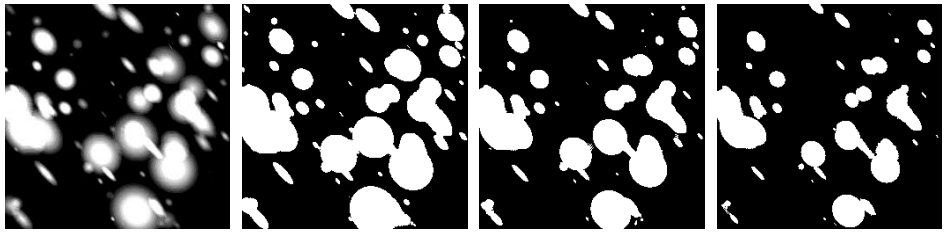
Image segmentation is a fundamental step in most applications of image analysis. Segmentation is a commonly used term for separating interesting and uninteresting objects, as well as distinguishing foreground from background (Reulke and Lippok (2008)). In computer vision, segmentation precedes the appropriate representation of the objects contained in an image and their classification according to specific features of interest (Ballard and Brown (1982)). A sensible segmentation is typically one in which pixels in the same category have similar grey

scale values and form a connected region, and neighbouring pixels which are in different categories have dissimilar values. Three general approaches to image segmentation are: *thresholding* techniques, *edge-based* methods and *region-based* techniques, described briefly below.

1.4.1 Thresholding

Thresholding is the simplest method of image segmentation. If the digital image is grey scale and a binary display is required, thresholding can be employed to produce a binary image (Shapiro and Stockman (2002)). This requires a threshold value t, used to categorise the pixel values of the foreground, namely pixel values on one side of t are displayed as black and those on the other side are displayed as white (Gonzalez and Woods (2008)). So pixel (i, j) takes one value if its grey scale value $f_{ij} \leq t$, otherwise it takes a second value. More than one threshold can also be used. For example if two threshold values are used, pixel (i, j) takes one value if $t_2 \leq f_{ij} \leq t_1$, otherwise it takes a second value.

Figure 1.6 shows a grey scale image of size 256×256 of ellipses of various sizes, and three segmentations using manually selected threshold values of 0.25, 0.5 and 0.75. Thresholding is carried out using Matlab command *im2bw*, in the Matlab Image Processing Toolbox, where the parameter level (ranging from 0 to 1) is set at the various threshold values. As the threshold increases, the number of object pixels reduces.



(a) Image of ellipses (b) Thresholding at level 0.25 (c) Thresholding at level 0.50 (d) Thresholding at level 0.75

Figure 1.6: Effect of thresholding a 256^2 grey scale image of ellipses, using various threshold values.

Thresholding is most effective when the intensity levels of the objects fall squarely outside the range of levels in the background. The threshold can be chosen empirically, by testing a range of values of t and determining which works best for identifying the object of interest, or automatically. There are several automatic algorithms for choosing the threshold, such as *Otsu's thresholding algorithm*, the *inter-means algorithm* and the *minimum error algorithm*. All of

these are histogram-based.

Otsu's thresholding: Otsu (1979) proposed a thresholding method which involves iterating through all the possible threshold values and calculating a measure of spread for the pixel levels to each side of the threshold. The aim is to find the threshold value for which the sum of foreground and background spreads is at its minimum, and to minimise the intra-class variance of the thresholded black and white pixels. This turns out to be the same as maximising the between-class variance. This operates directly on the grey level histogram. For each potential threshold T, the steps are:

1. Separate the pixels into two categories according to the threshold.
2. Find the mean of each category, $\bar{f}_1$ and $\bar{f}_2$.
3. Square the difference between the means, to get $(\bar{f}_1 - \bar{f}_2)^2$.
4. Multiply by the number of pixels in one category times the number in the other, to get $n_1 n_2 (\bar{f}_1 - \bar{f}_2)^2$.
5. Iterate the procedure until the optimum T is obtained, i.e. (4) is minimised.

This depends only on the difference between the means of the two categories, thus avoids having to calculate differences between individual intensities and the means of each category. The optimal threshold is the one that maximises the between-class variance (or, conversely, minimises the within-class variance).

Inter-means algorithm: Ridler and Calvard (1978) and Trussell (1979) proposed the inter-means method for choosing a single threshold. The steps are as follows:

1. Select an initial estimate of the threshold at t. A good initial value could be the median intensity of the image, such that

$$\sum_{k=0}^{t} h_k \geq \frac{1}{2} n^2 > \sum_{k=0}^{t-1} h_k,$$

where $h_k, k = 0, 1, \cdots, N$ represents the number of pixels in the image with grey scale value k, N is the maximum pixel value, and n^2 is the number of pixels in the $n \times n$ image.

2. Calculate the mean pixel value for both categories resulting from step 1. The mean pixel values for the first and second category respectively are

$$\mu_1 = \sum_{k=0}^{t} kh_k / \sum_{k=0}^{t} h_k \qquad \text{and} \qquad \mu_2 = \sum_{k=t+1}^{N} kh_k / \sum_{k=t+1}^{N} h_k.$$

3. Re-estimate t as half way between the two means, i.e.

$$t = [(\mu_1 + \mu_2)/2],$$

where [ ] denotes the integer part.

4. Repeat steps 2 and 3 until t stops changing value.

This threshold tends to divide the image histogram into two parts so that there are approximately equal numbers of pixels in both categories. Kittler and Illingworth (1986) suggested modifications to the above algorithm to overcome this tendency, as equal-sized categories may not be appropriate. The modified method assumes the image grey levels arise from a mixture of Gaussian distributions. The minimum error method is based on this modification.

Minimum error algorithm: The minimum error algorithm was proposed by Kittler and Illingworth (1986) which considers the image histogram as an estimate of the Gaussian probability distribution. It also assume that image foreground and background pixels follow Gaussian distributions with different means and variances. The minimum error algorithm also starts with an initial choice of threshold value and the steps are as follows:

1. Make an initial guess at a value for t.

2. Estimate the proportion p_1, mean μ_1, and variance σ_1^2 for pixels with values less than or equal to t (first category), by

$$p_1 = \frac{1}{n^2} \sum_{k=0}^{t} h_k, \qquad \mu_1 = \frac{1}{n^2 p_1} \sum_{k=0}^{t} kh_k \text{ and} \qquad \sigma_1^2 = \frac{1}{n^2 p_1} \sum_{k=0}^{t} k^2 h_k - \mu_1^2.$$

Similarly, estimate p_2, μ_2, and σ_2^2 for pixels with values $t+1, \ldots, N$ (second category).

3. A good estimate of t would be such that pixels with value k are allocated to the category with higher posterior probability, i.e. allocate to category 1

if

$$p_1\varphi_1(k) \geq p_2\varphi_2(k), \qquad \text{otherwise category 2,}$$

where $\varphi_i(k)$ represents the probability density function of a Gausian distribution with mean μ_i and variance σ_i^2, assumed to hold in category i.

4. By substituting the functional form of φ_i in the above inequality and taking logarithms we get

$$k^2\left(\frac{1}{\sigma_1^2}-\frac{1}{\sigma_2^2}\right)-2k\left(\frac{\mu_1}{\sigma_1^2}-\frac{\mu_2}{\sigma_2^2}\right)+\left(\frac{\mu_1}{\sigma_1^2}-\frac{\mu_2}{\sigma_2^2}+log\frac{\sigma_1^2p_2^2}{\sigma_2^2p_1^2}\right)\leq 0$$

or

$$k^2A-2kB+C\leq 0$$

where A, B, and C are:

$$A=\left(\frac{1}{\sigma_1^2}-\frac{1}{\sigma_2^2}\right), \quad B=\left(\frac{\mu_1}{\sigma_1^2}-\frac{\mu_2}{\sigma_2^2}\right) \quad \text{and} \quad C=\left(\frac{\mu_1}{\sigma_1^2}-\frac{\mu_2}{\sigma_2^2}+log\frac{\sigma_1^2p_2^2}{\sigma_2^2p_1^2}\right).$$

5. The threshold t is re-estimated in terms of the means and standard deviations of the two categories by

$$t=\left[\frac{B+\sqrt{B^2-AC}}{A}\right]$$

as the positive root of the quadratic equation $k^2A-2kB+C=0$.

6. Repeat steps 2 to 5 until t converges.

Majority filter: The *majority filter* smooths images by replacing pixel values with the majority (most frequent) pixel value in a neighbourhood. The steps, described by Glasbey and Horgan (1994), are as follows:

1. Make an initial guess at a value for t.

2. Segment the image, by thresholding, and store the result in an array g:

$$g_{ij}=1 \quad \text{if} \quad f_{ij}\leq t, \qquad \text{otherwise} \quad g_{ij}=2.$$

3. Apply a majority filter to g, and record the new labels in array g'. Therefore g_{ij} is set to the most common category in the set $g_{i+k,j+l}$ for $k,l=$

$-m, \cdots, m$. This is repeated for every value of i and j from $m+1$ to $n-m$.

4. Calculate the mean pixel value in category 1,

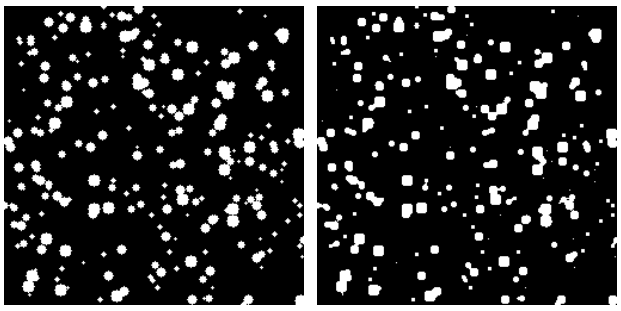
$$\mu_1 = \frac{1}{N_1} \sum \sum_{(ij):g'_{ij}=1} f_{ij},$$

where N_1 is the number of pixels in category 1. Similarly, calculate μ_2.

5. Re-estimate t as $[\frac{1}{2}(\mu_1 + \mu_2)]$.

6. Repeat steps 2 to 5 until t converges to a stable value, and then stop after step 3.

The majority filter, for example, can be implemented using Matlab 'bwmorph' function with the majority option. This filter replaces the value of a given pixel to 1 if at least five of its immediate eight-neighbours (see Section 3.4) are 1, and by 0 otherwise (Boland and Murphy (2001)). In effect it eliminates corners of the objects, therefore the disks in Figure 1.7 (a) turned to more or less square shapes in Figure 1.7 (b).



(a) Original image (b) Effect of a 3×3 majority filter

Figure 1.7: Effect of a majority filter on a 256^2 binary image of disks.

1.4.2 Edge-based thresholding

In edge-based segmentation an edge-detecting filter such as Prewitt's filter is applied to the image, and pixels are classified as edge and non-edge, depending on the filter output. Segmentation can be achieved by allocating to a single category all non-edge pixels that are not separated by an edge. However, edge-based methods centre around contour detection and the contour may not always be complete. There is a need to connect together broken contours lines, so this

approach is prone to fail in the presence of blurring. An algorithm such as the *connected components algorithm* can be used to label the objects defined by the edges, if these do provide complete contours.

Connected component algorithm: A pixel has 4 horizontal and vertical neighbours and 4 diagonal neighbours. Connectivity defines the relationship between pixels which specifies how many pixels are adjacent to a specific pixel. The relationship can be based on either 4- or 8-connectivity. A pixel is called 4-connected if it is connected to its 4 horizontal and vertical neighbours only but it is called 8-connected if the pixel is connected to its 4 horizontal and vertical neighbours as well as 4 diagonal neighbours. The connected component algorithm uses the concept of a pixel's connectivity.

The connected component algorithm scans an image from top to bottom and left to right in order to identify connected pixel regions, i.e. regions of adjacent pixels which share the same intensity value, e.g. k. The connected components labelling operator scans the image by moving along a row until it comes to a point, for example p, where p denotes the pixel to be labelled at any stage in the scanning process for which $k = 1$. For 8-connectivity, it examines the four neighbours of p, i.e. the neighbours (i) to the left of p, (ii) above it, and (iii and iv) the two upper diagonal terms. Based on this information, the labelling of p occurs as follows:

1. If all four neighbours are 0, assign a new label to p, else
2. if only one neighbour has $k = 1$, assign its label to p, else
3. if one or more of the neighbours have $k = 1$, assign one of the labels to p and record the equivalences.

After completing the scan, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. As a final step, a second scan is made through the image, during which each label is replaced by the label assigned to its equivalence classes. The algorithm is described in detail in Glasbey and Horgan (1994).

1.4.3 Region-based thresholding

There are several different approaches to this, namely *divisive* methods which split pixels into regions and *agglomerative* or merging algorithms which merge pixels into regions, and those that both split and merge. Typically, the image

is partitioned into connected regions by grouping neighbouring pixels of similar intensity levels. This can be done in several ways. Adjacent regions may then be merged under some criterion involving, perhaps, homogeneity or sharpness of region boundaries. The *watershed algorithm* is one of the simplest, yet powerful, region-based segmentation methods, now described.

Watershed segmentation

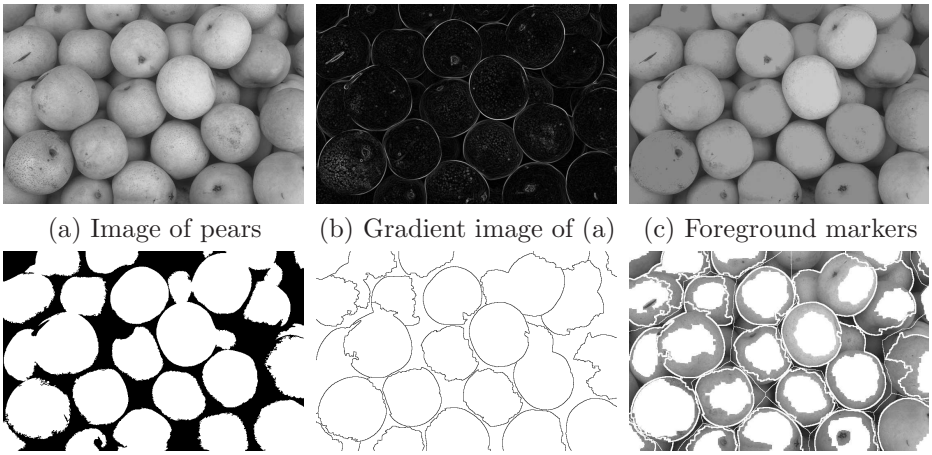
Watershed segmentation is a splitting algorithm which splits the image into regions. This is a way of automatically separating objects that touch each other by producing contours indicating their edges. The most intuitive formulation of the watershed transform is based on a flooding simulation (Dougherty and Lotufo (2003)). In this segmentation the input image is considered to be a topological surface or elevation map. The grey scale value at each pixel represents the height of the surface at that pixel, so the grey scale image can be thought of a 3-D surface.

Gonzalez and Woods (2008) suggested considering three types of points to interpret such a topological surface. These are: (a) points belonging to the regional minima; (b) points at which a drop of water, if placed at the location of any of those points, would fall with certainty to a single minimum; and (c) points at which water would be equally likely to fall to more than one such minimum. For a particular regional minimum, the set of points satisfying condition (b) is called the *catchment basin* or *watershed* of that minimum. The points satisfying condition (c) form crest lines on the topographic surface which are called *watershed lines*.

The objective of watershed segmentation is to produce watershed lines on the surface. The method can be visualised (Dougherty and Lotufo (2003)) by conceptually punching holes in each regional minimum of the image. Then the topography is slowly flooded from below by letting water rise from each regional minimum at a uniform rate across the entire image. When the rising waters coming from distinct minima are about to merge, a dam is built to prevent them merging. The flooding will eventually reach a stage when only the tops of dams are visible above the water-line, and these correspond to the watershed lines.

Figure 1.8(a) represents a 486×732 grey scale image of pears. The corresponding colour image is available in the Matlab help file and was converted to grey scale. The Matlab function *watershed* provides watershed segmentation by computing the gradient magnitude using the Sobel edge masks. Using morphological techniques, i.e. opening-closing reconstruction, it identifies foreground

markers and computes background markers by means of thresholding, and the watershed segmentation can be obtained from these. Figure 1.8(b) represents the filtered image where the gradient is high at the borders of the objects and low inside the objects, (c) is the opening-closing reconstructed image which allows the algorithm to find the foreground markers, (d) is the thresholded opening-closing reconstructed image, (e) shows the watershed segmentation of (a), and (f) contains the foreground markers, background markers and watershed segmentation superimposed on the original image.



(a) Image of pears (b) Gradient image of (a) (c) Foreground markers

(d) Background markers (e) Watershed segmentation of (a) (f) Compact visualisation

Figure 1.8: Watershed segmentation of a 486×732 image of pears, taken from the Matlab help file (using Matlab's functions).

Watershed from markers: The watershed transform is mainly applied to the morphological gradient image (see Section 2.6). This type of watershed requires a set of markers or seed points. Each marker must be placed on a sample region of the object to be segmented. The markers are used in the same way as the holes at the regional minima in the image surface, as described above. The approach is very appealing if one knows how to place the markers within the object to be segmented. Marker design in watershed transform is one of the most crucial steps for a successful segmentation (Dougherty and Lotufo (2003)). The watershed from markers can also be described as a flooding simulation process. It is an example of segmentation using seeded region growing using the markers as seed points.

1.5 Mathematical Morphology

Morphological operations can be employed for many purposes, including edge-detection, segmentation and enhancement of images. From the underlying morphological operations, an entire class of *morphological filters* can be constructed that can often be used in place of standard linear filters. Morphological filters are examples of nonlinear filters. This is a set-theoretic approach of image analysis providing a quantitative description of geometrical structure or texture in an image. We will discuss different aspects of mathematical morphology in Chapter 2, as it is used extensively in this thesis.

1.6 Measurement

Measurement is considered to be the final step of an image analysis. Measurements are usually taken from the image output from segmentation algorithms, which may also have been produced using morphological operators. In some applications, measurements can be obtained directly from the original image. Three general categories of measurements are size or length, measurement of shapes and boundary statistics (Gonzalez and Woods (2008)).

1.6.1 Measures of size

Two most common types of statistics used to describe size of objects are measurements of area and of distance. The *area* is just the number of pixels in an object,

$$area = \sum \sum_{(i,j)\in A} 1,$$

i.e. count 'one' for every pixel in A. If interest lies in the sum of pixel values within a specified region A, then the area (or volume) is given by

$$\sum \sum_{(i,j)\in A} f_{ij},$$

where f_{ij} denotes the grey scale value of pixel (i, j). Image area and volume remaining after sequences of morphological operations are also used extensively in this thesis.

Average breadth, which can be obtained by dividing the area of an object by its length, is another way to quantify objects.

Moments can be used to specify the location and spatial distribution (shape) of an object. The $(k,l)^{th}$ order moment is defined as

$$\mu_{kl} = \sum \sum_{(i,j)\in A} i^k j^l \qquad \text{for } k,l = 0,1,2,\ldots$$

If k and l are both zero, we obtain the zeroth-order moment μ_{00}, which is the area. The *centroid* or *centre* of gravity can be used to specify the location of an object in the image, defined in terms of the moments, as

$$centroid = \left(\frac{\mu_{10}}{\mu_{00}}, \frac{\mu_{01}}{\mu_{00}}\right),$$

where μ_{00} and μ_{10} are the $zero^{th}$ and first-order moments.

The second order central moments are defined as

$$\mu'_{20} = \mu_{20} - \frac{\mu_{10}^2}{\mu_{00}}, \quad \mu'_{02} = \mu_{02} - \frac{\mu_{01}^2}{\mu_{00}} \quad \text{and} \quad \mu'_{11} = \mu_{11} - \frac{\mu_{10}\mu_{01}}{\mu_{00}}.$$

These measure how dispersed the pixels in an object are from their centroid. The moment μ'_{20} measures the spread of the object over rows, μ'_{02} measures its spread over columns, and μ'_{11} is a cross-product term representing spread in the direction in which both row and column indices increase. These central moments are not rotationally invariant. Modified versions are available, for use when the orientation of objects does not matter (see Glasbey and Horgan (1994)).

Distance measurements between two specified pixels (i,j) and (k,l) can be defined by various different measures, including the traditional *Euclidean distance* $\sqrt{(i-k)^2+(j-l)^2}$, *Chessboard distance* $\max(|i-k|,|j-l|)$, and *City-block distance* $|i-k|+|j-l|$.

1.6.2 Measures of shape

The most commonly used shape statistic is a measure of *compactness*, defined as the ratio of the area of an object to the area of a circle with the same perimeter, i.e.

$$compactness = 4\pi \frac{area}{(perimeter)^2},$$

where perimeter is the conventional measure of boundary length.

Another measure of compactness is given in terms of the moments as

$$compactness = \frac{\mu'_{20} + \mu'_{02}}{\mu_{00}^2}.$$

This measures how dispersed the pixels in an object are from their centroid, compared to the most compact arrangement of the pixels.

The shape of an object can be measured in terms of *elongation*, defined by the ratio of object length to its breadth, i.e.

$$elongation = \frac{length}{breadth}.$$

There are other measures of shapes, such as convexity and roundness. For details, see Glasbey and Horgan (1994).

1.6.3 Boundary statistics

Boundary statistics provide information regarding the boundary of image objects. Gonzalez and Woods (2008) discussed different boundary statistics, namely an ordered set of boundary pixels, chain code and Fourier descriptors. An ordered set of boundary pixels can be generated by determining two points on the boundary of an object, where the second point is the next pixel location along the boundary in an anti-clockwise direction, say, after the first point. Then we choose a search direction between 0 to 3, if we only consider 4-connected neighbours, until we find the next boundary pixel and so on. Chain code, consists of the starting location and a list of subsequent directions $d_1, d_2, \ldots, d_N$, to provide a more compact representation of all the information in a boundary or edge. Another way of obtaining boundary statistics is to express the x and y coordinates separately as weighted sums of sine and cosine terms. The weights are the Fourier coefficients, which can be used to compute measures of shape. Using fewer terms in the sums gives a more approximate boundary representation and using more terms gives a more accurate representative.

1.7 Conclusion

This chapter provides a brief description of basic digital image processing techniques, especially different types of filtering and segmentation. Among various possible segmentation approaches we use Otsu's thresholding in Chapter 7. The next chapter describes mathematical morphology in detail.

Chapter 2

Mathematical Morphology

In this chapter, we describe mathematical morphology techniques in detail, as these are used extensively in the work of this thesis.

2.1 Introduction

Mathematical morphology is a set-theoretic approach to image analysis (Matheron (1975), Heijmans (1979), Serra (1983) and Dougherty and Lotufo (2003)), which uses concepts from set theory, geometry, and topology to analyse geometrical structures in images. It is a set of tools for extracting image components that are useful in the presentation and description of region shape, such as boundaries, skeletons, and convex hulls (Gonzalez and Woods (2008)). According to Petrou and García-Sevilla (2006), mathematical morphology is a collection of non-linear processes which can be applied to identify or remove image details smaller than a pre-determined geometric structure. It provides many useful pre- and post-processing techniques, especially in edge thinning and pruning (Giardina and Dougherty (1987)).

The morphological approach is generally based upon the analysis of an image in terms of some predetermined geometric shape, known as a *structuring element* (SE) (Giardina and Dougherty (1987)), and on how the image interacts with the structuring element.

2.2 Morphological Techniques

Different morphological operators and structuring elements provide different results which are useful for various purposes. Mathematical morphology provides a number of important operations for analysing an image such as *dilation*, *erosion*,

opening, *closing*, and many others. The applications of morphology to binary and grey scale images are described separately here. We focus first on the SE, as it determines the effect of the morphological operator on the image.

2.2.1 Structuring elements

In mathematical morphology, a SE is a pre-determined geometric shape, used to probe or interact with a given image, with the purpose of drawing conclusions as to how this shape fits or misses shapes in the image. The binary image and SE are both considered as sets of 0s and 1s. Typically there are 1s in positions corresponding to elements within the set, and 0s elsewhere. The locations of the 1s are called the *neighbourhood* defined by the SE.

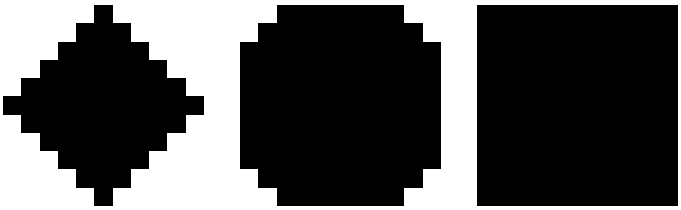
The size and shape of the SE greatly affect the results of morphological operations. For example, the results of applying any morphological operation to an image of stars using a square SE and a line SE will be quite different. Hence, determining the best size and shape of a SE is of crucial importance. However, the overall selection of a SE depends upon the geometric shapes we attempt to extract from the image data. For example, if we are dealing with biological or medical images, which contain few straight lines or sharp angles, a circular SE or a disk is an appropriate choice. When extracting shapes from geographic aerial images of a city, a square or rectangular SE will allow the extraction of angular features from the image better than any of the line SEs.

Whatever the shape of the SE, a reference point must be selected for it, as the morphological operations place the reference point of the SE at every single pixel of the image to observe the interaction of the image and the SE at that pixel (Gonzalez and Woods (2008)). Usually the central pixel of a SE is taken as the reference point if the SE is a symmetric shape. For other shapes the reference point needs to be defined. The location of the reference point is important. Two SEs which have the same shapes but different reference points can extract different information.

Figure 2.1 shows diamond, disk and square SEs, where the diamond and disk are of radius 5 pixels and the square is of width 5 pixels, where width=0.5∗(base length-1).

2.2.2 Dilation

Dilation is one of the fundamental operations of morphology, which dilates an object with a pre-determined object (SE). Dilation of an image is the process of



(a) Diamond (b) Disk (c) Square

Figure 2.1: Some commonly used SEs

placing the reference pixel of a SE on every pixel of the input image and marking every pixel covered by the SE as an output image pixel (Petrou and García-Sevilla (2006)). By adding the extra pixels to the input image it expands the size of objects on the scale of the SE neighbourhood.

The effect of dilation on a binary image is to enlarge the boundaries of regions of foreground pixels, so these areas grow in size, while holes within those regions become smaller. It fills in holes and broken areas, connects areas that are separated by spaces smaller than the size of the SE and adds pixels to the perimeter of each image object.

The mathematically favoured definition of dilation as stated in Giardina and Dougherty (1987), is as follows: Let A and B be two sets in two-dimensional Euclidean space $\mathbb{R}^2$, where A is the set to be dilated and B is the SE. The dilation of A by B is denoted by $A \oplus B$, and defined as

$$A \oplus B = \{w \in \mathbb{R}^2 : w = a + b, \text{for } a \in A \text{ and } b \in B\}, \tag{2.1}$$

where the plus sign refers to vector addition. This is equivalent to *Minkowski addition*. Given two sets A and x in $\mathbb{R}^2$, their Minkowski addition, denoted by $A \oplus x$, is the set

$$A \oplus x = \{a + x : \ a \in A\}. \tag{2.2}$$

where the plus sign refers to vector addition (Tuma and Walsh (1998)). Considering x as a vector in the plane, $A + x$ is A translated along the vector x.

Dougherty and Lotufo (2003) mentioned two equivalent definitions of dilation of A by B as

$$A \oplus B = \bigcup_{b \in B} A_b, \tag{2.3}$$

i.e. dilation is the process of translating the input image A by all points in the SE and taking the union. It also can be defined in terms of translating the SE

by all points in the image, as

$$A \oplus B = \bigcup_{a \in A} B_a. \tag{2.4}$$

Using the definition of dilation given by (2.4) is computationally time consuming as it involves translation for every point in the input image, whereas the definition given by (2.3) requires translation for every point in the SE.

Gonzalez and Woods (2008) define the dilation of a set A with another set B, where both of the sets are in $\mathbb{Z}^2$, as the set of all translations of the reflection of the set B about its origin by z (rotation by 180° of B) and then taking the intersection of this shifted result with the set A such the resulting set is non-empty. Mathematically, the definition is as:

$$A \oplus B = \{z : (\hat{B})_z \cap A \neq \emptyset\}. \tag{2.5}$$

Interpretation of (2.5) establishes the following equivalent definition of dilation, as

$$A \oplus B = \{z : [(\hat{B})_z \cap A] \subseteq A\}. \tag{2.6}$$

Since dilation thickens the objects in a binary image, it is referred to as an *extensive* operator, i.e.

$$A \oplus B \supseteq A.$$

Although the basic morphological operations are available in many image analysis software packages, in this thesis we have used Matlab. The Matlab definition of dilation, which is applicable for both grey level and binary images, is as follows:

$$A \,\text{dilated by}\, B = \{x : x = \text{max pixels in}\, A \,\text{in neighbourhood}\, B_x\}, \tag{2.7}$$

which works if the foreground of a binary image and SE are denoted as 1 and the background as 0, and B_x is the SE placed with its reference pixel at pixel x or the translation of B by x.

2.2.3 Erosion

Like dilation, *erosion* is also a basic operation of morphology, and entire morphological operations are based on these two primitive operations. Erosion operates by placing the reference pixel of the SE on every pixel of the input image and keeping only those pixels at which the SE fits fully inside the input image (Petrou

and García-Sevilla (2006)). Whereas dilation is a thickening operation, erosion shrinks or thins the image under analysis, as it removes some of the input image pixels where the SE does not fit fully.

Erosion generally decreases the sizes of objects and removes objects which are smaller than the applied SE. With binary images, erosion completely removes objects smaller than the SE and removes perimeter pixels from larger image objects.

The set-theoretic mathematical definition of erosion, according to Gonzalez and Woods (2008), is as follows: for A and B as defined above, the erosion of A by B is denoted by $A \ominus B$, and defined as

$$A \ominus B = \{w \in \mathbb{R}^2 : w = a - b, \text{for some } a \in A \text{ and } b \in B\}. \tag{2.8}$$

Equation (2.8) is known as *Minkowski subtraction*, defined as follows:

Given two sets A, and B in $\mathbb{R}^2$, their Minkowski subtraction, denoted by $A \ominus B$, is the set

$$A \ominus B = \bigcap_{b \in B} A + \hat{B}, \tag{2.9}$$

where $\hat{B}$ is the reflection of B, defined as the rotation of B by 180° about its reference pixel, and A is translated by every element of $\hat{B}$ and then the intersection is taken.

Dougherty and Lotufo (2003) define the erosion of set A by the SE B (a set usually smaller than A) as

$$A \ominus B = \{x : B_x \subset A\}. \tag{2.10}$$

Erosion can be defined in terms of dilation, as they are dual operations. The erosion of set A by set B is defined as the complement of the dilation of set A^c by the reflection of set B, i.e.

$$A \ominus B = (A^c \oplus \hat{B})^c. \tag{2.11}$$

Gonzalez and Woods (2008) define the erosion of set A by set B as the set of all points z such that B translated by z is contained in A. Mathematically, the definition is as:

$$A \ominus B = \{z : (B)_z \subseteq A\}. \tag{2.12}$$

Since equation (2.12) establishes that the eroded set contains those points of B which are in A, it is equivalent to say B is not overlapping with the complement

of A, so another equivalent way to define erosion of A by B is

$$A \ominus B = \{z : (B)_z \cap A^c = \emptyset\}. \tag{2.13}$$

Since erosion shrinks or thins objects in a binary image it is an *anti-extensive* operator, i.e.

$$A \ominus B \subseteq A.$$

2.2.4 Opening

Opening is the operation by which we dilate an image after we have eroded it. This is an important morphological operation, since it is useful for smoothing the contour of an object, breaking narrow gaps, and eliminating thin protrusions or spikes on an object (see Gonzalez and Woods (2008)). We use this operation repeatedly in later chapters.

The opening of a binary set A by the SE B is defined to be the union of all translations of B that are a subset of A. In effect, B is moved inside A and the opening consists of all the points of A that lie in some translated copy of B (Dougherty and Pelz (1991)), and is denoted by $A \circ B$. Notationally,

$$A \circ B = (A \ominus B) \oplus B. \tag{2.14}$$

Thus the opening of set A by B is the erosion of A by B, followed by a dilation of the result. Gonzalez and Woods (2008) point out the geometric fitting property of the opening operator, which leads to the following definition of opening:

$$A \circ B = \bigcup \{(B)_z : (B)_z \subseteq A\}. \tag{2.15}$$

Equation (2.15) indicates that opening of the set A by B is the union of all translations of B by z that fit into A.

2.2.5 Closing

Closing is the erosion of an image after it has been dilated. This also tends to smooth sections of contour but, as opposed to opening, it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in contours (Gonzalez and Woods (2008)).

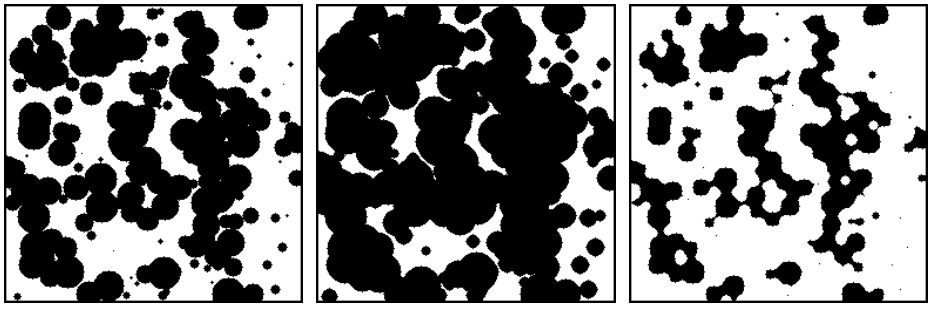
Mathematically, the closing of the set A by the SE B is denoted by $A \bullet B$ and

is defined as

$$A \bullet B = (A \oplus B) \ominus B. \tag{2.16}$$

So the closing of set A by B is to dilate A by B, then erode the result.

Figure 2.2 shows the effect of dilation of a binary image of size 261×261 of disks of various radii ranging from 1 to 15, using a disk SE of radius 4. In the dilated image the foreground pixels have increased by enlarging the objects as well as filling gaps smaller than radius 4. This shows the extensive nature of dilation. As erosion is anti-extensive, we see in the eroded image that the objects in the original image have shrunk. Since opening is erosion followed by dilation, we dilated the eroded image. In the opened image all disks smaller than radius 4 have disappeared and the rest remain. For closing, we eroded the dilated image. This fills the gaps between objects and adds extra foreground pixels by enlarging the object boundaries.



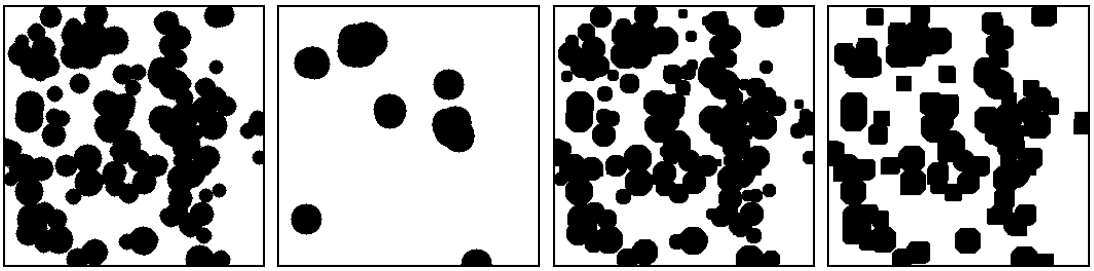
(a) Image of disks (b) Dilation of (a) (c) Erosion of (a)

(d) Opening of (a) (e) Closing of (a)

Figure 2.2: Effect of dilation, erosion, opening and closing of a 261^2 binary image of disks using a disk SE of radius 4.

The shape and size of image primitives (basic shape in the image) and SEs greatly affect the results of any morphological techniques. For example we show the effect of opening of the disk image in Figure 2.2 (a) using disk SE of radius 7 and 15 and square SE of width 7 and 15 in Figure 2.3. Opening the input image by a disk of radius 7 removes any disk smaller than radius 7. Similarly opening by a disk SE of radius 15 removes all disks smaller than radius 15. To compare the

effect of opening by a differently shaped SE than the image primitives, we opened the same image using a square SE of width 7 and 15, where width=0.5∗(base length-1). Opening disks by a smaller disk SE removes all disks smaller than its size, whereas opening by a smaller square SE removes small objects but also adds pixels to the corners of remaining ones, making the objects more square shaped.



(a) Opening by a disk of radius 7 (b) Opening by a disk of radius 15 (c) Opening by a square of width 7 (d) Opening by a square of width 15

Figure 2.3: Effect of opening a binary 256^2 image of disks using a disk and a square SE.

2.3 Properties of the Morphological Techniques

Duality of dilation and erosion Dilation and erosion are duals of each other with respect to set complement and reflection, where the complement of the set A, denoted by A^c, consists of the elements not in A, and reflection of A is defined as

$$\hat{A} = \{w : w = -a, \text{ for } b \in B\}.$$

The dilation of set A by B is the complement of the erosion of A^c by the reflection of the SE B (its rotation by 180^o about its reference pixel) and vice-versa. That is

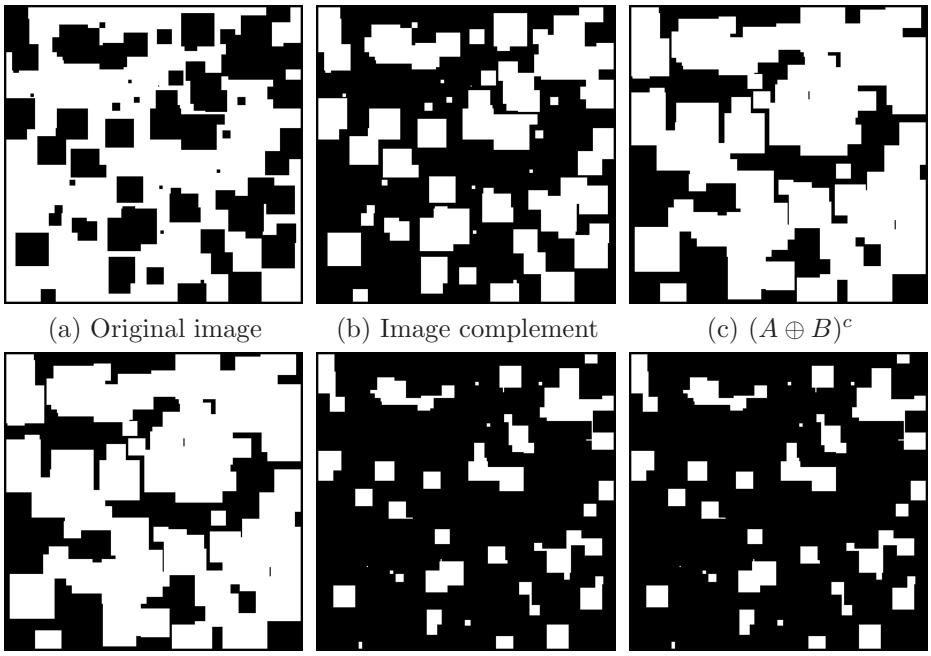
$$(A \oplus B)^c = A^c \ominus \hat{B},$$

and

$$(A \ominus B)^c = A^c \oplus \hat{B}.$$

The duality property is useful, particularly when the SE is symmetric, that is $\hat{B} = B$, where $\hat{B}$ is the reflection of B. Then we can obtain the dilation of an image A by B simply by eroding the background (complement) of A (Figure 2.4 shows an example) with the same SE B.

Duality of opening and closing: Like dilation and erosion, opening and closing are also duals of each other with respect to set complementation and



(a) Original image (b) Image complement (c) $(A \oplus B)^c$

(d) $A^c \ominus \hat{B}$ (e) $(A \ominus B)^c$ (f) $A^c \oplus \hat{B}$

Figure 2.4: Duality of dilation and erosion illustrated on a 256^2 binary image of squares of different width, using a square SE of width 2, where width=0.5∗(base length-1).

reflection. The opening of set A by B is the complement of the closing of A^c by the reflection of the SE and vice-versa. That is,

$$(A \circ B)^c = A^c \bullet \hat{B},$$

and

$$(A \bullet B)^c = A^c \circ \hat{B},$$

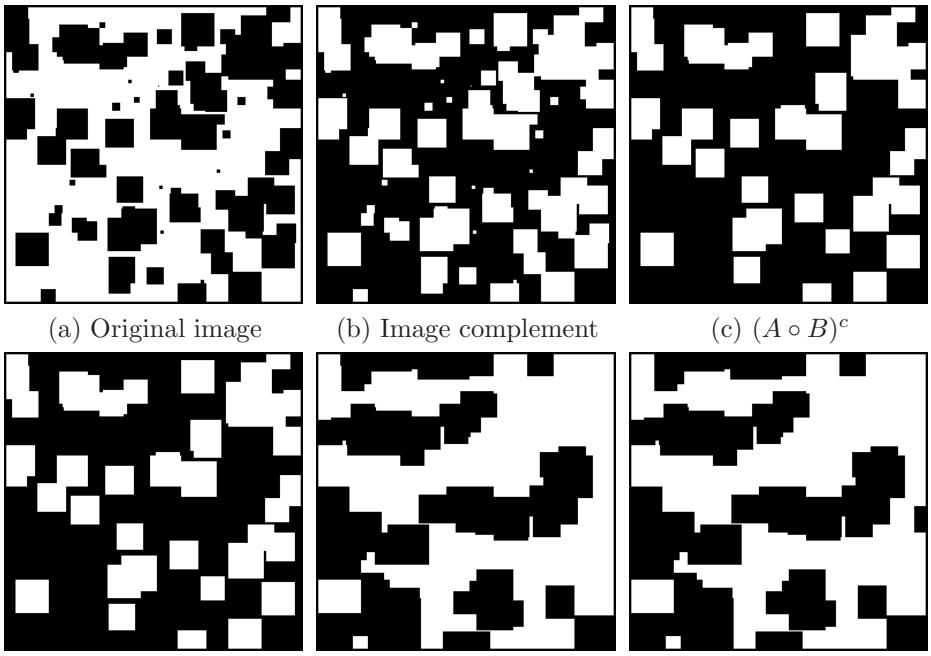
where A^c is the complement of A and $\hat{B}$ is the reflection of B. The duality of opening and closing is shown in Figure 2.5.

Other properties: The basic morphological operators follow some other important properties (Gonzalez and Woods (2008), Glasbey and Horgan (1994)), listed below:

1. Dilation is commutative, extensive and associative, i.e.

$$A \oplus B = B \oplus A,$$

$$A \oplus B \supseteq A,$$



(a) Original image (b) Image complement (c) $(A \circ B)^c$

(d) $A^c \bullet \hat{B}$ (e) $(A \bullet B)^c$ (f) $A^c \circ \hat{B}$

Figure 2.5: Duality of opening and closing illustrated on a 256^2 binary image of squares of different width, using a square SE of width 2, where width=0.5∗(base length-1).

and

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C.$$

2. Both dilation and erosion satisfy scalar multiplication, and erosion is anti-extensive, i.e.

$$t(A \oplus B) = tB \oplus tA$$

and

$$t(A \ominus B) = tB \ominus tA,$$

where $t > 0$ is a scalar multiple, and

$$A \ominus B \subseteq A.$$

3. Dilation and erosion are both translation invariant, i.e.

$$(A_x \oplus B) = (A \oplus B)_x$$

and

$$(A_x \ominus B) = (A \ominus B)_x.$$

4. Dilation and erosion are both monotonically increasing operators, i.e. if $A_1 \subset A_2$, for fixed B

$$(A_1 \oplus B) \subseteq (A_2 \oplus B)$$

and

$$(A_1 \ominus B) \subseteq (A_2 \ominus B).$$

5. Eroding A by $B \oplus C$ is the same as eroding A by B, then eroding the result by C:

$$A \ominus (B \oplus C) = (A \ominus B) \ominus C.$$

6. Dilating A by two disks B_r and B_s, one after another, is equivalent to dilating with one larger disk whose radius is the sum of the radii r and s of the smaller disks:

$$(A \oplus B_r) \oplus B_s = A \oplus B_{r+s}$$

if B_r and B_s are convex.

7. Eroding A by two disks B_r and B_s, one after another, is equivalent to eroding with one larger disk whose radius is the sum of the radii r and s of the smaller disks:

$$(A \ominus B_r) \ominus B_s = A \ominus B_{r+s}$$

if B_r and B_s are convex.

8. Dilation and erosion satisfy certain distributivity properties with respect to the set theoretic operations. Dilation distributes over union, but it does not distribute over intersections:

$$A \oplus (B_1 \cup B_2) = (A \oplus B_1) \cup (A \oplus B_2).$$

Erosion distributes from the right over intersection, i.e.

$$(A_1 \cup A_2) \ominus B = (A_1 \ominus B) \cap (A_2 \ominus B).$$

Relative to union, erosion satisfies left anti-distributivity, i.e.

$$A \ominus (B_1 \cup B_2) = (A \ominus B_1) \cap (A \ominus B_2).$$

9. Opening A by two disks B_r and B_s, one after another, is equivalent to

opening with the larger disk only, i.e.:

$$(A \circ B_r) \circ B_s = A \circ B_{max(r,s)}$$

if B_r and B_s are convex.

10. Opening and closing are increasing, i.e. if A_1 is a sub-image of A_2 then for any SE B

$$(A_1 \circ B) \subseteq (A_2 \circ B)$$

and

$$(A_1 \bullet B) \subseteq (A_2 \bullet B).$$

11. The opening and closing are both translation invariant, i.e.

$$(A_x \circ B) = (A \circ B)_x$$

and

$$(A_x \bullet B) = (A \bullet B)_x.$$

12. Opening is anti-extensive, i.e. $(A \circ B)$ is a sub-image of A and closing is an extensive operator, i.e. $(A \circ B) \supseteq A$, therefore the relation $A \circ B \subseteq A \subseteq A \bullet B$ always holds.

13. Opening and closing are both *idempotent*, i.e.

$$(A \circ B) \circ B = A \circ B$$

and

$$(A \bullet B) \bullet B = A \bullet B.$$

This means they have a one-off effect.

2.3.1 Hit-or-miss-transform

The hit-or-miss transform provides powerful sets of tools for various applications in image processing (Dougherty and Lotufo (2003)), including finding shapes in an image. To describe the hit-or-miss transform we need to consider the SE as a set with two components, i.e. $B = (B_1 \cup B_2)$, where B_1 is the set formed from elements of B associated with an object and B_2 is the set of elements of B associated with the corresponding background, and B_1 and B_2 are assumed to be

disjoint (Dougherty and Lotufo (2003)). The effect of the hit-or-miss-transform is the same as to erode the image A with B_1 and the complement of the image (A^c), with B_2, and to take the intersection of those two eroded sets. It identifies all the pixels in the image at which B_1 matches with the image foreground and at which B_2 matches to the background. Notationally, the hit-or-miss-transform is given by

$$A \circledast B = (A \ominus B_1) \cap (A^c \ominus B_2).$$

A point will be recognised as a 'hit' by the hit-or-miss transform if and only if B_1 translated to the point fits inside A, and B_2 translated to the point fits outside of A. Since the hit-or-miss operator operates by fitting the SE into both the image and the complement of the image, it probes the relation between the image and its complement relative to the SE.

2.4 Application of Binary Morphological Techniques

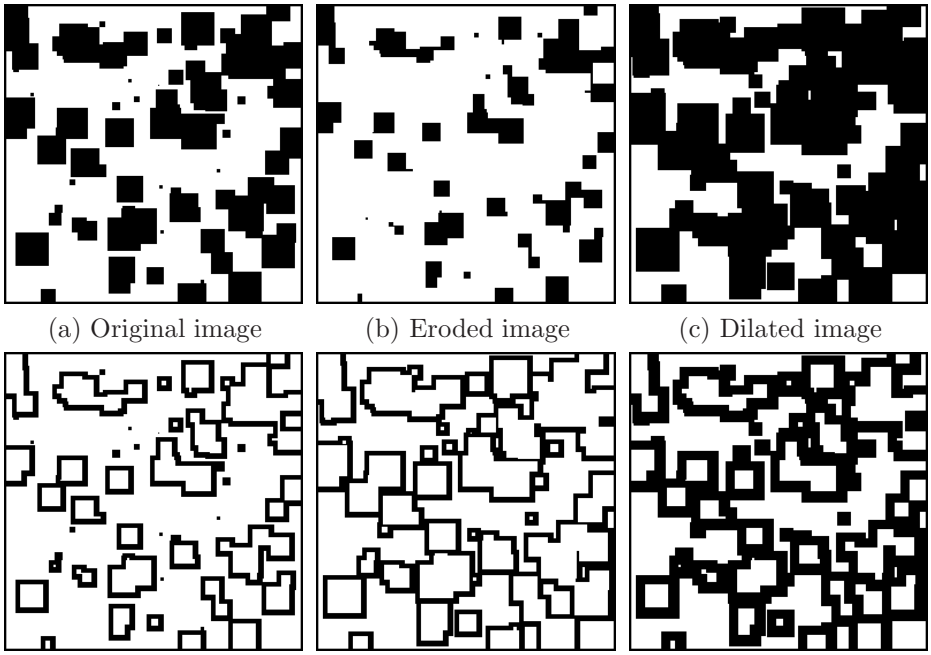
Morphological operations are widely used in digital image processing, mainly as a result of work by Matheron (1975) and Serra (1983). In general, opening followed by closing has a smoothing effect. When there is additive and subtractive noise, opening removes additive background noise and closing fills subtractive noise in the foreground (Dougherty and Lotufo (2003)). Therefore these operations are useful in practice. Some of the main applications of morphology are described below.

Boundary extraction: As dilation is an extensive operation and erosion is an anti-extensive operation, when applied to an image with a suitable SE, they can both be used to detect boundaries of a binary image. Boundary extraction of a binary image can be done simply by eroding the image by a suitable symmetric SE and then subtracting the result from the original image, (Gonzalez and Woods (2008)), giving the boundary of the set A by $\beta(A)$ as:

$$\beta(A) = A - (A \ominus B).$$

According to Dougherty and Lotufo (2003), $\beta(A) = A-(A\ominus B)$ provides only the *internal boundary* of an image. We may also be interested in the *external boundaries*, which in terms of the morphological operations are $\beta(A) = (A \oplus B) - A$, and $\beta(A) = (A \oplus B) - (A \ominus B)$ which provide a boundary that straddles the ac-

tual Euclidean boundary. This is also known as the *morphological gradient* (see Section 2.6). Among many others, a recent application of boundary extraction is in Mai and Wang (2008) who developed an effective image-processing method to automatically extract the boundary of a shoe pattern. They used a histogram thresholding technique to segment out a shoe pattern from the scanned input image and then applied boundary extraction on the segmented image to automatically detect and smooth the shoe-pattern boundary. Figure 2.6 shows the internal and external boundaries as well as the morphological gradient of an 256^2 binary image of squares.



(a) Original image (b) Eroded image (c) Dilated image

(d) Internal boundary (e) External boundary (f) Morphological Gradient

Figure 2.6: Effect of boundary extraction on a 256^2 binary image of squares, using a square SE of width 2, where width=0.5∗(base length-1).

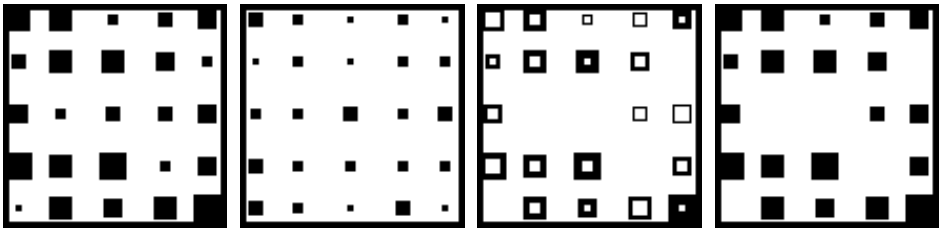
Hole filling: Holes may be the result of various causes: manual manipulation, e.g. removal of an object from an image, errors in the transmission of an image or video, etc. Sometimes it may be important to fill holes or missing regions in images. The hole is filled one pixel at a time by comparing the neighbourhood of each pixel to other areas in the image. Hole filling in a set A can be done by using the dilation, intersection and complement of the image.

Gonzalez and Woods (2008) define a hole as a background region surrounded by a connected border of foreground pixels, and provide an algorithm which starts with a point p in each hole and ends when all holes are filled. It begins by

forming an array X_0 of 0s (the same size as the array containing A), except that the locations in X_0 corresponding to the given point in each hole have been set to 1, then it dilates X_0 by a symmetric and convex SE B, and after each dilation takes the intersection with A^c. The dilation tends to fill the whole area, whereas the intersection with A^c limits the result to inside the region A. Mathematically, the following procedure fills all the holes with 1s:

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

using a symmetric SE B, if A is a set containing one or more holes, and $X_0 = \{p\}$. The algorithm terminates at iteration step k if $X_k = X_{k-1}$. The set X_k then contains all the filled holes, and the set union of X_k and A contains all filled holes and their boundaries, i.e. the filled object. The original image in Figure 2.7 contains images of size 101^2. The first one contains some larger squares, the second has relatively smaller squares and the third is obtained by subtracting the second one from the first one. Lastly the holes have been filled, by using the Matlab function 'imfill' with 4-connectivity (see Section 1.4).



(a) Image with larger squares (b) Image with smaller squares (c) $(a) - (b)$ (d) Hole filling

Figure 2.7: Hole filling using morphological operators on a 101^2 binary image of squares.

Conditional dilation: As dilation expands the image, by repeated dilation the input image can be grown unboundedly. Sometimes it is important to restrict the growth. This can be achieved by conditional dilation. Dougherty and Lotufo (2003) define conditional dilation of the image A which is a sub-image of C by the SE B as:

$$A \oplus_C B = \bigcup_{a \in A} B_a \cap C,$$

i.e. it is the union of all points in B, translated by all points in A, which overlap C.

Extraction of connected components: Binary images can be expressed as the union of connected regions. If each of these regions is maximally connected, i.e. is not a proper subset of a larger connected region within the image, then the regions are called *connected components* of the image (Dougherty and Lotufo (2003)). The extraction of connected components from a binary image plays a vital role in many automated image analysis applications. The procedure described by Gonzalez and Woods (2008) for extracting connected components is as follows: Again, the algorithm begins by forming an array X_0 of 0s (of the same size as the array containing A), except the locations in X_0 corresponding to a given point p in each hole have been set to 1. Then, the following iterative procedure finds all the connected components:

$$X_k = (X_{k-1} \oplus B) \cap A,$$

if B is symmetric, A is a set containing one or more connected components and $X_0 = \{p\}$. The algorithm terminates at iteration step k if $X_k = X_{k-1}$. The set X_k then contains the entire connected component.

Extraction of connected components is a widely used morphological technique. For example, Auran and Malvig (1996) presented an algorithm for the segmentation of echo clusters within a dynamic 3-D sonar image using the concept of cell connectivity between sonar beams.

Convex hull: A convex set is a set of elements from a vector space such that all the points on the straight line between any two points of the set are also contained in the set. Gonzalez and Woods (2008) describe the following sequence of morphological operations for obtaining the convex hull of a set A in an image:

Let B^i, $i = 1, 2, \ldots, n$, be a small set of SEs. The procedure starts with an initial image $X_0^i = A$ and consists of implementing the operation:

$$X_k^i = (X_{k-1} \circledast B^i) \cup A, \quad i = 1, 2, 3, \ldots \text{and} \quad i = 1, 2, 3, \ldots.$$

When the procedure converges, i.e. when $X_k^i = X_{k-1}^i$, then let $D^i = X_K^i$. Then the convex hull of A is

$$c(A) = \bigcup_{i=1}^{n} D^i,$$

i.e. the method iteratively applies the hit-or-miss-transform to A using B^1 until there is no more change. The procedure is repeated with B^2 until no further changes occur, and so on and the convex hull is the union of the results.

Thinning: One of the most common applications of morphological operations is *thinning* objects. This mainly depends on the hit-or-miss-transform (Dougherty and Lotufo (2003)). The thinning of a set A by a SE B is denoted by $A \otimes B$, and is defined as the difference between the original image and its hit-or-miss-transform, i.e.

$$A \otimes B = A - (A \circledast B) = A \cap (A \circledast B)^c.$$

That is, the thinning operation removes the part of A which has been detected by the hit-or-miss-transform from B. For a sequence of SEs, $B = B_1, B_2, \ldots B_n$, thinning can be carried out recursively (Gonzalez and Woods (2008)) as

$$A \otimes B = ((\cdots ((A \otimes B_1) \otimes B_2) \cdots) \otimes B_n).$$

So first A is thinned by B_1, then the resulting set is thinned by B_2, and so on until A is thinned with one pass of B_n. Thinning may be used to identify pixels that may be removed without affecting connectivity.

Thickening: *Thickening* is just the morphological dual of thinning, denoted by $A \odot B$ and defined as

$$A \odot B = A \cup (A \circledast B).$$

That is, the thickening operation adds the part of A which has been detected by the hit-or-miss-transform from B (Glasbey and Horgan (1994)). However thickening is often done by thinning the background of the set A and then taking the complement of the result. For a sequence of SEs $B = B_1, B_2, \ldots, B_n$, thickening can be carried out sequentially (Gonzalez and Woods (2008)) as

$$A \odot B = ((\cdots ((A \odot B_1) \odot B_2) \cdots) \odot B_n).$$

Skeletonisation: Skeletonisation is the process of reducing the foreground region of an image by removing as many pixels as possible without affecting the general shape of the image, i.e. after pixels have been removed, the remaining image should largely preserve the extent and connectivity of the original image. According to Glasbey and Horgan (1994) the skeleton should fulfil some basic requirements, such as (a) preserve the topology of the object, (b) be only one pixel thick and (c) be in the middle of the object. Figure 2.8 shows an arbitrary shaped binary image of size 101×101, its boundary and skeleton, obtained using the Matlab function 'bwmorph'.

(a) Original image (b) Object boundaries (c) Skeletonisation

Figure 2.8: Effect of skeletonisation on a 101^2 binary image.

Pruning: Sometimes a skeletonised image contains some barbs, i.e. some small features sticking out from the main skeleton. Very often it is desirable to remove those small barbs if they are not considered to be part of the true structure of an object. They can be removed by an operation called debarbing or *pruning*. Pruning can be regarded as a form of thinning, since it involves removing pixels on the basis of some criterion. It is useful if the skeleton is affected by some small features that are not of interest or are due to noise (Glasbey and Horgan (1994)).

2.5 Morphological Operations for Grey Scale Images

A grey scale image f_{ij} can be regarded as a $3-D$ stack of binary sets, because an image defined by a function in two-dimensions produces a map in the 3^{rd} dimension (Glasbey and Horgan (1994)). The pixel intensities provide the height in the third dimension. According to Dougherty and Lotufo (2003) the grey scale image is a real-valued function defined in Euclidean space or on the Cartesian grid. For mathematical morphology in three-dimensions, we need to define a SE b_{ij} of the same format. So both f_{ij} and b_{ij} are $2-D$ functions defined on $3-D$ sets, assigning intensity values for each distinct coordinate pair (i, j), and they are assumed to be discrete (Gonzalez and Woods (2008)).

First it is useful to notice how translation and reflection work in the case of grey scale images. The translation of a set B by a point $z = (z_1, z_2)$, denoted by B_z, is defined as

$$B_z = \{w : w = b + z, \text{ for } b \in B\},$$

i.e. coordinates (x, y) of B have been translated as $(x + z_1, y + z_2)$ to form B_z.

The reflection of the set B is denoted by $\hat{B}$ and is defined as

$$\hat{B} = \{w : w = -b, \text{for}\, b \in B\}.$$

Reflection is the rotation of B by 180^o about its reference pixel, i.e. if B has co-ordinates (x, y), $\hat{b}$ will have co-ordinates $(-x, -y)$.

2.5.1 Grey scale dilation

The dilation of image f by a flat 2-D SE b at any location (i, j) is defined as the maximum value of the image in the region outlined by the reflection of the SE $(\hat{b})$, when the origin of $\hat{b}$ is at (i, j) (Gonzalez and Woods (2008)). The dilation of f by b is denoted as

$$(f \oplus b)_{ij} = \max_{(k,l)\in b}\{f(i-k, j-l)\},$$

where $(k, l) = (0, 0)$ is the reference pixel of b. The reference pixel of the reflected SE visits every pixel in f, and the dilation is computed by taking the maximum values of the intensity of f in every neighbourhood of (x, y) coincident with b (Glasbey and Horgan (1994)). Dougherty and Lotufo (2003) define the dilation of a grey scale image f by a SE b as

$$(f \oplus b)_{i,j} = \min\{y : -\hat{b} + y \geq f\},$$

i.e. consider the negative of the reflected SE and find the minimum by which it can be raised and still remain beneath signal f. The effect of grey scale dilation is shown in Figure 2.9.

Gonzalez and Woods (2008) defines dilation of f by a non-flat SE b_N as

$$(f \ominus b_N)_{i,j} = \max_{(s,t)\in b_N}\{f(i-s, j-t) + b_N(s, t)\},$$

which adds every value of b_N from the input image. A non-flat SE is rarely used in practice because of its computational burden.

2.5.2 Grey scale erosion

The erosion of a grey scale image f by a flat SE b at any specific location is defined as the minimum value of the image in the region coincident with the SE when its origin is also set at that location (Gonzalez and Woods (2008)). Notationally,

the erosion of f by b is given by

$$(f \ominus b)_{ij} = \min_{(k,l)\in b}\{f(i+k, j+l)\}.$$

Thus, the erosion of a grey scale image is the process of transforming f by taking the minimum value of f in the neighbourhood about each pixel of f corresponding to the SE placed with its reference pixel at pixel (i, j) (Glasbey and Horgan (1994)). Erosion can be defined in terms of the dual of dilation as

$$(f \ominus b) = (f^c \oplus \hat{b})^c,$$

i.e. eroding f by b is the same as the complement of dilating f^c by the reflection of b. Dougherty and Lotufo (2003) define the erosion of a grey scale image f by a SE b as

$$(f \ominus b)_{i,j} = \max\{y : \hat{b} + y \geq f\},$$

i.e. it finds the maximum difference between f and the translated SE over the domain of $\hat{b}$.

Gonzalez and Woods (2008) defines erosion of f by a non-flat SE b_N as

$$(f \ominus b_N)_{i,j} = \min_{(s,t)\in b_N}\{f(i+s, j+t) - b_N(s,t)\},$$

which subtracts every value of b_N from the input image. The general effects of grey scale dilation and erosion are clearly described in Gonzalez and Woods (2008). Since grey scale dilation by a flat SE b computes the maximum intensity value of f in every neighbourhood of (i, j) coincident with b, the bright features are thickened and the intensities of the dark features are reduced. Small dark spots in images disappear as they are 'filled in' by the surrounding intensity value. Small bright spots become larger spots. Since grey scale erosion computes the minimum intensity value of f in every neighbourhood of (i, j) coincident with b, the sizes of bright features are reduced and the sizes of dark features are darkened and widened. Small bright spots disappear as they are eroded away down to the surrounding intensity value, and small dark spots become larger spots. So in general the eroded grey scale image is darker than the original image, and the dilated grey scale image is brighter than the original. For both dilation and erosion, the effect is most marked at places in the image where the intensity changes rapidly and regions of fairly uniform intensity are largely unchanged except at their edges.

Figure 2.9 shows a grey scale 256^2 image of ellipses and the effect of dilating

and eroding with a flat disk of radius 10. In general the dilated grey scale image is brighter than the original image. We observe that the bright features are thicker and darker features are reduced in the dilated image. The eroded grey scale image is darker than the original as the darker parts have expanded and the brighter parts have shrunk.

(a) Original image (b) Dilated image (c) Eroded image

Figure 2.9: Grey scale dilation and erosion of a 256^2 grey scale image of ellipses of random radii, shapes and sizes, by a flat disk of radius 10.

2.5.3 Grey scale opening and closing

As for binary images, grey scale opening and closing are also based on morphological dilation and erosion. The opening of an image is the collection of foreground parts of an image that fit a particular SE b. The grey scale opening is analogous to its definition in the binary setting (Gonzalez and Woods (2008)), i.e. erosion of f by b and then dilation of the resulting image by the same SE, namely,

$$f \circ b = (f \ominus b) \oplus b. \tag{2.17}$$

Dougherty and Lotufo (2003) define opening in terms of fitting, i.e. opening is the maximum over all morphological translations of the SE that fit underneath the input image,

$$f \circ b = \bigvee \{b_z + y : b_z + y \leq f\}, \tag{2.18}$$

where b_z is the translation of b by z. A horizontal translation of f by x is denoted as $f_x(i) = f(i - x)$ and a vertical translation of f by y is defined by $(f + y)(z) = f(z) + y$. Morphological translation is the combination of horizontal and vertical translation and is expressed as $(f_x + y)(z) = f(z - x) + y$.

A practical description of (2.18) is given by Dougherty and Astola (1994), in terms of fitting, as sliding the SE along beneath the input image and at each point recording the point on the SE's translation that is highest at that point.

Similarly, the closing of an image is the collection of background parts of an image that fit a particular SE, defined as dilation followed by erosion,

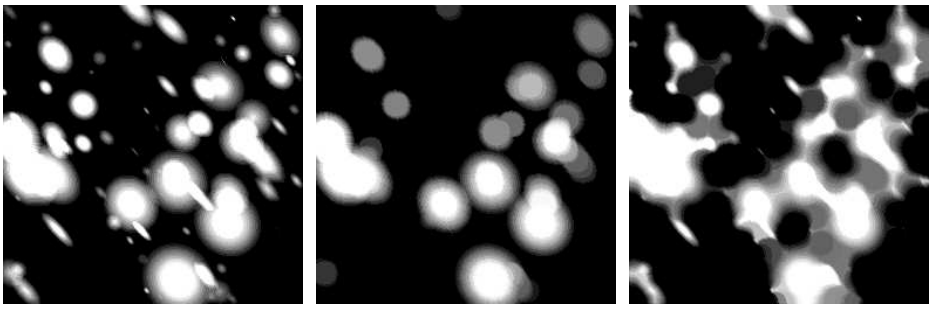
$$f \bullet b = (f \oplus b) \ominus b. \tag{2.19}$$

Grey scale closing can be defined in terms of its duality with opening as

$$f \bullet b = (f^c \circ \hat{b})^c, \tag{2.20}$$

i.e. the closing is the same as the complement of the opening of the image background (f^c) by the reflection of the same SE. Closing filters an image from above, whereas opening filters from below. In terms of fitting, closing can be interpreted as sliding the SE down from above the image and at each point recording the point on the SE's reflection that is highest at that point. The opening of a function f by a convex set b cuts down the peaks of f, whereas the closing of f by b fills up the valleys of f.

Figure 2.10 illustrates grey scale opening and closing of an image of ellipses, using a disk SE of radius 10. In the opened image all ellipses with semi-minor axis smaller than 10 have disappeared and the brightness of the larger ones is reduced, whereas in the closed image the brightness of all ellipses has increased so the opened image is generally darker than the closed image.



(a) Original image (b) Opened image (c) Closed image

Figure 2.10: Grey scale opening and closing of a 256^2 grey scale image of ellipses of random radii, shapes and sizes, by a flat disk of radius 10.

Properties of the basic grey scale morphological filters: Grey scale dilation, erosion, opening and closing satisfy the following properties:

1. Both grey scale dilation and erosion are translation invariant:

$$(f_z \oplus b) = (f \oplus b)_z \quad \text{and} \quad (f_z \ominus b) = (f \ominus b)_z.$$

2. Dilation is commutative:

$$(f \oplus b) = (b \oplus f).$$

3. Dilation and erosion are duals:

$$(f \oplus b) = (f^c \ominus \hat{b})^c \quad \text{and} \quad (f \ominus b) = (f^c \oplus \hat{b})^c.$$

4. Both grey scale opening and closing are translation invariant:

$$(f_z \circ b) = (f \circ b)_z \quad \text{and} \quad (f_z \bullet b) = (f \bullet b)_z.$$

5. Opening lies beneath the original image, i.e. it is anti-extensive, and closing lies over the original image, i.e. it is extensive:

$$f \circ b \leq f \quad \text{and} \quad f \bullet b \geq f,$$

so the following relation is always true:

$$f \circ b \leq f \leq f \bullet b.$$

6. If $f_1 \leq f_2$, then the opening of image f_1 by a SE b is a subset of the opening of image f_2 by the same SE, i.e.

$$f_1 \circ b \leq f_2 \circ b.$$

7. If $f_1 \leq f_2$, then the closing of image f_2 by a SE b is a subset of the closing of image f_1 by the same SE, i.e.

$$f_1 \bullet b \leq f_2 \bullet b.$$

8. Opening and closing are both *idempotent*, and have a one-off effect i.e.

$$(f \circ b) \circ b = f \circ b \quad \text{and} \quad (f \bullet b) \bullet b = f \bullet b.$$

2.6 Some Applications of Grey Scale Morphology

Morphological gradient

The gradient of a scalar field is a vector field which indicates the greatest rate of increase of the scalar field, and whose magnitude is the greatest rate of change. The gradient of the function $f(x)$ of a finite dimensional vector $(x = x_1, x_2, \cdots, x_n) \in \mathbb{R}^n$ is defined by

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right).$$

In image processing the gradient plays a vital role in edge detection. A higher gradient suggests a more rapid light-to-dark (or dark-to-light) change in the grey scale case (see Dougherty and Lotufo (2003)). Although a number of gradients exist, the most commonly used gradient in image processing is the *morphological gradient*.

Since dilation thickens regions in an image and erosion shrinks them, by subtracting them we obtain a non-negative quantity, known as the morphological gradient. Let f be the original image and b_1, and b_2 be two SEs. The morphological gradient $grad(f)$ is defined by

$$grad_{b_1,b_2}(f) = (f \oplus b_1) - (f \ominus b_2). \tag{2.21}$$

as for the binary image. Equation (2.21) indicates that it is essentially the sum of two partial gradients, namely, the *external gradient* and the *internal gradient*, referred to in the binary case as the external and internal boundary of an image (see Section 2.4). The external gradient is defined by

$$grad_{b_1}^{ext} = (f \oplus b_1) - f,$$

and the internal gradient is

$$grad_{b_2}^{int} = f - (f \ominus b_2).$$

The SEs b_1 and b_2 may be the same.

Top-hat transforms

Hat transformations play a pivotal role in image pre-processing. The opening of a grey scale image lies beneath the original image. Subtracting the opened image

from the original image is known as the *open top-hat transform*, or simply the *top-hat transform* in morphology, i.e.

$$f\hat{\circ}g = f - (f \circ g).$$

Since opening is anti-extensive, opening lies beneath f and $f\hat{\circ}g$ is always non-negative.

The complementary operator to the open top-hat transform is the *close top-hat transform*, known as the *bottom-hat transform*, defined as

$$f\hat{\bullet}g = (f \bullet g) - f.$$

Since closing is extensive, the grey scale closing lies over the original image, so $f\hat{\bullet}g$ is always non-negative. Figure 2.11 contains a 200×500 grey scale image of a fish (available at http://www.peipa.essex.ac.uk) and shows the effect of the top-hat transform and the bottom-hat transform.

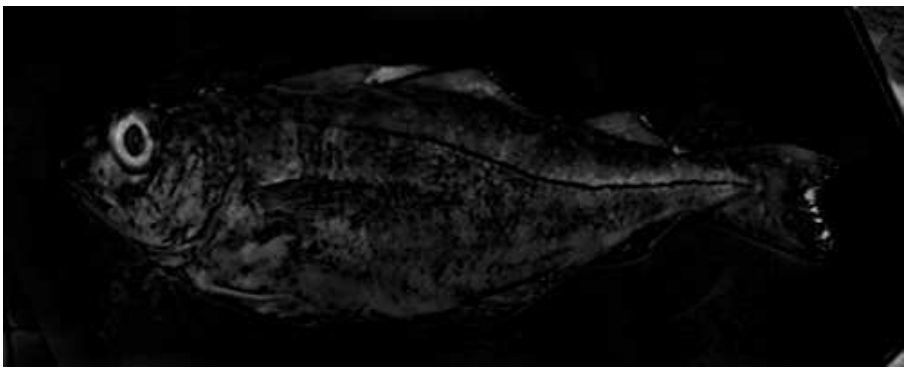
The top-hat transformation is used to reduce uneven illumination. It highlights the bright features of the image, as opening eliminates the bright features from the image and they appear again when the opened image is subtracted from the original image. Similarly, bottom-hat transformation highlights the dark features of the image. A non-flat ellipsoid SE with radius 10 and height 2 was used where the radius determines the shape of the ellipsoid and height specifies its colour. It is clear from Figure 2.11 that top-hat transformation eliminates the uneven distribution of the dark and bright features of the images as well as indicating brighter regions of the fish, and bottom-hat transformation indicates the darker parts of the fish. We make use of these transformations in Chapters 6 and 7, for improved texture classification.

Morphological reconstruction:

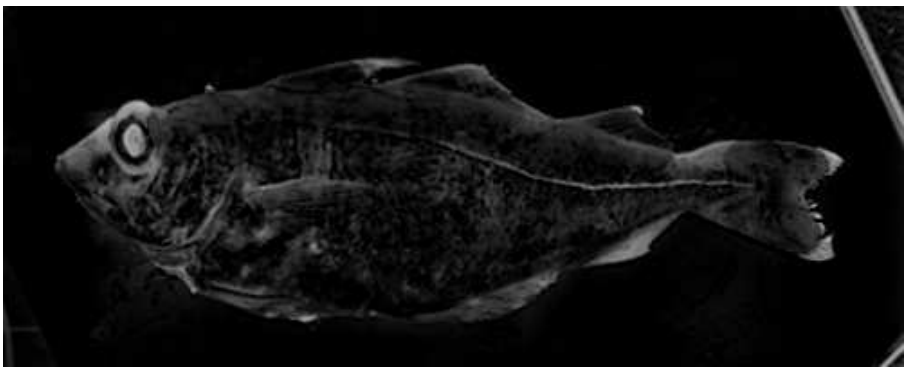
Morphological reconstruction is a powerful morphological operation using the concept of connectivity (see Section1.4) in images, both for binary and grey scale, instead of a SE. It can be defined either by the thresholding *superposition principle* or can rely on *geodesic dilation* and *geodesic erosion* (Gonzalez and Woods (2008)). Morphological reconstruction has many applications including converting a complex image background to uniform intensities.

Binary reconstruction: To define morphological reconstruction by thresholding superposition, Vincent (1993) defined two binary images I and J on the same discrete domain D, such that $J \subseteq I$. I is called the *mask* image and J

(a) A fish image of size 200×500



(b) Top-hat transform of (a)



(c) Bottom-hat transform of (a)

Figure 2.11: Effect of top-hat and bottom-hat transformation of an image of a fish, using a ellipsoid of radius 10 and height 2 (see Matlab function 'strel' with option 'ball').

is the *marker*. The reconstruction $\rho_I(J)$ of I from marker J is the union of the connected components of I which contain at least one pixel of J, i.e.

$$\rho_I(J) = \bigcup_{J \cap I_k \neq \emptyset} I_k. \tag{2.22}$$

Very often reconstruction is defined in terms of *geodesic distance*. The geodesic distance between two pixels p and q in a given set is the length of the shortest

path joining p and q which is included in the given set. Gonzalez and Woods (2008) define geodesic dilation of size $n \geq 0$ of the marker image J with respect to the mask I, denoted by $\rho_I^{(n)}(J)$, as

$$\rho_I^{(n)}(J) = \rho_I^{(1)}[\rho_I^{(n-1)}(J)], \tag{2.23}$$

with $\rho_I^{(0)}(J) = J$, and $\rho_I^{(1)}(J)$ is the geodesic dilation of size 1 of the marker J with respect to the mask I, defined as

$$\rho_I^{(1)}(J) = (J \oplus B) \cap G, \tag{2.24}$$

where B is a suitable SE. That is, the reconstruction of I from J is obtained by iterating elementary geodesic dilations of J inside I until there is no change. This can be defined in terms of *geodesic erosion* as well.

Grey scale reconstruction: In terms of thresholding superposition, Vincent (1993) defined two grey scale images I and J in the same domain D, taking values in the discrete set $\{0, 1, \ldots, N-1\}$ such that $J \leq I$ (i.e., for each pixel $p \in D_I, J(p) \leq I(p)$). The grey scale reconstruction $\rho_I(J)$ of I from marker J is given by

$$\rho_I(J) = \max\{k \in [0, N-1] \mid p \in \rho T_{k(I)}(T_k(J))\} \quad \forall p \in D, \tag{2.25}$$

where the $T_k(I)$ are k successive thresholds of I such that

$$T_k(I) \subseteq T_{k-1}(I) \quad \forall k \in [1, N-1].$$

In terms of geodesic dilation, grey scale reconstruction can be obtained by iterating grey scale dilations of J under I until there is no change, i.e.,

$$\rho_I(J) = \bigvee_{n \geq 1} \partial_I^{(n)(J)}. \tag{2.26}$$

τ-opening: τ-opening is a union of parametrised openings in which parameters for each opening are individually defined and a SE can be parametrised relative to both size and shape (Chen and Dougherty (1991)). If the aim is to pass portions of an image conforming to any one of a number of shape primitives, a single opening will not be appropriate as it will pass only parts of the image which conform to the shape of the SE. Such an effect can be accomplished by using a filter composed of a number of different openings. Dougherty and Lotufo (2003)

define τ-opening as a morphological filter such that

$$\Psi(A) = \bigcup_{k=1}^{n} A \circ B_k, \tag{2.27}$$

for some class $B = \{B_1, B_2, \ldots, B_n\}$ of SEs.

2.7 Conclusion

This chapter provides a condensed description of some widely used morphological techniques for digital image processing. We illustrate most techniques with example images. Grey scale opening is used extensively in most of the subsequent chapters in the thesis, and top- and bottom-hat transformations are used in Chapters 6 and 7 as a means of image pre-processing to eliminate the uneven intensity variation in the real images. The next chapter provides a discussion of texture feature extraction and classification techniques, before focusing on morphological granulometry in Chapter 4.

Chapter 3

Overview of Texture Analysis and Classification

In this chapter, we define texture and discuss different approaches to analysis of texture images as well as classification rules.

3.1 Introduction

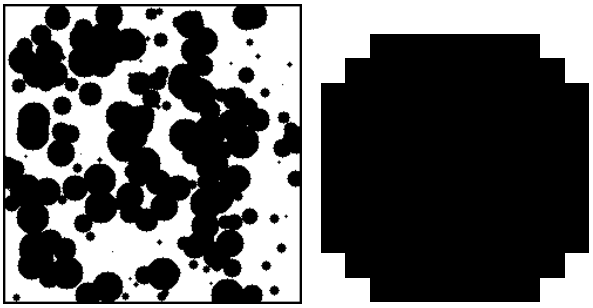
Owing to the diversity of natural and artificial textures (Jain and Farrokhnia (1990)) there is no generally agreed definition of texture and no unique mathematical model to synthesise texture (Nixon and Aguado (2002)). Some of the definitions are perceptually motivated and others are driven completely by the application (Tuceryan and Jain (1998)). Texture is an important cue in object recognition as it can tell something about the material from which the object is made (Petrou and García-Sevilla (2006)). Gonzalez and Woods (2008) defined texture as a surface property which gives combined information on the smoothness, coarseness, and regularity of objects.

Texture contains important information about the structural arrangement of surfaces and their relationship to the surrounding environment. Texture is something which we feel when we interact with our surroundings, either by touching or observing an object or an image, since the sense of touch cooperates with the eyes to give a better understanding of the surroundings. Texture is characterised not only by the grey value at a given pixel, but also by the grey value 'pattern' in a neighbourhood surrounding the pixel.

The statistical approach defines texture by a quantitative measure of the arrangement of intensities in a region. Dougherty et al. (1992) define it by perceptual descriptors, such as 'smooth', 'coarse', or 'regular', or as a pattern comprised

of repeated *texture primitives* or texture elements (*texels*). Image primitives are the basic elements of the image, i.e. the basic objects of which an image is composed. A more formal definition of texture, given by Livens et al. (1997) in terms of neighbourhoods, is as the set of local neighbourhood properties of the grey levels of an image region as well as their spatial relationships.

Tuceryan and Jain (1998) have accumulated an archive of definitions of texture, one of which is mentioned here. An image texture is non-figurative and cellular. It is described by the number and types of its primitives repetitions and the spatial organisation or layout of its primitives. The repetitiveness of the texels (the texture unit) determines the type of texture and decides the texture analysis approach (Jain (1989)). A fundamental characteristic of texture is that it cannot be analysed without a frame of reference for the primitive being stated or implied. For any smooth grey-tone surface, there exists a scale such that when the surface is examined, it has no texture. Then as resolution increases, it takes on a fine texture and then a coarse texture. Figure 3.1 shows two binary images, of which (b) is the (digitised version of the) primitives of image (a).

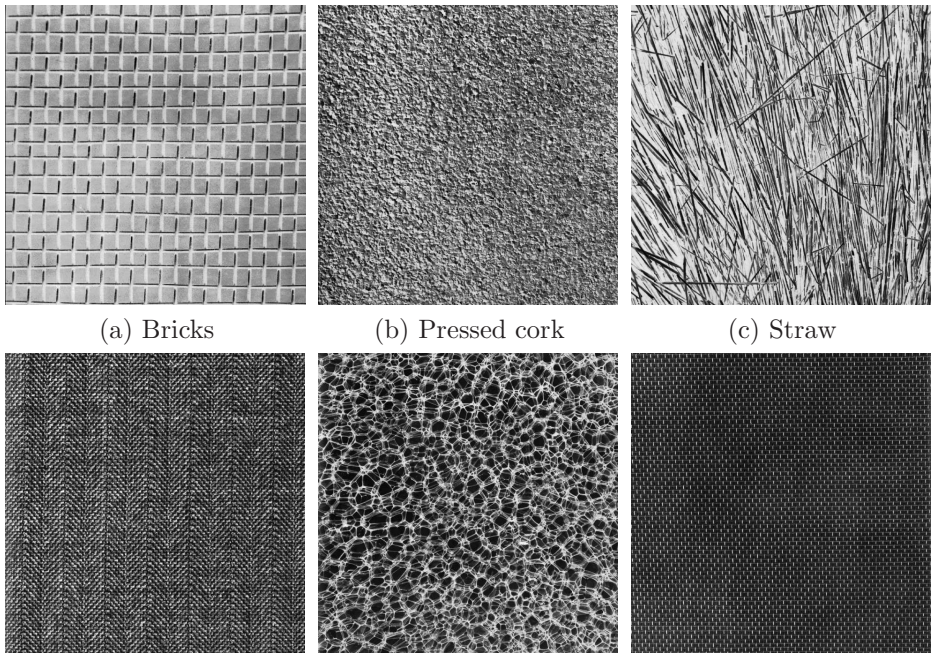


(a) Image of disks (b) Primitive of image (a)

Figure 3.1: An image and its primitive.

In computer graphics there are two major types of textures, i.e. *deterministic* or regular and *statistical* or irregular textures (Tuceryan (1994)). Deterministic texture is created by repetition of a fixed geometric shape such as a circle or square. Examples of deterministic textures are patterned wallpaper and bricks. Texels are represented by a placement rule which gives an arrangement of the primitives which yields the texture. Statistical textures are created by changing patterns with fixed statistical properties. In these textures the placement of the texture primitives is random and irregular, and the placement rule description for such textures may be extremely complicated. Most natural textures like wood or stone, grass, canvas are examples of statistical textures. Statistical textures are typically represented in terms of spatial frequency properties. In Figure 3.2, (a),

(d) and (f) represent some deterministic textures whereas (b), (c) and (e) show statistical texture images.



(a) Bricks (b) Pressed cork (c) Straw

(d) Herringbone weave (e) Bubbles (f) Unknown

Figure 3.2: Some example textures from the Brodatz album (Brodatz (1966)).

In this work we are particularly interested in texture images which result from a material subject to some sort of damage or decay, often as a result of organic processes, e.g. textured images of corrosion of metal.

Texture analysis is primarily concerned with three major issues, i.e. *texture classification*, *texture segmentation*, and *texture synthesis*. Texture classification involves identifying a given textured region with a class in a given set of texture classes, whereas texture segmentation is concerned with automatically determining the boundaries between various texture regions in an image, i.e. segmenting an image into regions according to the textures of the regions. More specifically, a texture classification system involves two steps (Masotti and Campanini (2008)): a feature extraction step, in which a set of texture features is extracted from the image under study, and a classification step, in which a texture class membership is assigned to it according to the extracted texture features.

Texture synthesis is the process of algorithmically constructing a large digital image from a small digital sample image by using its structural content (Paget and Longstaff (1998)). It is a way to create textures, by inferring the parameters of a real image texture and using them to create new texture images (Petrou and García-Sevilla (2006)).

A great deal of work has been done on texture analysis over the past few decades. The existing methods of texture analysis can be characterised as *statistical approaches*, *geometrical approaches*, *model-based approaches* and *transform-based methods*. Model-based approaches include *Markov random fields* and *auto-regressive models* (Harrison et al. (2008)). Statistical methods include features derived from the histogram, gradient, run-length matrix, and co-occurrence matrices. Structural or geometrical approaches mainly consist of mathematical morphology. Transform-based methods describe the textural properties of the objects by features derived from transformations used in image analysis, e.g. the Fourier transform, Gabor transform, and wavelet transform. Many authors (e.g. Jafari-Khouzani and Soltanian-Zadeh (2005), Xiao and Wu (2007), Wang and Yong (2008)) refer to this as multi-resolution texture analysis, as the transformations represent the image at different scales. They transform a texture image into a local frequency representation by convolving the original image with a bank of filters with some tuned parameters. However, very often the existing methods match the criteria of more than one approach.

This chapter gives a condensed description of the existing methods of texture analysis. Some of the most widely used texture analysis methods are described, with their relative performance.

3.2 Morphological Granulometry on Texture Analysis

Morphological granulometry, introduced by Matheron (1975), is extensively used in texture analysis and also in this thesis. A binary image is considered to be a collection of grains (Dougherty and Pelz (1991)), and granulometry sieves the grains through filters of increasing size. Grains with size smaller than the holes (filters) drop out and only the grains with larger sizes will remain. The shape of the holes is determined by the shape of the *structuring elements*. As a result, the underlying image that remains successively decreases in volume. A probability distribution function can be generated using the rate of decrease, and its moments, known as *granulometric moments*, are used to characterise the image. The granulometric moments have been extensively used to extract textural information from images. A detailed description of morphological granulometry is given in Section 4.1. Here we mention some applications of morphological granulometry for texture analysis.

Recently, Mavilio et al. (2010) used granulometric moments to characterise the

evolution of a dynamic process concerning paint drying. The pattern spectrum of the granular distribution of the grey scale temporal history of the speckle pattern (THSP) images was obtained. The first four granulometric moments (mean, standard deviation, skewness and kurtosis) were obtained from each THSP image and were used as texture features. The Mahalanobis distance (Mahalanobis (1936)) among the granulometric features was used to measure the textural difference between THSP images of drying paint.

Kyriacou et al. (2009) used pattern spectrum texture features to classify ultrasound images of atherosclerotic carotid plaques either as asymptomatic or symptomatic. Two different classifiers, i.e. a probabilistic neural network and support vector machines were used and the highest classification accuracy of 73.7% was obtained for support vector machines. Granulometric moments were used by Theera-Umpon and Dhompongsa (2007) to analyse white blood cell images using features extracted from the cell nuclei. They applied both the Bayes classifier and a neural network classifier and found that better classification was achieved by using the neural network classifier.

Díaz et al. (2007) carried out granulometric analysis on corneal endothelium specular images by means of the germ-grain model. If the granulometry is applied to the complement of a set they referred to it as *anti-granulometry*. They applied anti-granulometry to images composed of inscribed circles for controls and also pathological endothelia, and showed that the anti-granulometric size distribution of the inscribed circles discriminated well between the controls and the pathological endothelia.

McKenzie et al. (2003) used granulometric moments as texture features and developed parallel evolution functions (PEFs) using multiple regression modelling. The methodology was developed on computer-generated images where evolution of the artificial images depended explicitly on some evolution parameters which were set up as a known function of time before generating the images. The synthetic images were then used to relate granulometric moments to evolution parameters. Back-prediction was used to determine the evolution time of a new image based on the artificial image model and the observed granulometric moments from the new image. For classifying a sequence of corrosion images the PEF approach was found to be especially useful for small test set sizes. The work in this thesis builds directly on McKenzie et al. (2003).

Granulometry was successfully used in estimating the shape of a random pattern by Batman and Dougherty (2001), where the random pattern was determined by a multivariate probability distribution, and they used granulometric

features to estimate the parameters of the multivariate distribution. Granulometric size distributions and their moments are widely used as size-shape descriptors, and have proved very useful in medical imaging, materials sciences, and character recognition (Goutsias and Batman (2001)). Dougherty et al. (1993) formulated linear granulometric moments of additive and subtractive binary images. Dougherty et al. (1992) applied local binary granulometries to 10 Brodatz textures and achieved overall 99.8% accuracy (measured by number of correctly classified pixels as a proportion of the total number of pixels to be classified). They also used classification accuracy to determine a minimum window size required for effective classification. They examined six different window sizes, and found that the highest accuracy (98%) was obtained with a 20-pixel window size. Ayala and Domingo (2001) used the spatial size distributions as shape detectors. Dougherty and Pelz (1991) used size distribution statistics for process control to analyse electro-photographic images. Size distributions were successfully used by Maragos (1987) to study shape-size complexity, multi-scale shape representation, and symbolic image modelling.

3.3 Transformation-based Methods

Transformation-based methods mainly comprise the Fourier transform, Gabor transform and wavelet transform, the second two of which are multi-resolution methods. Some variations on Gabor filters and wavelets transforms are also mentioned.

3.3.1 Fourier transform

There are many ways of transforming image data into alternative forms that are more amenable for texture analysis. The Fourier transform (FT) is the most common image transform that takes an image in the spatial domain and transforms it into the frequency domain. This is an analysis of the global frequency content in the image, but many applications require the analysis to be localised in the spatial domain. This is handled by introducing spatial dependency into the Fourier transform (Tuceryan and Jain (1998)), which allows extraction of localised texture information.

The short-time FT is a Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of an image as it changes over time. The FT (a one-dimensional function) of the resulting signal is taken as a

window is slid along the time axis, producing a two-dimensional representation of the signal (Gonzalez and Woods (2008)). Mathematically, this is written as:

$$F_w(u,\xi) = \int_{-\infty}^{\infty} f(t)w(t-\xi)e^{-j\omega t}\,dt, \tag{3.1}$$

where $w(t)$ is a window function, usually a *Hann window* or *Gaussian 'hill'* centred around zero, $f(t)$ is the signal to be transformed, and $j = \sqrt{-1}$.

The discrete Fourier transform (DFT) is a sampled Fourier transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image, i.e. the image in the spatial and Fourier domains are of the same size (Gonzalez and Woods (2008)).

For an image f_{xy} of size $M \times N$, the two-dimensional DFT is given by:

$$F_w(k,l) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} f_{ij}e^{\{-j2\pi(\frac{ik}{M}+\frac{jl}{N})\}}, \tag{3.2}$$

where f_{ij} is the image in the spatial domain, the exponential term is the basis function corresponding to each output point $F_w(k,l)$ in Fourier space, and the discrete variables k and l take values $0, 1, 2, \ldots, M$ and $0, 1, 2, \ldots, N$ respectively.

The FT plays a pivotal role in image processing applications, including enhancement, analysis, restoration, and compression as well as in texture analysis, and many texture analysis techniques (such as the Gabor filter and wavelets) are based on the FT. Some recent applications of FT in texture analysis are mentioned below:

Liao and Chung (2010) introduced the composite Fourier domain (CFD) which was constructed by taking the local FT of the original texture images and a global multi-dimensional FT was then applied to the local FT images to obtain the multi-dimensional frequency domain coefficients. A null-space based linear discriminant analysis (nLDA) was derived from the traditional LDA and was applied to the CFD to find the optimal discriminant sub-space in CFD. Higher classification accuracy was obtained using a support vector machine with a radial basis kernel for Brodatz texture images.

The FT was also found useful for segmentation of dynamic textures in Li et al. (2009). The authors computed the phase spectrum for each of the dynamic texture images using a 3-D discrete FT and then applied 3-D inverse fast FT on the phase spectrum to obtain the reconstructed texture images, which were

then smoothed using an average filter. Finally, binarisation was applied to the smoothed reconstructed images to obtained segmented images.

A rotation-invariant texture classification technique was proposed in Xiao et al. (2007). They rotated the images of interest from 0° to 180° in 10° increments and used the Hough transform to convert the rotation image to a translation in the parameter space, by isolating features of a particular shape within the image. Rotation-invariant features were obtained by applying 2-D FT to the Hough transformed images. The features were able to characterise the texture images.

Xiao and Wu (2007) proposed a rotation-invariant texture classification technique based on the Radon and Fourier transforms. They first calculated the Radon transform of an image and then the FT was computed to extract the corresponding rotation-invariant features. The Radon transform of an image f_{xy} is defined as its integration along a particular line defined by a normal distance r from the origin and normal angle θ. This gives the corresponding Radon transform point $R(r, \theta)$ as

$$R(r, \theta) f_{xy} = \int_{\infty}^{\infty} \int_{\infty}^{\infty} f_{xy} \partial(r - x \cos\theta - y \sin\theta) dx dy, \tag{3.3}$$

where $-\infty < r < +\infty$ and $0 < \theta < \pi$. By using a $2D$ FT, rotation-invariant features are produced. Finally to classify the textures they used a k-nearest neighbour classifier. They tested their method on 15 Brodatz texture images, and obtained 100% classification accuracy for almost all of them in the case of no noise, and the method was robust to additive white noise.

A FT was used successfully to detect structural defects in fabric in Chan and Pang (2000), where the defects were of four types, i.e. double yarn, missing yarn, broken fabric and variation in yarn density. Jing et al. (2009) developed a face recognition algorithm based on an angular FT. The algorithm first determines an optimum angle to use to obtain the angular FT of the original images. *Fisher-face* techniques were applied to extract discriminative texture features from the transformed images. The method was applied to the well-known ORL and FERET face databases and satisfactory classification results were obtained for both datasets using a nearest neighbour classifier.

3.3.2 Gabor filters

Gabor (1946) first introduced the windowed Fourier transform, which represents a signal in a joint time-frequency domain (Debnath (2002)). These functions,

named as Gabor functions, have been used extensively in their 2-D form in texture analysis of digital images and are used for various kinds of image analysis applications. They have been used for texture segmentation, face detection and recognition, text detection and localisation in document images, and script identification in multi-script scenarios, for example. In the space domain, a Gabor filter corresponds to a sinusoidal wavelet within a Gaussian envelope, and local image texture is described by the Gabor filter that gives a maximum image-filter convolution (Pasternack et al. (2009)).

A Gabor function in 1-D is defined by Petrou and García-Sevilla (2006) as:

$$g(u; u_0, \omega_0, \sigma) \equiv \exp\left\{\frac{-(u-u_o)^2}{2\sigma^2}\right\} + j\omega_0 u, \tag{3.4}$$

and the Fourier transformation of equation (3.4) is given by

$$G(u; u_o, \omega_0, \sigma) = \sqrt{2\pi}\sigma \exp\left\{-j(\omega-\omega_0)u_0 + \frac{\sigma^2(\omega-\omega_0)}{2}\right\}, \tag{3.5}$$

where u is the spatial co-ordinate, ω_0 is the sinusoidal plane wave of some frequency, ω is the observed spatial frequency, and σ is the standard deviation of the Gaussian envelope along the x-axis. Gabor filters provide a multi-resolution approach to texture characterisation (Jafari-Khouzani and Soltanian-Zadeh (2005)).

The Gabor function was extended to two dimensions by Daugman (1980). A 2-D Gabor function is a harmonic oscillator, which is a sinusoidal plane wave of some frequency and orientation within a Gaussian envelope, whose frequency, orientation and bandwidth are controlled by its parameters (Fogel and Sagi (1989)). A *canonical* Gabor filter in the spatial domain is given as:

$$g(u, v) = \exp\left[-\frac{1}{2}\left(\frac{u^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2}\right)\right] cos\left(2\pi u_0 u + \psi\right), \tag{3.6}$$

where u_0 is the sinusoidal plane wave of some frequency, σ_u and σ_v are the standard deviations of the Gaussian envelope along the x-axis and y-axis respectively, and ψ is the phase of the sinusoidal plane wave along the x-axis (i.e. the 0^o orientation). When $\psi = 0$, the FT of the Gabor filter in (3.6) is given by

$$G(u, v) = 2\pi\sigma_u\sigma_v\left[\exp\left(-\frac{1}{2}\left(\frac{(u-u_0)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2}\right)\right) + \exp\left(-\frac{1}{2}\left(\frac{(u+u_0)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2}\right)\right)\right]. \tag{3.7}$$

Equations (3.4) and (3.6) can be thought of as a family of functions for different pairs of values of ω_0 and u_0, so they represent an image in the time and

frequency domains simultaneously and become suitable to represent any image f_{xy} in terms of this family of functions (Jain and Farrokhnia (1990)).

To analyse an image in terms of Gabor filters, first the FT of the image is taken, then the result is multiplied by a Gaussian window centred at various frequencies, and then the inverse FT of the results is taken. The central frequencies of the Gaussian windows are taken in such a way that all frequency bands of the image are covered (Petrou and García-Sevilla (2006)).

The Gabor transform was applied to target echo-signals and the trace of the Gabor transform coefficient matrix and the eigenvalues were used to describe the textural information of the target echo-signals in Yang et al. (2009). Pasternack et al. (2009) used analogue optical Gabor-like filters to analyse biological sample textures. They used Gabor filters to characterise non-spherical sub-cellular particles with the aid of a digital micro-mirror device. The morphometric features of sub-cellular organelles were characterised by their differential response to Gabor filters with different dimensions and orientations.

Mengko and Pramudito (2002) applied Gabor filters to classify osteoporosis level, based on change in trabecular pattern. Classification was based on the energy features extracted by Gabor filtering, and the classification result was compared to the *Singh Index* which was determined by a physician. The extracted features in the form of energy from 55 radiographs almost matched their Singh Index values. Haley and Manjunath (1995) proposed a modified Gabor filter using a Gabor wavelet transform to create a multi-resolution space frequency representation of a texture image. First they represented an input image as a polar form of the Gabor function, and the resulting expression was re-expressed as the basic wavelet function. The micro-features were defined in terms of the wavelet coefficients. The modified Gabor filter was applied on 13 texture images and overall 99% classification accuracy was achieved.

Jain and Farrokhnia (1990) used a multi-channel filtering approach and a fixed set of Gabor filters was used to characterise the channel that preserved almost all the information in the input image. An unsupervised texture segmentation algorithm using a fixed set of Gabor filters was proposed, and applied to several images to demonstrate its performance. Fogel and Sagi (1989) applied the Gabor filter model to separate micro-patterns of an image and found the pattern spectrum of each basic element by means of Gabor filters, then they calculated the dissimilarities between two pattern spectra.

There are many advantages of Gabor filters over other approaches. Some of these advantages are:

1. they simulate the human visual system,

2. they are direction dependent,

3. they have optimal joint localisation or resolution, in both the spatial and the spatial-frequency domains (Daugman (1985)), and

4. they meet the equality criteria in the *uncertainty principle*, where the uncertainty principle states that certain physical quantities, like position and momentum, cannot both have precise values at the same time. The narrower the probability distribution for one quantity, the wider it is for the other.

However, Gabor filters are criticised by Wang and Yong (2008), as there is a compromise between redundancy and completeness in the design of Gabor filters because of their non-orthogonality.

3.3.3 Wavelets

The shifting and scaling properties of the FT, in which an image is represented as a sum of sinusoids, led to the evolution of wavelets for texture analysis and image and signal analysis, as these provide simultaneous representation and localisation of both time and frequency for non-stationary signals (e.g. music, speech, images), whereas the standard FT is only localised in frequency. These decompose an image into a complete set of wavelet functions which form an *orthogonal* basis (Petrou and García-Sevilla (2006)). The concept of wavelets in its present form, proposed by Morlet (1984), is now well established and has found many applications in signal and image processing.

Wavelet transforms have advantages over traditional FTs for representing functions that have discontinuities and sharp peaks, and for accurately deconstructing and reconstructing finite, non-periodic and/or non-stationary signals. Compared to the wavelet transform, using the Gabor transform requires selection of the filter parameters for different textures (Wang and Yong (2008)).

A wavelet is a mathematical function used to divide a given function or continuous-time signal into different scale components. Each scale component can then be studied with a resolution that matches its scale. A wavelet transform is the representation of a function by wavelets. The wavelets are scaled and translated copies (known as 'daughter wavelets') of a finite-length or fast-decaying oscillating waveform known as the 'mother wavelet' (Mallat (1999)).

Unser (1995) defines the wavelet transform as a multi-resolution decomposition for finite energy functions f of a continuous variable x, i.e. $f(x) \in L_2$, where L_2 represents the space of square summable sequences. It provides a time frequency representation of an image, and the wavelet coefficients of an image are the projections of the image onto multi-resolution subspaces (Jafari-Khouzani and Soltanian-Zadeh (2005)).

Wavelet transforms are classified into discrete wavelet transforms (DWTs) and continuous wavelet transforms (CWTs). Both DWTs and CWTs are continuous-time (analogue) transforms, which can be used to represent continuous-time (analogue) signals. The CWT operates over every possible scale and translation, whereas the DWT uses a specific subset of scale and translation values (Mallat (1999)).

Orthonormal wavelets: A family of functions $\psi_{a,b}$ can be generated from a single function $\psi \in L^2(\mathbb{R})$ by the operation of binary dilations and dyadic translation of ψ, so that

$$\psi_{a,b}(x) = 2^{a/2}\psi(2^a x - b), \tag{3.8}$$

where $a, b, x \in \mathfrak{R}$ $(a > 0)$, and the factor $2^{a/2}$ ensures orthonormality. So, a wavelet $\psi \in L^2(\mathbb{R})$ is called orthonormal if the family of functions $\psi_{a,b}$ can be generated by (3.8).

Haar wavelets: The Haar transform can be thought of as a sampling process in which rows of the transform matrix act as samples of finer and finer resolution. The Haar wavelet's mother wavelet $\psi(t)$ can be defined as

$$\psi(t) = \begin{cases} 1 & 0 \leq t < \frac{1}{2} \\ -1 & \frac{1}{2} \leq t < 1 \\ 0 & \text{otherwise,} \end{cases}$$

and its scaling function $\Phi(t)$ can be described as

$$\Phi(t) = \begin{cases} 1 & 0 \leq t < 1 \\ 0 & \text{otherwise.} \end{cases}$$

Historically, the first orthonormal wavelet basis is the Haar basis. However, although the Haar wavelet ψ has good time-localisation, it does not possess an optimised frequency localisation and its FT $\hat{\psi(k)}$ decays as $|k|^{-1}$ as $k \to \infty$

(Debnath (2002)).

Daubechies wavelets: Daubechies (1988) proposed a series of wavelet bases which have compact support and maximum number of vanishing moments, named after her name as *Daubechies wavelets*. These constitute a family of orthogonal wavelets defining a discrete wavelet transform (DWT) and are characterised by a maximal number of vanishing moments, i.e. moments which are equal to 0, for some given support. In the Daubechies-n wavelet, n represents the order of the wavelet basis function required for reconstructing the input image, which is equivalent to the number of vanishing moments.

Continuous wavelet transforms

By means of the shifting and scaling properties of the FT, the CWTs are constructed by translating and dilating a single mother wavelet, which is localised in both spatial and frequency domains (Livens et al. (1997)). For a prototype function $\psi(x) = L^2(\Re)$, the mother wavelet, the family of functions can be obtained by shifting and scaling this $\psi(x)$ as:

$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}}\psi_{a,b}\left(\frac{x-b}{a}\right), \tag{3.9}$$

where $a, b, x \in \Re\,(a > 0)$. Parameter a is a scaling factor and b is a shift factor. Normalisation ensures that $\|\psi_{a,b}(x)\| = \|\psi(x)\|$. The mother wavelet has to satisfy the following admissibility condition

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\Psi_{a,b}(\omega)|}{\omega} d\omega, \tag{3.10}$$

where $\Psi_{a,b}(\omega)$ is the FT of $\psi(x)$, defined by Petrou and García-Sevilla (2006) as:

$$\Psi_{a,b}(\omega) = \sqrt{a}\Psi(a\omega)\exp(jb\omega). \tag{3.11}$$

In practice $\Psi(x)$ will have sufficient decay so that the admissibility condition reduces to

$$\int_{-\infty}^{\infty} \psi(x)dx = \Psi(0) = 0. \tag{3.12}$$

Thus, the wavelet will have bandpass behaviour. The CWT of a function $f(t) \in \Re$ is then defined as:

$$W_c f(x) = \int_{-\infty}^{\infty} \psi *_{a,b} (t) f(t) dt. \tag{3.13}$$

Changing a or b in equations (3.9) and (3.10) generates two families of functions which constitute two equivalent sets of elementary texture information. Every wavelet transform corresponds to a high- and low-pass filter, and decomposes images into sub-images. Every sub-image contains information of a specific scale and orientation, which is conveniently separated. Spatial information is retained within the sub-images. Through CWT analysis, a set of wavelet coefficients $W_c f_{(a,b)}$ is obtained. These coefficients indicate how close the signal is to a particular basis function.

Discrete wavelet transforms

As parameters a and b take continuous values, the resulting $W_c f(x)$ in equation (3.13) becomes a very redundant representation. The wavelet given by (3.9) can be discretised by constraining a and b to a discrete lattice ($a = 2^n, b \in \mathbb{Z}$). The imposed constraints are that the transform should be non-redundant, complete, and should constitute a multi-resolution representation of the original signal (Petrou and García-Sevilla (2006)).

The discretisation is performed by setting $a = a_0^j$ and $b = k\, a_0 b_0$ for $j, k \in \mathbb{Z}$, where $a_0 > 1$ is a dilation step and $b_0 \neq 0$ is a translation step. The family of wavelets then becomes

$$\psi_{j,k}(x) = a_0^{-j/2} \psi(a_0^{-jx} - k b_0), \tag{3.14}$$

and the wavelet decomposition of a function $f(x)$ is

$$f(x) = \sum_j \sum_k W_d f_{j,k}(x) \psi_{j,k}(x), \tag{3.15}$$

where the 2-D set of coefficients $W_d f(j,k)$ is called the DWT of a given function $f(x)$.

The selection of $\psi(x)$ is made in such a way that the basis function set $\psi_{j,k}(x)$ is an orthonormal basis of $L^2(\Re)$, so that

$$W_d f(x) = \int_{-\infty}^{\infty} \psi^*_{a,b}(t) f(t) dt. \tag{3.16}$$

The most widely used form of such discretisation with $a_0 = 2$ and $b_0 = 1$ is described as the standard DWT (Daubechies (1990)). A 2-D DWT is often referred as the *pyramid-structured wavelet decomposition* (PSWT), e.g. in Wang and Yong (2008).

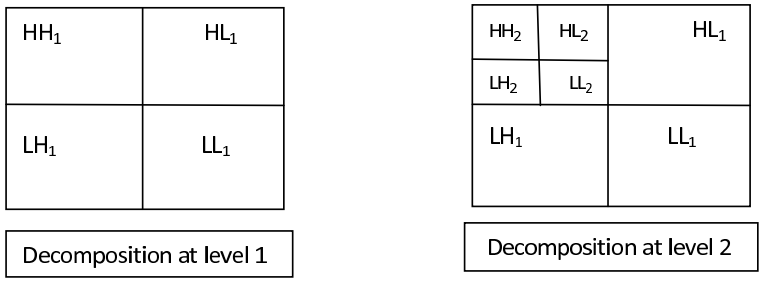


Figure 3.3: Two levels of a 2-D discrete wavelet decomposition.

The DWT decomposes a given image into different scale components. Each scale component can then be studied with a resolution that matches its scale. Figure 3.3 represents the DWT of an image at two different levels. First, the input image is decomposed into four sub-bands, labelled as HH_1, HL_1, LH_1 and LL_1. The first sub-band HH_1 corresponds to an approximation image and the last three represent detail images. To do further decomposition, the sub-band HH_1 is decomposed into four sub-bands again, among which the top-left is the approximation sub-band representing the coarse coefficients and the rest correspond to the detail images. This decomposition continues until the desired level is obtained. Different statistical features, such as mean, standard deviation, energy and entropy, defined below, can be calculated from each sub-band at different scales or levels and used as texture descriptors for classification.

Energy can be based on either the square or absolute values of the pixel intensities. Entropy provides a measure of randomness of an image. The higher the entropy, the greater the variability in the image grey levels. For a constant image entropy would be zero. Entropy is often considered as a histogram-based measure, which involves quantising the image intensities according to the number of bins used to produce the histogram. The histogram is then normalised to produce relative frequencies and entropy is computed from the normalised histogram. For any $M \times N$ sub-band image with pixel intensities $f_{i,j}$, the traditional way of computing mean, standard deviation (sd), energy and entropy is:

$$mean = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} f_{i,j},$$

$$sd = \sqrt{\frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (f_{i,j} - mean)^2},$$

$$energy = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\|f_{i,j}\| \quad \text{or} \quad \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} f_{i,j}^2$$

$$entropy = -\sum_{i=1}^{K} p_i \log_2 p_i,$$

where p is the normalised grey level histogram with K bins and $\log_2$ is the base 2 logarithm.

Wavelet packet transform

Wavelet packet decomposition (WPD) can be derived from the 2-D DWT, which offers a richer space-frequency representation. The key difference between 2-D DWT and WPD is that for WPD, at the second and or further decomposition levels all sub-images are decomposed into four sub-images (3 detail sub-bands and 1 approximation sub-band), whereas for the 2-D DWT only the approximation sub-image is decomposed into smaller four sub-images. The WPD decomposes a 1-D signal in all low and high frequency regions and wavelet decomposition of a 2-D signal can be achieved by applying the 1-D wavelet decomposition along rows and column of the image separately (Wang and Yong (2008)).

Applications of wavelet transforms

Wavelet transformations are now very commonly used in texture analysis. Some of its recent applications are mentioned here.

Tsiaparas et al. (2011) compared different multi-resolution approaches for feature extraction for use in texture classification of ultrasound images of carotid atheromatous plaque (into symptomatic and asymptomatic classes). They applied DWT, stationary wavelet transform (SWT), WPD and the Gabor transform, and computed statistical features, i.e. the mean and sd of the detail sub-images from each decomposition scheme. The features were plugged into a support vector machine and a probabilistic neural network, and higher classification was obtained with WPD features using the support vector machine.

Wavelet-based texture features were found useful for discriminating benign and malignant micro-calcification (MC) clusters on mammograms for breast cancer diagnosis by Karahaliou et al. (2008). DWT coefficients at three decomposition levels and grey-level co-occurrence matrix features were computed from images of regions of interest, and performance of the combined features was investigated using a probabilistic neural network.

Wang and Yong (2008) developed a texture classification algorithm based on the wavelet transform combined with linear regression. It was proposed to analyse correlation between pairs of sample images obtained by wavelet decompositions rather than features extracted from those sample images, as it was believed that the spatial correlation between sample images belonging to the same kind of texture at different scales of WPD is distinct and able to characterise the texture. A 2-D WPD was applied instead of 2-D DWT as the former captures more spectral information in the image. Firstly, the original image was decomposed using a 2-D WPD and the energy of all four sub-images was computed. They used n levels of decomposition, where n was such that the final sub-images were larger than 16^2, and computed the energy for each sub-image. This gave $k = 4^n$ sub-images. For a single image, the energy vector is of length 4^n, called the *channel-energy vector*. Similarly, by decomposing all j sample images belonging to the same texture, they formed the *channel-energy-matrix*, $M_{j\times k}$, where rows correspond to different sample images and columns represent energies from different sub-images. A $k \times k$ covariance matrix with k rows and k columns was derived from $M_{j\times k}$, where $c_{i,j}$ is the correlation coefficient ρ between the i^{th} and j^{th} frequency channels. The channel-pairs with correlation $\rho > T$ where T is a threshold value, were considered to be the more informative frequency regions and were used in the computation. They extracted the energy values $(x_1, y_1)', \ldots, (x_n, y_n)'$ for two frequency channels (X and Y) from each such pair and employed linear regression by assuming that energy from X was a cause of the energy from Y. For any energy from channel X, the corresponding residual was considered as a texture descriptor. For any channel pair, if the residual lay within $\mu\pm3\sigma$, it was concluded that it belonged to frequency channel Y. They applied their method to 40 640^2 grey level Brodatz texture images and compared it with other traditional multi-resolution methods, such as the PSWT, the tree-structured wavelet transform, and the Gabor transform and found that their method performed better in most cases.

Traditional wavelet transforms were criticised by Chaux et al. (2006). Firstly, as they are not shift-invariant, their performance is usually limited by the shift variance with respect to the value of the transformed coefficient at a given scale. Secondly, in higher dimensions, standard wavelets possess poor directionality properties, which are very important for feature detection. They proposed an M-band wavelet transform which does not suffer from those drawbacks and has advantages over several classical dyadic orthonormal wavelet bases.

Jafari-Khouzani and Soltanian-Zadeh (2005) proposed a new rotationally in-

variant technique using Radon and wavelet transforms. They first calculated the Radon transform of the image and then used a translation-invariant wavelet transform in each of the four frequency sub-bands LL, LH, HL, and HH (diagonal, vertical, horizontal and approximation sub-band images) produced by two levels of decomposition of an image to calculate the frequency components and extract corresponding features. The proposed method was applied to 54 Brodatz texture images of size 512^2, using four different wavelet bases and different numbers of neighbours in the k^{th} nearest-neighbour classifier. The rate of correctly classified pixels was 95.6%.

Unser (1995) used a discrete wavelet frame (DWF) to characterise texture properties. He suggested that the DWF should perform better than most traditional single resolution approaches. The classification performance of the DWF was tested on 12 Brodatz texture images, each histogram-equalised with a requantisation to 32 grey levels, so as to be indistinguishable on the basis of first-order statistics only. A Gaussian maximum-likelihood classifier was used and trained on all images except the one being classified. Classification accuracy of 99.2% was obtained for almost all the texture images and performance was compared with the traditional wavelet transform. The classification results showed that the DWF was superior to the discrete wavelet transform.

Complex wavelet transform

Although the wavelet transform is a powerful image processing tool, it has four fundamental shortcomings. Firstly, it is not shift invariant, i.e. a small shift of the image grey levels greatly affects the wavelet coefficient oscillation pattern around singularities. Secondly, a higher-dimensional wavelet transform suffers poor directionality when the transform coefficients reveal only a few feature orientations in the spatial domain. Thirdly, the wavelet coefficients of an image do not contain any phase information, as filtering the image with a 2-D DWT increases its size and adds phase distortion. Fourthly, the wavelet coefficients tend to oscillate between positive and negative values around singularities, which complicates wavelet-based image processing (Kingsbury (2005)).

The complex wavelet transform is a complex-valued extension of the standard DWT. It is a 2-D wavelet transform which provides multi-resolution, sparse representation, and useful characterisation of image structure. Complex wavelet transforms (CLWTs) can be broadly classified in two groups, i.e. redundant CLWTs and non-redundant CLWTs.

One of the most promising redundant type of CLWTs is the dual-tree complex

wavelet transform (Kingsbury (2001)) which overcomes the drawbacks of the standard DWT. Two classical wavelet trees with real filters are developed in parallel, with the wavelets forming (approximate) Hilbert pairs. A dual tree of real wavelet filters is used to generate the real and imaginary part of the complex wavelet coefficients. The requirement for the dual-tree setting for forming Hilbert transform pairs is the well-known half-sample delay condition. The resulting complex wavelet is then approximately analytic, i.e. approximately one-sided in the frequency domain (Daubechies (1990)).

Since the FT does not suffer from the drawbacks of the DWT, Kingsbury (1999) built a wavelet transform with a complex-valued scaling function and complex-valued wavelet that decomposes the real/complex images into real and imaginary parts. The real and imaginary coefficients are used to compute amplitude and phase information. The filter bank structure of the CLWTs resembles the filter bank structure of the standard DWT but with twice the complexity. Two sets of filters h_x and g_x, each consisting of a high-pass and a low-pass filter, are jointly designed such that the complex wavelet transform $\psi(t) := \psi_h(t) + j\psi_g(t)$ is approximately *analytic*, or $\psi_g(t)$ is approximately the Hilbert transform of $\psi_h(t)$ (Kingsbury (2005)).

The properties of the dual tree complex wavelet transform can be summarised as: approximate shift invariance, good directional selectivity in 2-D, providing phase information, allowing perfect reconstruction, limited redundancy and efficient order-N computation (only twice that of the simple DWT for 1-D).

3.4 Model-based Approaches

As mentioned earlier, the model-based approach includes Markov random fields and auto-regressive models. There are variations of the Markov random field, among which the Gaussian Markov random model will be briefly discussed.

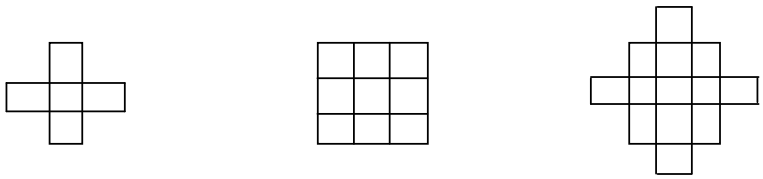
3.4.1 Markov random fields

A *random field* is created by performing a random experiment at each location of the field and assigning the outcome of the experiment to that location. A random field is called a *Markov random field* (MRF) if it possesses the *Markovian property*. A process possesses the Markovian property if the conditional probability distribution of the future state of any process, given the present and past states, depends only upon the current state, i.e. it is conditionally independent of the past states (the path of the process) given the present state.

To build a MRF in terms of a coin-tossing experiment, for example, consider an empty grid of size $n \times n$ and fill the spaces by the number of heads obtained by throwing an unbiased coin 255 times for each grid position. This generates a random field. If however the coin is biased, so that the outcome in a specific grid location depends on the values of the neighbouring locations, a Markov random field arises (Petrou and García-Sevilla (2006)).

An image is called Markovian if the probability distribution of the intensity at any specific location directly depends on the values of the neighbouring intensities (Reulke and Lippok (2008)). MRFs are used for modelling images, and have been applied to many aspect of image processing, such as texture synthesis, texture classification, image segmentation, image restoration, and image compression (Tuceryan and Jain (1998)). The introduction of MRFs in a Bayesian framework has resulted in a unified, coherent framework that enables treating many image processing problems as statistical inference problems (Krishnamachari and Chellappa (1997)).

The choice of neighbourhood depends on the type of random field used. The first-order neighbourhood of a pixel consists of its four-connected neighbours, and the second-order neighbourhood of a pixel consists of its eight-connected neighbours (see Figure 3.4, which shows some commonly used neighbourhoods in a MRF).



(a)

Figure 3.4: First-order neighbourhood (left); second-order neighbourhood (middle); third-order neighbourhood (right) of the central pixel.

Reulke and Lippok (2008) modelled MRFs using a Gibbs distribution. A is assumed to be a Markovian image, for which the probability distribution of the intensity of P_i depends on the intensities of the neighbouring pixels W_i (for a specified neighbourhood). In the Gibbs distribution, the conditional probability that pixel P_i takes value p_i given that the neighbouring pixels W_i have value p_j is

$$P_r(P_i = p_i \mid W_i = p_j) = \frac{\exp\{-h(p_i, p_j)\}}{\sum_{p=0}^{P} \exp\{-h(p, p_j)\}},$$

in which P is the maximum intensity value, and h is a parametrised energy function characterising the MRF. There is no unique functional form of the energy

function since it depends entirely on the problem under study.

The MRF technique was successfully used by Reulke and Lippok (2008) for segmentation of roads in panchromatic images for traffic observation from an aeroplane platform, for example. They assumed the image to be a MRF and used an auto-binomial model as the underlying energy function. To estimate the parameters they used *maximum pseudo-likelihood* estimation, and approximated the overall probability by the product of all conditional probabilities (Besag (1986)). The model parameters were used to characterise the texture, and can be used for texture segmentation. After estimating parameters, segmentation of Brodatz textures was carried out. They also investigated some factors, e.g. influence of parameter normalisation, size of the texture window, size of the neighbouring system, image quality, and image scaling, which have a significant effect on texture segmentation. By using normalised parameters, the segmentation error was reduced to 2%, whereas the un-normalised parameters yielded 16% segmentation error.

Thakoor et al. (2007) applied the *hidden Markov model* to characterise the shape of a texture and then used a weighted likelihood method to estimate the model parameters. Wu and Chung (2007) introduced a boundary MRF model, which can yield appropriate segmentation even with complex boundaries and is robust to noise corruption. This study concerned medical image segmentation rather than texture segmentation.

Apart from texture classification and segmentation, MRF models have also been used for creating textures (texture synthesis) (Petrou and García-Sevilla (2006)). They created an empty 64×64 grid where the grey value at each position was drawn from the binomial probability density function with parameters $n = 8$ and $\theta = 0.5$, to create a random field. Then for each pixel (i, j) they computed the sums

$$s = g_{i,j-1} + g_{i,j+1} - g_{i-1,j} - g_{i+1,j}, \tag{3.17}$$

where $g_{i,j}$ is the grey value of pixel (i, j), using a first order neighbourhood structure. Using these values of s, they computed the binomial parameter θ as a function of the values of the neighbours using the following relation

$$\frac{\theta}{1+\theta} = e^s.$$

Then the new value for a specific pixel was drawn using the probability density function with the new parameter. In the first step they updated the grey values only for the pixels with non-overlapping neighbours and left the remaining pixels

with their old values. In the next step, they updated the next set of pixels with no overlapping neighbours, and carried out this procedure until all sets of pixels with distinct neighbourhoods had been assigned updated values.

Gaussian Markov random model: When the outputs of the random experiment performed to determine each pixel value arise from a Gaussian probability distribution and the parameters of the probability distribution are functions of the values of the neighbouring pixels, the image forms a Gaussian MRF (GMRF).

A simple form of a GMRF model is given by Petrou and García-Sevilla (2006) as

$$p(g_{ij} \mid g_{i\prime j\prime \in N_{i,j}}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ \frac{(g_{ij} - \sum_{l=1}^{L} a_l g_{i\prime j\prime})^2}{2\sigma^2} \right\}, \tag{3.18}$$

where $p(g_{ij} \mid g_{i\prime j\prime \in N_{i,j}})$ is the probability of pixel (i, j) having grey value g_{ij}, given the values of its neighbours (i, j), L is the total number of pixels in the neighbourhood N_{ij} of pixel (i, j) and a_l is the parameter with which a neighbour influences the value of pixel (i, j). The maximum likelihood method can be used to estimate the parameters of the Gaussian Markov model, and the estimated parameters are used to characterise the texture.

Lehmann (2011) used GMRF based features for segmentation of 2-D texture images by modelling them as the concatenation of two 1-D hidden Markov autoregressive models for the rows and the columns, respectively. A segmentation algorithm was developed using the concept of *turbo decoding* used for error-correcting codes.

Krishnamachari and Chellappa (1997) presented multi-resolution models for texture segmentation using a GMRF. They estimated the GMRF parameters either by minimising Kullback-Leibler distances or based on local conditional distribution invariance. Since the data at lower resolutions can be approximated by a GMRF, given the number of classes and the associated parameters at the fine resolution, the GMRF parameters at lower resolutions were obtained by the conditional distribution invariance approximation. Then segmentation was performed at the coarsest resolution and the results of segmentation were passed on to the most immediate higher resolution and the process repeated until the finest resolution was reached. They tested their method on Brodatz textures and compared its efficiency with a single resolution approach. The percentage of correctly classified pixels reached 92.8% for the Brodatz textures, and for synthetic textures it was 96.8%. They concluded that their method performed better than the single resolution method both in terms of classification accuracy and computational

requirements.

3.4.2 Auto-regressive model

The auto-regressive model for texture classification can be characterised both as a statistical approach and a model-based approach. A considerable amount of work has been done on development of models for random field image processing (Jain (1981)). In a 2-D auto-regressive (AR) model the grey level of each pixel is represented as a linear weighted sum of the grey levels of its neighbouring pixels, with addition of white noise (Deguchi (1986)).

For an image with intensities $f(x, y), x, y = 1, 2, \ldots, N$, the AR model is given by

$$f(x, y) = \sum \sum_{(p,q) \in D} a_{p,q} f(x - p, y - q) + n_{xy}, \tag{3.19}$$

where n_{xy} is Gaussian white noise with zero mean and variance σ_n^2, a_{pq} are coefficients used to characterise the texture, and D is a rectangular neighbourhood region defined as

$$D = \{(p, q) : -D_x \leq p \leq D_x, -D_y \leq q \leq D_y, (p, q) \neq (0, 0)\},$$

where D_x and D_y are the neighbourhood dimensions.

Alata and Ramananjarasoa (2005) used a 2-D quarter plane auto-regressive model for an unsupervised textured image segmentation. They first estimated both the number of textures and the model parameters associated with each existing texture, and used a simulated annealing method for maximum posterior estimation of the specific region. Zheng (1997) showed that the estimated parameters of an AR model based on a noisy image are asymptotically unbiased. The AR model is not usually rotationally invariant. Mital and Leng (1992)modified it to be rotationally invariant under all configurations of pixels for texture analysis, and found that the modified model gave better results than the original model.

3.5 Statistical Approaches

In statistical texture analysis, texture features are computed from the statistical distribution of observed combinations of intensities at specified positions relative to each other in the image. According to the number of intensity points in each combination, statistics are classified into first-order, second-order and higher-order statistics. First-order ones use the grey level histogram, so are based on

single pixel intensities, e.g. grey level mean or sd. Second-order ones use 2-D combinations of pixel grey levels, for example grey level co-occurrence measures, defined below.

Among existing statistical approaches to texture analysis, auto-correlation based texture features, grey level co-occurrence matrices, grey level run-length distribution techniques, local binary pattern, improved local binary pattern and coordinated cluster representation are discussed here. Although some authors mention morphological granulometry as a statistical method of texture analysis, we discuss it separately in Section 4.1, as it is used extensively in the work in this thesis.

3.5.1 Auto-correlation based texture features

The auto-correlation function (ACF) of an image can be used to assess the regularity as well as the coarseness of texture. It evaluates the linear spatial relationships between texture primitives (basic shapes in the image). If the primitives are large, the function decreases slowly with increasing distance, whereas it decreases rapidly if the texture consists of small primitives (Sharma and Singh (2001)). The ACF of an image $f(x, y)$ of size $M \times N$ can be defined as

$$\rho(x, y) = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} f(i, j) f(i + x, j + y)}{\sum_{i=1}^{M} \sum_{j=1}^{N} f^2(i, j)}. \tag{3.20}$$

This can be used directly as a signature, or by inferring the periodicity of texture from it, or by extracting parametric features from it to characterise a texture (Petrou and García-Sevilla (2006)). The ACF is the Fourier transform of the power spectral density function and vice versa. A useful property of the auto-correlation function is mentioned in Kurita and Otsu (1993), i.e. it is shift-invariant. As a result they successfully used the higher-order local auto-correlation features for texture classification.

The practical implementation of the ACF for identifying textures is described by Petrou and García-Sevilla (2006). The ACF $\rho(x, y)$ can be used directly to represent the texture and compare it point by point with the auto-correlation function of another texture to see how similar or dissimilar the two textures are. The comparison can be done by computing any statistical measure of similarity (e.g. the sum of squares of the differences, or the correlation coefficient between two functions) or by plotting $\rho_x(x) \equiv \sum_y \rho(x, y)$ versus x and $\rho_y(y) \equiv \sum_x \rho(x, y)$ versus y and using these curves as the texture signatures.

3.5.2 Grey level co-occurrence matrices

The *grey level co-occurrence matrix* (GLCM), or the grey level spatial dependence matrix, is one of the most popular ways to characterise image texture. Co-occurrence matrices are based on second-order statistics, that is, the spatial relationships of pairs of grey values of pixels within a specified region (Haddon and Boyce (1993)).

For an $M \times N$ image containing G grey levels, the GLCM is a $G \times G$ matrix with entries $C(i, j)$, such that $C(i, j)$ is the number of pairs of pixels, at distance d apart and lying on a line at angle φ to the reference direction of the image, with grey levels i and j respectively (Clausi (2002), Petrou and García-Sevilla (2006)). If the texture is coarser in one direction than the other, then the degree of spread of the values about the main diagonal in the GLCM will depend on the direction φ.

The normalised GLCM, $c(i, j)$, is obtained by dividing each entry of $C(i, j)$ by the sum of the $C(i, j)$, i.e.

$$c(i, j) = \frac{C(i, j)}{\sum_{k=1}^{M} \sum_{l=1}^{N} C(k, l)}$$

which normalises the co-occurrence values to lie between 0 and 1 and provides the joint frequency distribution of pairs of pixels with grey level i and j, at a given direction and specified distance apart.

Rather than using all possible grey levels (0-255), the original image's grey levels may be scaled down (quantised) to a smaller number, which reduces the size of the GLCM to give a less sparse matrix. It is expected that coarser quantisation may reduce both classification accuracy and feature space separability of the classes. The literature suggests use of different quantisations, e.g. 8, 16, 32, 64, or the difference between maximum and minimum intensities, as it is not guaranteed that a higher level of quantisation will lead to better classification accuracy in any given type of image (Clausi (2002)). According to Soh and Tsatsoulis (1999), use of quantisation at level 256 is not necessary, 8 level quantisation is undesirable (as it is too coarse) and 64 level quantisation is efficient and sufficient, and the inter-pixel distance is more important than the orientation.

GCLMs capture texture properties but are not directly useful for further analysis, such as comparison of two textures. Haralick et al. (1973) and Haralick et al. (1979) proposed some texture features that can be computed from the GLCM for more compact texture representation, including:

1. Maximum probability: $\max_{(i,j)} c(i, j)$

2. Energy: $\sum_{i=1}^{G}\sum_{j=1}^{G} c(i,j)^2$

3. Entropy: $-\sum_{i=1}^{G}\sum_{j=1}^{G} c(i,j)\log c(i,j)$

4. Contrast: $\sum_{i=1}^{G}\sum_{j=1}^{G}(i-j)^2 c(i,j)$

5. Homogeneity: $\sum_{i=1}^{G}\sum_{j=1}^{G}\frac{c(i,j)}{1+|i-j|}$

6. Correlation: $\frac{1}{\sigma_i\sigma_j}\sum_{i=1}^{G}\sum_{j=1}^{G}(i-\mu_i)(j-\mu_j)c(i,j)$,
   where $\mu_i = \sum_{i=1}^{G} i\sum_{j=1}^{G} c(i,j)$, $\quad \mu_j = \sum_{j=1}^{G} j\sum_{i=1}^{G} c(i,j)$,
   $\sigma_i^2 = \sum_{i=1}^{G}(i-\mu_i)^2\sum_{j=1}^{G} c(i,j)$, and $\sigma_j^2 = \sum_{j=1}^{G}(j-\mu_j)^2\sum_{i=1}^{G} c(i,j)$.

7. Inverse difference moment: $\sum_{i=1}^{G}\sum_{j=1}^{G}\frac{c(i,j)}{1+(i-j)^2}$

8. Autocorrelation: $\sum_{i=1}^{G}\sum_{j=1}^{G} ijc(i,j)$

9. Dissimilarity: $\sum_{i=1}^{G}\sum_{j=1}^{G}|i-j|c(i,j)$

10. Cluster shade: $\sum_{i=1}^{G}\sum_{j=1}^{G}(i+j-\mu_i-\mu_j)^3 c(i,j)$

11. Cluster Prominence: $\sum_{i=1}^{G}\sum_{j=1}^{G}(i+j-\mu_i-\mu_j)^3 c(i,j)$

12. Horizontal mean: μ_i

13. Vertical mean: μ_j

14. Horizontal sd: σ_i

15. Vertical sd: σ_j

The maximum probability is just the value of the most frequent co-occurrence. Energy, also known as uniformity or angular second moment (Gong et al. (1992)), measures textural uniformity, i.e. pixel pair repetitions. For a texturally uniform or homogeneous image a few GLCM elements will be close to 1, while many will be near 0 (Baraldi and Parmiggiani (1995)) so energy will be near to its maximum value of 1. Entropy measures randomness of the image intensity distribution. It is highest when all GLCM entries are of similar magnitude, corresponding to random grey levels, and small when these are unequal, i.e. the entropy for a homogeneous image will be lower than that of an inhomogeneous image.

Contrast measures local variations in the GLCM, whereas correlation measures the association of the grey levels of the specified pixel pairs. Homogeneity, also known as the *inverse difference*, measures closeness of the distribution of the GLCM values to the GLCM diagonal. It will be larger if the pixel pairs take the

same or similar grey levels. Inverse difference moment also measures the relative closeness of the distribution of GLCM values to the GLCM diagonal and has a high value when the high values of the GLCM are near the main diagonal of the GLCM, i.e. the squared difference $(i-j)^2$ becomes smaller. Cluster shade and cluster prominence are measures of asymmetry, so provide measures of skewness in the image. Larger values of cluster shade and cluster prominence indicate lack of symmetry. Lastly, correlation is a measure of linear association between the pixel grey levels of the image.

The GLCM characterises the spatial relationships between the pixel grey levels, and has proved useful in various texture classification applications because of its ability to extract spatial information. GLCM features, i.e. contrast and entropy, were successfully used to segment images of chromosomes by Chanda and Majumder (1988). Soh and Tsatsoulis (1999) obtained 94.2% classification accuracy for SAR sea ice images using co-occurrence matrix features in Bayesian classifiers. Clausi (2002) computed 8 different GLCM features using different quantisations from (0 to 255) of SAR sea ice images and used the features jointly and separately to classify the images. They advocated use of three features such as contrast, entropy, and correlation for quantisations between 24 and 64. GLCM features were used in a self-organising map in de Almeida et al. (2010) to classify Brodatz texture images with 97% classification accuracy. GLCM features and linear discriminant analysis (LDA) were successfully used to classify colour images of colon cancer in Shuttleworth et al. (2002). These features were also found to be useful for classifying colour texture images in Palm (2004).

We use co-occurrence-based features (1–5 and 7 above as commonly used GLCM features) in Chapter 8 to classify synthetic images as well as real images of corroded metal and also of tea granules.

3.5.3 Grey level run length distribution

This approach is also a statistical approach to texture analysis. The grey level run length (GLRL) method is a way of extracting higher-order statistical texture features. The technique was described and applied by Galloway (1975), to extract information in an image from its grey level runs. Run-length statistics capture the coarseness of a texture in specified directions. A run is defined as a string of consecutive pixels which have the same grey level intensity along a specific linear orientation (Chu et al. (1990)). Fine textures tend to contain more short runs with similar grey level intensities, while coarse textures have more long runs with different grey level intensities.

A run length matrix is a 2-D matrix formed by the number of runs of different lengths and grey levels, arranged according to the lengths and grey values. Galloway (1975) computed five features from run-length matrices analogous to the properties used with grey level co-occurrence matrices, which are called *short run emphasis* (SRE), *long run emphasis* (LRE), *grey level non-uniformity* (GLN), *run length non-uniformity* (RLN) and *run percentage* (RP), defined as:

$$SRE = \sum_{i=1}^{G} \sum_{j=1}^{R} \frac{p(i,j)/s}{j^2} \tag{3.21}$$

$$LRE = \sum_{i=1}^{G} \sum_{j=1}^{R} \frac{j^2 p(i,j)}{s} \tag{3.22}$$

$$GLN = \sum_{i=1}^{G} \left(\sum_{j=1}^{R} p(i,j) \right)^2 /s \tag{3.23}$$

$$RLN = \sum_{j=1}^{R} \left(\sum_{i=1}^{G} p(i,j) \right)^2 /s \tag{3.24}$$

$$RP = \sum_{i=1}^{G} \sum_{j=1}^{R} p(i,j)/n, \tag{3.25}$$

where G is the number of grey levels, R is the longest run, s is the total number of runs in the image n is the number of pixels in the image, and $p(i,j)$ is an element of the run length matrix (Chu et al. (1990)).

Albregtsen et al. (2000) successfully used grey level run length matrices for analysing the textures of liver cell nuclei. They combined information from the entries of the normalised run length matrix, based on the class distance matrices, to obtain adaptive features for texture classification. Arul et al. (1993) used two approaches to texture analysis, namely spatial GLCMs and grey level run-length matrices to determining beef quality grades in terms of the distribution (or marbling) of intramuscular fat in a beef segment. The results showed good potential of these approaches for tissue characterisation and objectively evaluating beef quality.

3.5.4 Local binary pattern

Local binary pattern (LBP) was introduced by Ojala et al. (1996) and is the simplest way of extracting texture features. It detects binary texture patterns in

a 3×3 neighbourhood of a grey scale texture image and uses that as a measure of texture. It starts by selecting a region of interest and comparing each pixel with its 8-nearest neighbours, then assigns the value 1 for any neighbour with pixel value greater than the centre pixel value and 0 otherwise, to provide an 8-digits binary number. So there are $2^8 = 256$ binary patterns that can be defined for a 3×3 neighbourhood. An LBP code for an 8-neighbour is

$$LBP = \Sigma_{p=0}^{P-1} s(g_p - g_c) 2^p$$

where $P = 8$, g_c is the intensity of the central pixel and g_p are the intensities of the 8-nearest neighbours, and

$$s(t) = \begin{cases} 1 & t \geq 1 \\ 0 & \text{otherwise.} \end{cases}$$

The LBP for each pixel is computed and stored and can be used as texture descriptor. The basic version of LBP has been extended by introducing different neighbourhoods, e.g. 8- or 16-neighbourhood and circular neighbourhood (Ojala et al. (2002)). Using a circular neighbourhood and considering all rotated versions of the same pattern to be equivalent can reduce the length of the LBP code to 36 from 256. Fernández et al. (2011) proposed an extension of LBP, i.e. improved local binary pattern (ILBP), that assigns labels to each pixel in a 3×3 neighbourhood by using the mean value of the 9 grey levels as a threshold.

3.5.5 Coordinated cluster representation

The coordinated cluster representation (CCR) was introduced by Kurmyshev and Cervantes (1996) to characterise a binary image in terms of a histogram of the occurrence of texture pattern. Computation of the CCR of an $L \times M$ image requires selecting a rectangular window $W = I \times J$, where $I < L$ and $J < M$, and then scanning each pixel of the window. Then binary patterns are found for each pixel of the window and the histogram is formed. The normalised histogram is considered as an image spectrum and used as a texture descriptor (Fernández et al. (2011)). In LBP the central pixel is excluded while computing the LBP code, there are $2^8 = 256$ possible LBP codes, whereas CCR considers all pixels, so there are $2^9 = 512$ LBP codes, which makes LBP more limited than CCR.

Different versions of LBP and CCR are used for texture classification. For example, Fernández et al. (2011) used LBP, ILBP and CCR for classifying granite texture images and found that ILBP provided better classification. A rotation

invariant LBP was developed in Ojala et al. (2002) and applied to 20 types of texture images from the Outex image data base. They considered four rotation angles, namely 0°, 30°, 45° and 60° while computing LBP code and achieved 100% classification rate using histograms computed at all rotations. The classification was based on a chi-square goodness of fit test to compute the dissimilarity of sample images and the model histograms. LBP, ILBP and CCR were also used in Harrison et al. (2011) to identify sabellaria spinulosa colonies in sidescan sonar imagery. These features were compared with Gabor filter bank features and dual-tree complex wavelet based features using a 1-NN classifier and it was found that Gabor features provides better classification results.

3.6 Illumination Resistant Texture Analysis

Texture analysis becomes more challenging due to the fact that large variations in the visual stimulus arising from illumination conditions, viewing directions, poses, and disguises are all common in real applications (Eleyan et al. (2008)). Ho et al. (2006) examined perception of the roughness of 3-D textures under changes in illumination and showed that visual perception of 3-D texture is not invariant under changes in lighting direction.

A texture image is a function of texture surface, the illumination, the camera and its viewing position. None of the standard methods of texture analysis consider the effect of illumination direction on texture, surface texture is greatly affected by the illumination direction (Varma and Zisserman (2005)). For example, Chantler et al. (2002) show that two images of the same surface texture sample captured using different illuminant tilt angles look considerably different. Existing texture analysis techniques assume either that image texture is due solely to surface marking or that the source of illumination is omni-directional, and commonly use a set of scanned images from the Brodatz texture album (Brodatz (1966)), to test their performance, whereas many of these texture images clearly violate one or more of these assumptions (Chantler (1995)).

Recently, Hwang et al. (2011) applied FT for face recognition under uncontrolled variation in illumination. They obtained an *integral normalised gradient image* by normalising and integrating the smoothed gradient of a facial image and extracted hybrid Fourier features from different Fourier domains in different frequency bandwidths. They obtained an 81.5% correct verification rate on 2-D face images under various illumination directions.

3.7 Comparative Studies of Existing Methods

Galloway (1975) showed that the grey level co-occurrence technique is better than the grey level run-length method for texture analysis. Kurita and Otsu (1993) extracted higher-order local auto-correlation features from 30 texture images and showed that the local auto-correlation method achieved a 93.2% recognition rate, whereas the higher-order local auto-correlation features achieved 99.6% accuracy.

One comparative study of texture classification methods by Dettori and Semler (2007) found that wavelet-based features performed worse in texture classification than GLCM and the grey level run length measures when they were applied to 340 images of size 512^2. The accuracy rates for the wavelet-based texture descriptors ranged between 85%-93%, whereas GLCM had accuracy rates of 94%-97% and the grey level run length measures had accuracy rates of 91%-98%. The classification accuracy of auto-correlation and co-occurrence based methods was compared by Sharma and Singh (2001) on images in the Meastex database, and co-occurrence performed better than the auto-correlation method, having recognition rates of 79.2% and 76.1% respectively.

Tang (1998) developed a new run length method based on the dominant run length method and the Bhattacharyya distance measure, and observed that the grey level run length features performed comparably with co-occurrence features and better than wavelet features. The grey level run length method achieved 97% accuracy on 8 Brodatz image classes, with 225 images in each class and 99.9% accuracy on 16 Vistex images with 225 images in each class. Co-occurrence and wavelet features were used on 16 Vistex images with accuracy rates of 100% and 98% respectively.

Ayala and Domingo (2001) proposed use of granulometric features from a spatial size distribution, for shape and texture analysis. They compared their method with the MRF approach, GLCM, the Gabor method and fractal dimension. They concluded that granulometry performs at least as well as GLCM when granulometry is applied to the foreground image. The MRF and Gabor methods had results extremely close to their method, however their method required fewer features than the MRF and Gabor methods.

Unser (1995) experimented with the DWT on 12 Brodatz textures and found that the DWT did better than most traditional single resolution approaches.

Although mathematical morphology is a well established approach in image processing for vector-valued (grey level) images, it has limited application so far in 3-D matrix-valued (colour) images. Recently, Burgeth et al. (2007a) extended the fundamental concept of mathematical morphology to 3-D matrix-based im-

ages based on the *Loewner ordering* (developed in Horn and Johnson (1994)) of symmetric matrices in higher dimensions, and Burgeth et al. (2007b) generalised non-linear partial differential equations (PDEs) that simulate dilation and erosion to the 3-D setting. The corresponding non-linear system of PDEs provides a novel way of using mathematical morphology in 3-D. But both papers focused on image processing rather than texture analysis.

MRFs involve an energy function which has no unique form. So the choice of the energy function requires expert knowledge. Mostly the optimisation schemes associated with parameter estimation are iterative, which requires use of a further simulation algorithm, e.g. simulated annealing, which adds an additional computational load. In practice, auto-correlation based texture features are easier to use for classifying textures.

GLCM concerns the spatial arrangement of objects in the image. It is almost guaranteed to produce distinct features for different textures, but the computational burden is the main drawback of its use. Rotation invariant GLCM features can be obtained by averaging features from different orientations, e.g. $0°, 90°, 45°$ and $135°$.

Gabor filters possess some advantages although the sinusoidal plane wave is computationally very expensive. Wavelet transforms require some assumptions on the parameters, and also the optimisation procedure is a constrained optimisation which is loaded with computational burden.

The pattern spectrum is considered to be a very powerful tool for analysing textures, but it also possesses some drawbacks. The opening operation on which the granulometry is based is very sensitive to image noise. Presence of noise can drastically change the pattern spectrum of an image. Also it does not consider the spatial arrangement of image objects, thus images of different structures may have very similar pattern spectra (Zingman et al. (2007)). Using granulometry on the image background overcomes this (see Section 4.3). Regardless of the drawbacks, the use of morphological granulometry has an important role in shape-based texture analysis and is the main method used in the work of this thesis to obtain features for texture classification.

3.8 Texture Classification Rules

Classification is the process of assigning classes to objects. More precisely, classification is the assignment of new objects of unknown class to one of a number of pre-defined classes, based on a set of variables or features (Hand (1981)). In the

statistical approach, each texture pattern is represented in terms of d features or measurements and is viewed as a point in d-dimensional space.

Given a set of observed measurements represented as a pattern vector $\boldsymbol{x}$, the texture classification problem is to assign the pattern to one of C possible classes π_i, $i = 1, \ldots, C$. The aim of classification is to choose a combination of these features that allow pattern vectors or feature vectors belonging to different categories to occupy compact and disjoint regions in the d-dimensional feature space. A decision rule partitions the measurement space into C sub-spaces Φ_i, $i = 1, \ldots, C$. The effectiveness of the feature space is determined by how well patterns from different classes can be separated (Jain et al. (2000)). If an observation vector is in region Φ_i then it is assumed to belong to class π_i (Webb (2002)).

There are two main divisions of classification: namely *supervised classification* and *unsupervised classification*. In supervised classification a set of data samples with associated classes is used in classifier design. The supervised component in this classification methodology refers to the user-defined training classes. It is important that these classes are a homogeneous sample of the respective class, but at the same time include the range of possibilities for that class. Unsupervised classification is used to cluster pixels in a dataset based on statistics only, without any user-defined training classes. This is an exploratory data technique which may be used before supervised classification methods.

Some commonly used approaches of texture classification are described below.

3.9 Bayesian Classifiers

A traditional powerful statistical approach to texture classification based on probability theory is *Bayesian classification*. Given the class conditional density functions of a feature vector, the Bayesian classifier can be shown to supply the statistically optimum solution to the problem of supervised classification. Given the *a priori* probabilities $p(\pi_i)$ of the classes π_i, $i = 1, \ldots, C$, the problem is to assign an observed feature vector $\boldsymbol{x}$ to one of the C classes. Intuitively we may assign $\boldsymbol{x}$ to the class with highest prior probability $p(\pi_i)$, but the Bayes decision rule makes use of the likelihood $p(\boldsymbol{x} \mid \pi_i)$ of $\boldsymbol{x}$, and Bayes theorem and assigns $\boldsymbol{x}$ to the class with the largest a posteriori probability $p(\pi_k \mid \boldsymbol{x})$. For C classes, the observed feature vector $\boldsymbol{x}$ is taken to belong to class π_i if

$$p(\pi_i \mid \boldsymbol{x}) > p(\pi_j \mid \boldsymbol{x}) \quad \text{for all} \quad i \neq j, \quad j = 1, \ldots, \ C. \tag{3.26}$$

In terms of $p(\pi_i)$ and the class conditional densities $p(\boldsymbol{x} \mid \pi_i)$,

$$p(\pi_i \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \pi_i)p(\pi_i)}{p(\boldsymbol{x})}. \tag{3.27}$$

So the decision rule using Bayes theorem may be written as: assign $\boldsymbol{x}$ to π_i if

$$p(\boldsymbol{x} \mid \pi_i)p(\pi_i) > p(\boldsymbol{x} \mid \pi_j)p(\pi_j) \quad \text{for all} \quad i \neq j. \tag{3.28}$$

A detailed description is given in Webb (2002).

The densities $p(\boldsymbol{x} \mid \pi_i)$ in (3.28) are usually unknown and have to be estimated from the training samples. Very often it is difficult to obtain reliable estimates, but assumptions can be made, e.g. assuming that the class conditional distributions are all multivariate normal. This leads to *quadratic discriminant analysis* (QDA), where the mean vectors and covariance matrices are both different for different classes. The problem can be simplified further by assuming equal covariance matrices for all classes, which converts the QDA to *linear discriminant analysis* (LDA) (Hand (1981)), now described.

3.9.1 Linear discriminant analysis

Linear discriminant analysis (LDA) assumes that the class conditional densities $p(\boldsymbol{x} \mid \pi_i)$ are multivariate normal, with identical covariance matrices, i.e.

$$p(\boldsymbol{x} \mid \pi_i) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right], \tag{3.29}$$

where p is the number of feature variables.

Classification is achieved by assigning a pattern to a class for which the posterior probability $p(\pi_i \mid \boldsymbol{x})$ is the greatest. So the discriminant rule is to assign $\boldsymbol{x}$ to class π_i if $g_i > g_j$, for all $j \neq i$, where

$$g_i(\boldsymbol{x}) = log(p(\pi_i)) - \frac{1}{2}log(|\Sigma|) - \frac{1}{2}(\boldsymbol{x} - \mu_i)^T\Sigma^{-1}(\boldsymbol{x} - \mu_i) \tag{3.30}$$

where $p(\pi_i)$ is the prior probability of class i, μ_i and Σ are the mean and covariance matrix of the i^{th} class and can be replaced by the maximum likelihood estimates of $\hat{\mu}_i$ and $\hat{\Sigma}$ based on a training sample.

LDA is a widely used statistical pattern recognition and texture classification approach which can work very well. Bekios-Calfa et al. (2011) applied LDA for gender recognition and found that LDA on a linearly selected set of features can achieve accuracies as high as for the state-of-the-art classifier SVM.

Palaniappan and Huan (2005) used LDA and MLP neural networks to classify

electro-encephalogram signals from five different tasks with features extracted using an autoregressive model. In most cases, LDA provided superior classification performance.

3.9.2 Maximum likelihood classifier

Like LDA the (Gaussian) maximum likelihood (ML) classifier also assumes that the class conditional probabilities are normal and so they can be described by a mean vector and a covariance matrix. With equal *a priori* probabilities for each class, classifying a feature vector $\boldsymbol{x}$ to the class to which it has the highest posterior probability $p(\pi_i \mid \boldsymbol{x})$ of being a member is the same as classifying to the class with the highest likelihood $p(\boldsymbol{x} \mid \pi_i)$.

3.9.3 Minimum distance classifier

The minimum distance (MD) classifier uses the mean vector of each class, but ignores information about how the classes are distributed in the feature space. It characterises each class by its mean position. The mean pattern vector of class j is

$$m_j = \frac{1}{N_j} \sum_{\boldsymbol{x} \in \pi_j} \boldsymbol{x} \qquad j = 1, 2, ..., C. \tag{3.31}$$

where N_j is the number of training feature vectors from class π_j. Using the Euclidean distance to determine closeness reduces the problem to computing the distance measures

$$D_j(\boldsymbol{x}) = \|\boldsymbol{x} - m_j\| \qquad j = 1, 2, ..., C. \tag{3.32}$$

Alternatively, a given pattern $\boldsymbol{x}$ of unknown class may be classified to π_k if its Mahalanobis distance $D_M(\boldsymbol{x}, \pi_k) = (\boldsymbol{x} - m_k)^T \Sigma^{-1} (\boldsymbol{x} - m_k)$ to π_k is smaller than those to all other classes, i.e.

$$\boldsymbol{x} \in \pi_k \quad \text{iff} \quad D_M(\boldsymbol{x}, \pi_k) = \min\{D_M(\boldsymbol{x}, \pi_i) \;\; i = 1, \ldots, C\}.$$

3.10 K-Nearest Neighbour Classifier

The K-nearest neighbour approach is a simple non-parametric technique which does not require *a priori* assumptions about the distributions of the training samples. For a dataset containing points in C classes, to assign a point x into one

of the C classes, the K-nearest neighbour (K-NN) classifier draws a hyperplane around x, which encircles the nearest K points to x and assigns x to the class for which N_i/K is largest, where the numerator is the number of points amongst the K neighbours which are in the i^{th} class. For $K = 1$, K-NN becomes the nearest neighbour and x is assigned to the same class as throughout the closest point to x in the training set (Webb (2002)).

3.11 Artificial Neural Networks

An artificial neural network (ANN) is an information processing prototype based on the operation of biological neural networks, such as the human brain, which processes information. A neural network consists of units (neurons), arranged in layers, which convert an input vector into some output, e.g. a prediction. Each unit takes an input (data), applies a (often nonlinear) function (the activation function) to it and then passes the output on to the next layer (Izenman (2008)). Many activation functions, such as sigmoid, softmax and logistic functions can be used in neural networks.

An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation: the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not) for particular input patterns. In the using mode, when a trained input pattern is encountered at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

A more sophisticated neuron is the *McCulloch-Pitts* neuron. A McCulloch-Pitts model uses components which have some of the characteristics of real neurons. A real neuron has a number of inputs which are 'excitatory' and some which are 'inhibitory'. What the neuron does depends on the sum of inputs. The excitatory inputs tend to make the cell fire and the inhibitory inputs make it not fire, i.e. not pass on the signal (Izenman (2008)).

Consider a cell that gives out a binary state, zero or one, on or off. The inputs then carry a binary signal and the only thing that matters is the number of 'on' signals on the excitatory versus the inhibitory inputs. If an inhibitory input is on, the cell cannot fire (i.e. is off) no matter what the excitatory inputs are doing. So the McCulloch-Pitts neuron can be defined precisely as a cell which can output a 0 or a 1, which has a number of excitatory inputs, a single inhibitory input and uses a threshold value to produce its output. At time t it looks at its excitatory

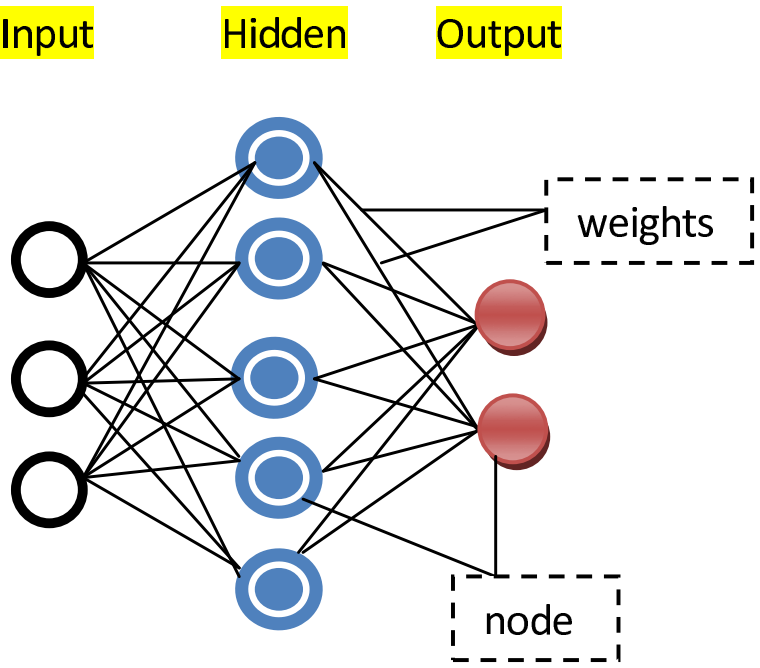


Figure 3.5: A feed-forward single hidden layer neural network.

inputs and counts the number of ones present. If the count is equal to or greater than the threshold and the inhibitory input is zero, then at time $t + 1$ the cell outputs a one, otherwise it outputs a zero.

3.11.1 Types of neural network

There are various sorts of neural networks. Some of them are discussed briefly below.

Feed-forward neural network: Feed-forward ANNs allow signals to travel one way only; from input to output. The data processing can extend over multiple (layers of) units, but no feedback connections are present, i.e. no connections extending from outputs of units to inputs of units in the same layer or previous layers. This type of organisation is also referred to as bottom-up or top-down (Zheng et al. (2010)). A single hidden layer feed-forward neural network (FFNNET) consisting of 5 units or neurons in the hidden layer is shown in Figure 3.5. A classical example of feed-forward neural networks is the *Multilayer perceptron* (MLP).

Multilayer perceptrons: This is the most common neural network model. This multivariate statistical technique non-linearly maps an input vector $X =$

$(X_1, \ldots, X_r)$ of variables to an output vector $Y = (Y_1, \ldots, Y_s)$ of variables. It also has a layered structure like feed-forward neural networks. Between the inputs and the output variables there are *hidden* variables arranged in one or more layers. Each layer receives input units from a layer directly below and sends their output to units in a layer directly above the unit (Izenman (2008)). The hidden and output variables are called *nodes*, *neurons* or *processing units*.

The training of an MLP is usually done using a *backpropagation* (BP) algorithm that involves two phases (Rumelhart et al. (1986)), the *forward phase* and the *backward phase*. In the forward phase, the weights of the network are fixed and the input object data $\boldsymbol{x}_i$ are propagated through the network layer by layer. The forward phase finishes with the computation of an error measure $e_i = d_i - y_i$, where d_i is the desired response and y_i is the actual output produced by the network in response to the input object $\boldsymbol{x}_i$. During the second phase, the partial derivatives of the error measure e_i with respect to the different parameter values are propagated through the network in the backward direction, hence the name of the algorithm. The network weights can then be adapted using any gradient-based optimisation algorithm. The whole process is iterated until the weights have converged (Haykin (1998)).

Recurrent neural networks: A network of neurons with feedback connections is called a recurrent neural network (RNN), e.g. the human brain is a RNN. This type of neural network is also known as a feed-back network as it contains feedback connections. Schmidhuber et al. (2007) define a RNN as a mathematical abstraction of a biological nervous system which may accomplish complex mappings from input sequences to output sequences. According to Graves et al. (2009), as a RNN contains self-connected hidden layers, previous information remains in the network's internal state, allowing it to make use of past information, hence it is capable of producing better classification results. They found that a RNN was superior for recognising unconstrained handwriting than a hidden-Markov model. Gomez et al. (2008) developed an on-line controlling system using a RNN which can be used efficiently for aircraft or robot control.

Examples of recurrent networks are *Kohonen feature maps* and *Hopfield neural networks*. In a Kohonen feature map each node is fully connected with the input layer. The Hopfield network consists of a set of n inter-connected neurons which update their activation values asynchronously and independently of other neurons (Egmont-Peterson et al. (2002)).

Learning vector quantisation: The learning vector quantisation (LVQ) is another algorithm for learning classifiers from labelled data samples. Instead of modelling the class conditional densities directly, it models the discrimination function defined by the set of labelled *codebook vectors* and the classification is based on a nearest neighbour search between the codebook vectors and the input data, i.e. a data point x_i is assigned to a class k according to the class label of the closest codebook vector, i.e. for which $argmin_k \|x_i - m^k\|$ is a minimum, where m_k is the centre of the k^{th} class (Hollmén et al. (2000)). Codebook vectors represent the centres of different classes and are of the same dimension as the input data. These are known as neurons in other forms of neural networks.

Applications of neural networks: Some recent applications of neural networks are mentioned here. Yang and Lunetta (2008) applied MLP neural networks, a maximum likelihood classifier and a decision tree for cropland mapping of the Great Lakes Basin (GLB) in the USA using Normalised Difference Vegetation Index (NDVI) time series data. MLP neural networks produced better results than the other classifiers for classifying cropland versus non-cropland areas across the entire GLB. Borah et al. (2007) applied learning vector quantisation (LVQ) and MLP networks to classify images of tea granules, using wavelet-based features. They obtained 80% classification accuracy using LVQ networks whereas MLP yielded 74.7% accuracy. Neural networks were also used successfully to predict the stability of RNA/DNA hybrid duplexes in Ma et al. (2004). An unsupervised neural network was developed in Kumar and Manolakos (1997) which can perform segmentation and labelling of objects in an image.

3.12 Support Vector Machines

The support vector machine (SVM) is a supervised classifier derived from machine learning theory by Cortes and Vapnik (1995), with a strong theoretical foundation. Currently, it is widely used in object detection and recognition, content-based image retrieval, text recognition, biometrics, speech recognition, etc. SVM is a powerful classification technique, which is robust in producing high classification accuracy even in high-dimensional data spaces with non-linearly separable classes (Kim et al. (2002) and Izenman (2008)).

3.12.1 Binary classification

We may encounter a complicated classification problem even in the case of binary classification. In the simplest case the objects in the two classes are linearly separable but for non-linearly separable classes classification is a challenging task. We discuss both situations here.

For binary classification, consider n training objects $\{x_j, y_j\}_{j=1}^{N}$, where the $\boldsymbol{x}_j$ are input features, $\boldsymbol{x}_j \in \Re^m$ and $y_j \in \{-1, +1\}$ are class labels. For an unknown object $\boldsymbol{x}$, the SVM tries to find an optimum hyperplane using the principle of *structural risk minimisation* so that the distance between the nearest objects and the hyperplane is maximised. This distance is known as the *margin* and the nearest objects are known as *support vectors*. The decision function can be written in terms of the support vectors as:

$$f(\boldsymbol{x}) = sign\left[\sum_{i=1}^{sv_s} \alpha_i y_i \boldsymbol{x}_i^T \boldsymbol{x} + b\right], \tag{3.33}$$

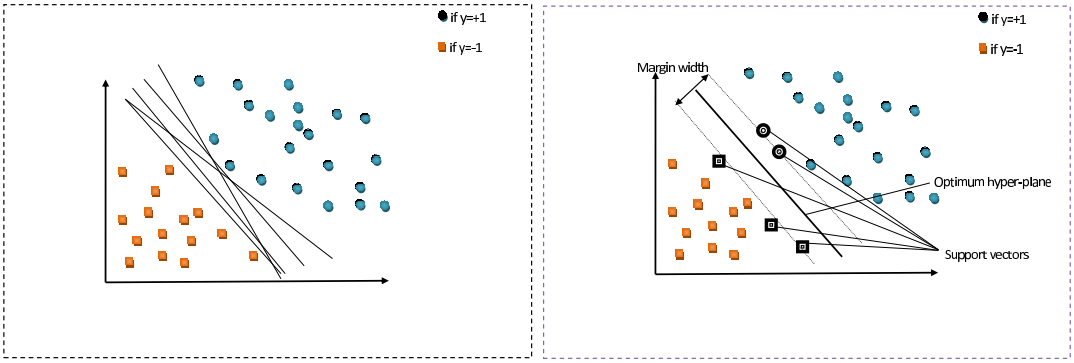
where the α_i are coefficients of the above optimisation equation (Lagrange multipliers), b is an off-set parameter, the subset $\{\mathbf{x_i}\}_{\mathbf{i=1}}^{\mathbf{sv_s}}$ is the set of support vectors, $\boldsymbol{x}$ is the object we wish to classify and the sign of the function determines the class membership of $\boldsymbol{x}$, hence $f(\boldsymbol{x})$ becomes ± 1. For each support vector $\boldsymbol{x}_i$, the decision function, $f(\boldsymbol{x}_i) > 0$ for $y = +1$, $f(\boldsymbol{x}_i) < 0$ for $y = -1$ and $f(\boldsymbol{x}_i) = 0$ if $\boldsymbol{x}_i$ falls on the optimum hyperplane. The values α_i and b are found by maximising

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j,$$

subject to $\alpha_i \geq 0$ and $\sum_i \alpha_i y_i = 0$, where n is the number of training cases and the upper limit of α_i, is known as the cost which penalises classification errors in the training set.

In practice, there may exist many separating hyperplanes which can separate the two classes (Figure 3.6(a)), but the SVM chooses the one which ensures the maximum margin (Figure 3.6(b)).

To deal with classification of a dataset where a linear classifier is not appropriate, Shigeo (2005) adds an extra step in the optimisation procedure. Rather than finding the optimal hyperplane in the original input space, the SVM maps $\boldsymbol{x}$ into a higher dimensional space where the objects are projected by means of a function $\Phi(\boldsymbol{x})$. So the original observations are transformed in a Hilbert space $\Re$ through a non-linear mapping $\Phi : \Re^m \rightarrow \mathfrak{F}$, where $\mathfrak{F}$ is the new feature space. The feature



(a) Several hyperplanes (b) Optimum hyperplane

Figure 3.6: SVM for linearly separable feature space.

space $\mathfrak{F}$ is usually very high-dimensional and the inner product $\langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle$ becomes computationally expensive to evaluate.

To avoid this complexity, Cortes and Vapnik (1995) introduced the so-called *Kernel Trick*. The kernel trick is to use a non-linear kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, instead of projecting then computing the inner product $\langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle$. In the projected space, in terms of the kernel trick the classification equation in (3.33) can be rewritten as

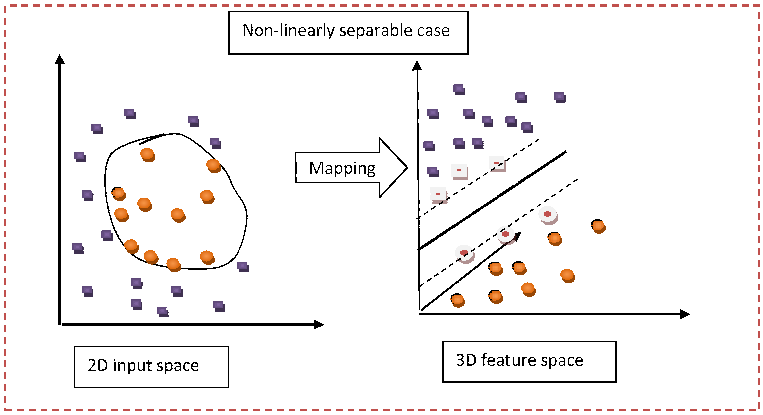
$$f(\boldsymbol{x}) = sign\left[\sum_{i=1}^{sv_s} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right], \tag{3.34}$$

where α_i are Lagrange multipliers, $\boldsymbol{x}_i$ are the support vectors and $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the specified kernel.

Figure 3.7 shows classification of non-linearly separable data using SVMs. Objects are non-separable in the original 2-D input space, so the first step is to project the objects into 3-D space, through a mapping function Φ, where the objects are separable. SVMs find the optimum separating hyperplane in the projected space using decision function (3.34).

The kernel function plays an important role in mapping the input vector onto a high-dimensional feature space. Using different kernels one can construct learning machines with different types of non-linear decision surface in the input space. There are many different kernel functions available. Some commonly used kernels in SVM are listed in Table 3.1, though the radial basis kernel is the most widely used one (Tsiaparas et al. (2011) and Bouguila (2011)).

In some cases when no hyperplane exists which can separate the data into two classes, due to the presence of some overlapping objects. To deal with the overlapping situation, a more flexible situation is suggested, known as the *soft*



(a)

Figure 3.7: SVM for non-linearly separable features.

Table 3.1: Commonly used kernels relating two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$.

Kernel type	Functional form	Parameters
Linear	u′ v	None
Polynomial	$(\gamma \mathbf{u}'\mathbf{v} + \eta)^{\delta}$	$\gamma,\ \eta,\ \delta$
Radial basis	$\exp(-\gamma \|\mathbf{u} - \mathbf{v}\|^{\mathbf{2}})$	γ
Laplacian	$\exp\{-\frac{\|\mathbf{u}-\mathbf{v}\|}{\sigma}\}$	σ
Sigmoid	$\tanh(\gamma \mathbf{u}'\mathbf{v} + \eta)$	$\gamma,\ \eta$

margin solution (Izenman (2008)).

3.12.2 Multi-class support vector machines

Although SVMs were originally developed for binary classification problems, they can be extended to the multi-class situation (Kim et al. (2002)). Izenman (2008) mentions different approaches to tackle the multi-class classification problem, in which the input features $\boldsymbol{x} \in \Re^m$ and $Y \in \{1, 2, \ldots, K\}$, where K is the number of classes. These are:

One-to-rest classification: The K-class classification is first reduced to binary classification sub-problems of the type 'k^{th} class' vs. 'not k^{th} class'. A new $\boldsymbol{x}$ is then assigned to the class with the largest value of $\hat{f}_k(\boldsymbol{x})$, $k = 1, 2, \ldots K$, where $\hat{f}_k(\boldsymbol{x})$ is the optimal SVM solution for the binary problem of the k^{th} class versus the rest.

One-to-one classification: Using $K(K-1)/2$ pairs of classes, a classifier $\hat{f}_{i,j}$ is constructed by coding the i^{th} class as positive and j^{th} class as negative,

$i, j = 1, 2, \ldots K, i \neq j$. Then a new $\boldsymbol{x}$ is assigned to either the i^{th} or j^{th} class using majority voting or with pairwise coupling (which involves pairwise comparison). Each of the $K(K-1)/2$ cases produces a pairwise probability and these probabilities are pairwise coupled into a set of posterior probabilities to make the final class allocation decision.

Li (2009) found that one-to-one classification outperformed one-to-rest classification for classifying copper clad laminate defects using wavelet coefficients and SVM classifiers.

The SVM was originally developed for two-class classification, and for the multi-class case a combination of SVMs is used with possibly lower performance than with binary classification (Weston and Watkins (1998)). Also, for a large scale dataset using an SVM is time consuming. Due to these inherent problems, Kim et al. (2003) constructed the SVM ensemble. They defined an ensemble of classifiers as a collection of several classifiers whose individual decisions are combined using bootstrap aggregating (bagging) to classify the test examples. They successfully applied the SVM ensemble technique to classify the well-known Fisher's Iris dataset (an easy pattern recognition problem), UCI hand-written digits, and for fraud detection. For fraud detection in a mobile telecommunication payment system, a *user profiling* method was used to track suspicious changes in user behaviour and general patterns of fraud were detected by an SVM based on a large amount of user-action data analysis.

The least squares support vector machine (LS-SVM) was recently proposed in Suykens and Vandewalle (1999) by adding a least squares term in the cost function and it involves only the equality constraints in the optimisation phase. The LS-SVM requires significantly less computational time as it has to solve a set of linear equations, whereas training the original SVM means solving a quadratic programming optimisation problem.

3.12.3 Applications of SVMs

Despite being a relatively new classification approach, the SVM has already been recognised as a better classifier in many complex situations, especially for high-dimensional feature spaces. Commonly SVMs are combined with use of wavelet features for classification. Here we describe some recent applications and extensions of SVMs.

Tsiaparas et al. (2011) compared the performance of SVM and probabilistic neural networks using different features sets extracted from ultrasound images of the carotid artery using different multi-resolution techniques, i.e. WDT, WPD,

SWT and the Gabor transform. SVMs performed better on classifying atheromatous plaque into either a symptomatic or asymptomatic class than probabilistic neural networks.

SVMs were found to be useful in the problem of count data modelling using finite mixture distributions in Bouguila (2011). A hybrid deterministic annealing expectation-maximisation was used to estimate the parameters of the mixture model where the number of clusters was selected using the *minimum distance length* criterion. For the UCI dataset the lowest classification error rate of 45% was obtained using a SVM.

Mazanec et al. (2008) did an experiment on the FERET database by applying LDA and SVM for face recognition and found that the highest rate of face recognition was obtained for a combined classifier LDA+SVM. Chaplot et al. (2006) proposed use of SVMs using wavelet features as input to classify magnetic resonance brain images and obtained 98% classification accuracy. Another approach used kernel principal component analysis as a feature extraction method and combined it with SVMs to classify face images (Li and Chen (2005)).

The performance of four different classifiers, i.e. the Bayes classifier, Mahalanobis distance classifier, LVQ and SVM were assessed in Li et al. (2003). Using wavelet coefficients computed from 30 Brodatz texture images, they found that the classification accuracy using SVM could be as high as 95%. They considered 1- to 5-level wavelet decomposition. SVMs produced more accurate classification compared to the Bayes classifier and LVQ.

A surprising application of SVM is found in Kim et al. (2002). Instead of extracting the texture features using any of the conventional techniques, an SVM was directly applied to the grey values of the input image. On average, the classification error rate was below 20%. The potential of SVM for recognising 3-D objects is examined in Pontil and Verri (1998). It is argued that SVM is very useful for direct 3-D object recognition. SVM was found to be one of the best classifiers to classify spam email in Drucker et al. (2002).

SVM provided satisfactory results for detecting defects on copper clad laminate using 2-D wavelet features and the radial basis kernel function, in Li (2009). The highest classification accuracy was from 60% to 90% for various sets of the parameter value γ and the cost, so choosing these appropriately is important for good performance.

3.13 Conclusion

Here we have briefly described the most widely used texture feature extraction techniques, with some recent applications. The granulometric approach is the main focus of the work which is presented in detail in Chapter 4 and the approach is extensively used in Chapters 5–7. To compare with granulometric features, we derived GLCM-based features and wavelet-based features from two sets of real images in Chapter 8. The first six GLCM features listed in Section 3.5 are widely used in the literature, so we use them to classify the synthetic images generated as evolving textures,but we use more of them for the real images in Chapter 8. SVMs, FF-NNETs and LDA are used throughout Chapters 5–8 to compare the performance of the new regression-based classification approach which is developed in Chapter 5 with that of well established techniques.

Chapter 4

Granulometric Approach to Texture Analysis

This chapter gives a detailed description of binary and grey scale granulometry as well as univariate and multivariate granulometry. We have applied granulometry on binary and grey scale images to obtain their *pattern spectrum* and their associated moments which will subsequently be used in Chapter 5 for *texture classification*.

4.1 Morphological Granulometry

Mathematical morphology is a type of non-linear filtering for extracting information from an image. It can be used in many aspects of image analysis, such as enhancement, segmentation, restoration, edge detection, texture analysis and shape analysis (Theera-Umpon and Dhompongsa (2007)). Morphological granulometry was introduced by Matheron (1975) in the binary case to characterise the size and shape information of a random set and it was further developed and extended by many others for grey scale images (Heijmans (1979), Serra (1983), Dougherty and Astola (1994), Goutsias et al. (1995) and Dougherty and Lotufo (2003)).

Granulometry is based on one of the basic morphological operations, usually *opening*. As well as being monotonically increasing, trans-location invariant, anti-extensive, and idempotent (see Section 2.2), opening has another important property which plays a key role in the construction of the granulometry. This is, for a sequence of SEs $E_1, \ldots, E_n$ of increasing size, if $E_{k+1} \circ E_k = E_{k+1}$, then E_{k+1} is said to be E_k *open*, $k = 1, \ldots, n$ (Dougherty and Astola (1994)). The following subsections provide a description of granulometry for both binary and

grey scale images.

4.1.1 Binary granulometry

An *opening granulometry* is based on a sequence of morphological openings using scaled structuring elements. Opening granulometry is carried out by opening a binary image A by a series of scaled SEs $\{E_1, E_2, \ldots, E_n\}$, e.g. successively larger disks, such that E_{k+1} is E_k open. At each stage of opening, the finer details are successively eliminated and the image area (sum of foreground pixels) of the input image is successively reduced. The granulometry is defined as the set of operations $\{A \circ E_k\}$, $k = 1, \ldots, n$, where the SE E is known as the *generator of the granulometry*, k is the scaling factor, E_k is a scaled version of E in the continuous domain and in the discrete domain E_k can be formed by k successive dilations of E by itself (Dougherty and Astola (1994)).

Let $P[A]$ be the original image area and $P[A \circ E_k]$ be the image area left after k successive openings of A by E. As the scale k increases, more image details are removed and eventually the image area drops to zero. Successive openings create a decreasing sequence of images, i.e. $A \circ E_1 \supset A \circ E_2 \ldots \supset A \circ E_n$. A significant drop in image area between two consecutive openings indicates that the image contains many objects smaller than the SE applied at that stage. This decreasing sequence represents the cumulative proportions of the image area dropped at each opening, hence a cumulative distribution can be formed from this sequence. The image area removed by k successive openings can be found by subtracting the image area left after k openings from the original image area, which is known as the *size distribution*, $\Omega(k)$, i.e.

$$\Omega(k) = P[A] - P[A \circ E_k]. \tag{4.1}$$

As k increases, the number of object pixels in $P[A \circ E_k]$ decreases, so $\Omega(k)$ is an increasing function of k. If we divide the image area removed after k successive openings of A by E by the original image area $P[A]$, we obtain the *Normalised size distribution* as:

$$\Phi(k) = \frac{\Omega(k)}{P[A]} = 1 - P[A \circ E_k]/P[A].$$

For sufficiently large k, $P[A \circ E_k]$ could be null, resulting in $\Phi(k)$ being equal to 1 and for a sufficiently small k, $\Phi(k)$ becomes 0. What is 'sufficient' depends on the size of the original image, its area, the shapes of the objects in it, as well

as the shape and size of the applied SE. Since $\Phi(k)$ is monotonically increasing from 0 to 1, it can be considered as a cumulative distribution function (cdf). So, its derivative $\Phi'(k) = \frac{d\Phi(k)}{dk}$ is a probability density function (pdf), known as the *granulometric size distribution* or *pattern spectrum* (PS) (Maragos (1987)) of the image relative to the granulometry.

Since $\Phi'(k)$ is a pdf, it possesses statistical moments, known as *granulometric moments*. In the discrete case, the scaling factor k is an integer, so the m^{th} granulometric moment may be calculated as

$$\mu^{(m)}(A) = \sum_{k=1}^{T} k^m \Phi'(k),$$

where T is the largest size of SE needed to remove all image area. These moments can be used to characterise the pdf and therefore the size distribution of objects, as features describing the texture of the original image.

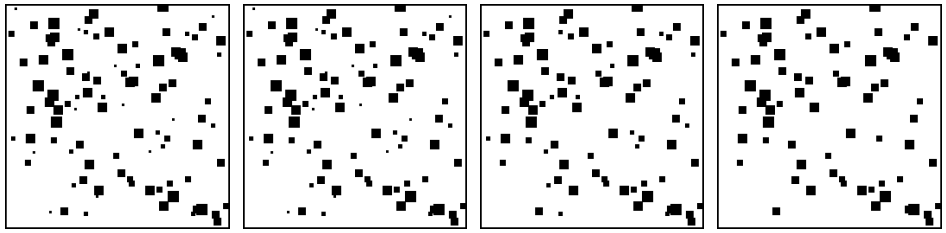
We use these moments to compute the m^{th} central moment about the mean, i.e.

$$v^{(m)}(A) = \sum_{k=1}^{T} (k - \mu^{(1)})^m \Phi'(k).$$

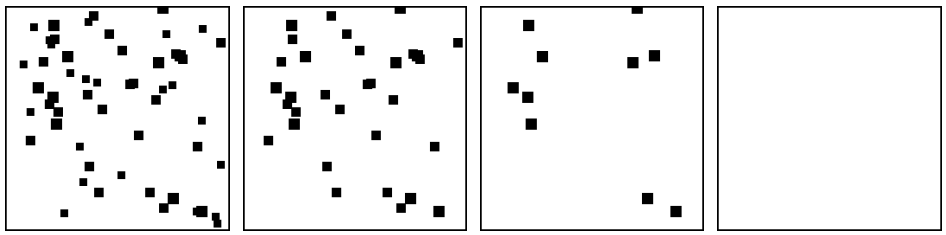
The mean, sd, skewness and kurtosis of the PS are computed using central moments. The first moment is the mean about the origin ($\mu^{(1)}$), the sd (σ) is the square root of the second moment about the mean, the skewness is the ratio of the third moment about the mean to the cube of the sd (σ), i.e. $v^{(3)}/\sigma^3$, and kurtosis is the ratio of the fourth moment about the mean to the square of the variance, i.e. $v^{(4)}/\sigma^4$. We consider the *excess kurtosis* $v^{(4)}/\sigma^4 - 3$, a commonly used correction to make the kurtosis of the normal distribution equal to 0. We use these in later chapters for classification to predict the evolving time state of a texture image using an appropriate model.

Figure 4.1 (a) contains a binary image of size 256×256 consisting of 100 randomly dispersed square objects of different sizes. First the image is padded with zeroes to avoid edge-effects of opening and then it is successively opened by a flat square of size 3, 5, 7, 9, 11, 13 and 15 and the resulting images are shown. As we can see, Figure 4.1 (b) is the same as Figure 4.1 (a) since the smallest square is of size 3. In Figure 4.1(c) all squares smaller than size 5 have disappeared. As the size of the SE increases all squares smaller than the applied SE disappear from the image. When the SE is of size 13 only 9 squares remain in the opened image, and they all drop out by applying a larger SE of size 15.

Figure 4.2 shows the successive drop of pixels, the normalised size distribution



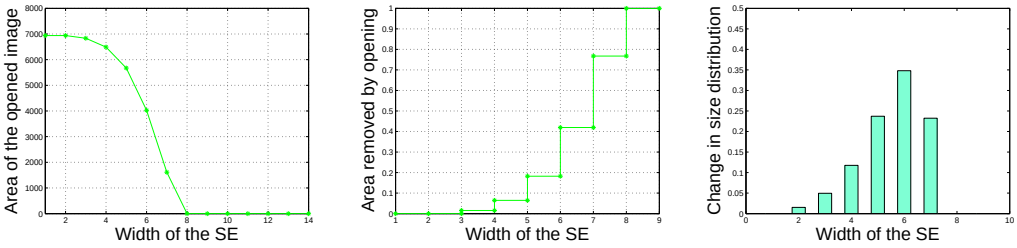
(a) Binary image of squares (b) Opened by 3×3 square (c) Opened by 5×5 square (d) Opened by 7×7 square



(e) Opened by 9×9 square (f) Opened by 11×11 square (g) Opened by 13×13 square (h) Opened by 15×15 square

Figure 4.1: Effect of successive openings of a 256^2 binary image of squares using a square SE of increasing size.

and the pattern spectrum of Figure 4.1 (a). The SE was a square of size $2*j-1$, $j=1,2,\ldots,N$. In Figure 4.2(a), the steepest drop is observed for SE of width 6, as most of the squares are of width 6. The pattern spectrum also reflects this as the highest bar is at width 6, and there were almost equal numbers of squares of width 5 and 7 (the next most frequent sizes).



(a) Successive volume dropped (b) Size distribution (c) Pattern spectrum

Figure 4.2: Successive area dropped, size distribution and pattern spectrum of a 256^2 binary image of squares (Figure 4.1 (a)), from granulometry using a square SE.

4.1.2 Grey scale granulometry

Like binary granulometry, grey scale granulometry is also based on opening. We now think of the grey scale image as having a volume, given by the sum of the pixel intensities or heights in the grey scale surface. A detailed account of grey

scale opening is given in Section 2.5. Grey scale opening of a function f by a structuring element g is given by the supremum of all grey scale translations of g that lie beneath the surface f, i.e.,

$$f \circ g = \sup\{g_x + y : g_x + y \leq f\} \tag{4.2}$$

where $g_x(z) = g(z - x)$ and y is the off-setting parameter.

A grey scale granulometry results from successive openings by expanding the structuring elements to iteratively reduce the grey scale height of an image. Considering a set of structuring elements $\{g_1, g_2, \ldots, g_n\}$ such that $g_{k+1} \circ g_k = g_{k+1}$, then $\{f \circ g_k\}$ is a decreasing sequence. If g_0 is assumed to be a point function, n denotes some arbitrary stopping point for the granulometry, and $\Omega(k)$ is now the difference in volume between $\{f \circ g_k\}$ and $\{f \circ g_{k+1}\}$, the normalised size distribution is defined in the same way as its binary counterpart, i.e. $\Phi(k) = 1 - \Omega(k)/\Omega(0)$, where $\Omega(0)$ is the initial volume of f (Chen and Dougherty (1994)). Having such a normalised size distribution, which is a cdf, its derivative $\Phi'(k) = \frac{d}{dk}\Phi(k)$ can be obtained. It is known as the grey scale pattern spectrum. Since it is a pdf, it possesses statistical moments. The p^{th} granulometric moment may be calculated as

$$\mu^{(p)}(f) = \int_{k=1}^{\infty} k^p \, \Phi'(k) dk,$$

or in practice as the discrete sum

$$\mu^{(p)}(f) = \sum_{k=1}^{K} k^p \, \Phi'(k),$$

where K is the size of the largest SE used to reduce the image volume to zero, as in the binary case.

Again we computed the mean, sd, skewness and kurtosis as for the binary PS moments and employed them for texture characterisation in the later sections and chapters.

4.1.3 Other types of granulometries

Apart from the granulometry generated by successive openings, some others are also available. We will briefly discuss a few of them.

Algebraic granulometry: According to Dougherty and Lotufo (2003), a collection of image operations $\{\Psi_t\}$, $t > 0$, is called an *algebraic granulometry* if it satisfies the following properties:

1. $\{\Psi_t\}$ is an anti-extensive operation for all t,
2. $\{\Psi_t\}$ is increasing for all t, and
3. The order of sieving does not matter, i.e. $\Psi_t\Psi_s = \Psi_s\Psi_t = \Psi_{\max\{t,s\}}$.

The basic granulometry $\{A \circ tB\}$ described in Sub-section 4.1.1 is an algebraic granulometry.

Euclidean granulometry: If a granulometry $\{\Psi_t\}$ is translation invariant as well as following the Euclidean property, namely

$$\Psi_t(A) = t \times \Psi_1(t^{-1}A),$$

for any $t > 0$ and any binary Euclidean image A, and if Ψ_1 represents a unit sieve, then Ψ_1 is called a *Euclidean granulometry*. The most important Euclidean granulometry is a union of openings of scale t by a generator (set of SEs) $g = \{B_1, B_2, \ldots, B_n\}$,

$$\Psi_t(A) = \bigcup_{k=1}^{n} A \circ tB_k. \tag{4.3}$$

Multivariate granulometry: The concept of a multivariate granulometry is introduced by Batman and Dougherty (1997). It can be derived from a Euclidean granulometry by letting the elements of the granulometric generator $g = \{B_1, B_2, \ldots, B_n\}$ have their own parameters $\{t_1, t_2, \cdots, t_n\}$, and assuming that none of these is open with respect to another, i.e. $B_i \circ B_j \neq B_i$ for $i \neq j$. The multivariate granulometry is defined as

$$\Psi_t(A) = \bigcup_{k=1}^{n} A \circ t_k B_k, \tag{4.4}$$

where Ψ_t is an n dimensional granulometry with generator B. The corresponding normalised size distribution is Φ and the multivariate pattern spectrum for A can be defined by the partial derivatives of Φ with respect to $t_1, t_2, \ldots, t_n$. Using more parameters in this way represents a more complete environment to extract texture features, which may give better image information.

Logical granulometry: Logical granulometries are the combination of the *disjunctive granulometry* and *conjunctive granulometry* (Dougherty and Lotufo (2003)). The disjunctive granulometry is based on *reconstructive opening*. Reconstructive opening is the process of reconstruction of an image by opening which requires two SEs. The first SE B determines the shape of the fitting criterion in opening and the second SE E specifies the connectivity for the reconstruction and is usually a 3×3 square SE. The reconstructive opening of A is $\bigcup A \circ_E B_k$.

Using the concept of reconstructive opening, the disjunctive granulometry is defined as

$$\Psi_t(A) = \bigcup_{k=1}^{n} A \circ_E tB_k,$$

where B is the SE specifying the fitting criterion and E is the connectivity for the reconstructive opening.

Conjunctive opening is an inverse operation of reconstructive opening which considers the intersection instead of the union of $A \circ_E B_k$. A conjunctive granulometry is defined as

$$\Psi_t(A) = \bigcap_{k=1}^{n} A \circ_E tB_k,$$

where B and E are the same as in reconstructive opening.

Finally the logical granulometry can be formed by combining conjunctive and disjunctive granulometry as

$$\Psi_t(A) = \bigcup_{k=1}^{n} \bigcap_{j=1}^{m_k} A \circ tB_{k,j}. \tag{4.5}$$

for fixed t and $B_{k,j}$ specifies the shape used in the reconstructive and conjunctive granulometry. The k^{th} opening is taken from the intersection of the reconstructive openings by $B_{k,1}, B_{k,2}, \ldots, B_{k,m_k}$.

4.2 Texture Evolution

To use granulometry and develop our methodology, we have built a database of evolving binary and grey scale texture images of objects, where the number of objects grows with time. We experimented with different parameter settings and different object shapes (disks, squares, ellipses, and pyramids). We have produced sequences of synthetic texture images of pyramids and ellipses, from which texture images of cones can be derived by setting the ratio of the semi-major and semi-minor axes to 1. To generate some synthetic texture images in

different orientations, we rotate the objects at specified angles. The rotation angles of the objects are allowed to vary within given limits.

The underlying concept is as follows. We start with a 2-D blank image of given size and a 3-D blank image of the same size, in which the 3^{rd} dimension is used to store the result obtained at each time step. An initial probability α is set, which determines with what probability a new object appears at a given time step. For a new object we decide its initial grey level from a uniform distribution on a specified range. Once the first object appears, the growth time t starts and is stored, and the location of the object 'centre' is generated randomly and stored. Once a second or subsequent object appears, another probabilistic decision is made to decide if the existing objects are updated. Each existing object is independently updated (or not) with pre-specified probability δ (or 1-δ) respectively. Once a object is to be updated we choose its grey level again from a uniform distribution on a certain range. If an object is updated, we decide its growth rate (γ), i.e. how fast it should grow.

The program records at which point of time an object appears, its location, and the size of each object. The process repeats until growth time becomes 100 and the 3-D image stack is of size 100.

In the simulated images, new objects appear and objects are updated randomly according to probabilistic decisions. As a result, in some cases an object starts to grow at an earlier stage and in other cases only later. Consequently, granulometric moments at a specific time differ considerably from one simulation to another. Such a wide range of moments may produce a wide distribution of predicted texture evolution times, hence high average prediction error rates.

To remove the enormous dissimilarity among images at a specific time, especially at the earlier stages of evolution, we let the evolution time start when the first object appears and run until 100 growth steps are completed. At every subsequent step, we determine whether to add a new object and revisit the existing objects for possible updates.

For the pyramid images, the initial grey level for each pyramid was chosen from a uniform distribution on the range 40 to 60. For updating a pyramid, we draw a random number from a discrete uniform distribution on the range $[1,\ 2]$ for the growth rate γ and allow the pyramids to expand by 1 or 2 pixels on each side. The rotation angle was set to 45° (as using other angles between 0 to 90 had no effect on the shape of the pyramids). The process was repeated until growth time 100 was reached.

A similar method was applied to generate ellipse images. Here the initial

grey level for each ellipse was drawn from a discrete uniform distribution on the range 10 to 20. The growth rate for an updating ellipse is chosen from a discrete uniform distribution on the range [1, 3], which allows the ellipse to enlarge by 1 to 3 pixels around the edges. The shape of an updating ellipse is characterised using the ratio of the semi-major to semi-minor axis of an ellipse, which is chosen from a Gaussian distribution. Different means and sds were experimented with, and it was found that mean 0.666 and sd 0.3 generate a ratio of the semi-major to semi-minor axes which prevents the ellipse from being a straight line or a cone. However, extreme values of the ratios are clipped at 0.1 and 0.9. Ellipses are allow to rotate randomly between 125° to 145° and the process is repeated until growth time 100.

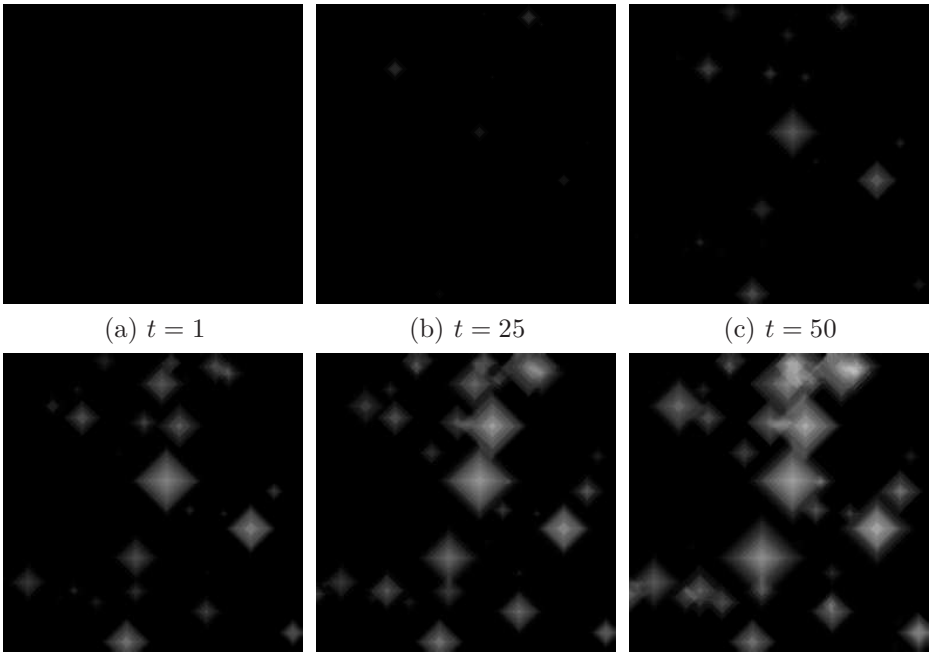
To evaluate our methodology we need several sets of images of different sizes. We have generated image stacks of 3 different sizes, e.g. 100^2, 256^2 and 512^2. Different settings of the underlying parameters (probability of adding a new object α, probability of updating an existing object δ and the growth rate of object γ) are used for different sizes with a view to obtain images with a sufficient number of objects at the final stage of evolution.

For 100^2 images, α takes value 0.5 for pyramids and 0.8 for ellipses, δ takes 0.1 for pyramids and 0.3 for ellipses, and γ takes values according to the discrete uniform distribution on $[1,\ 2]$ for both pyramids and ellipses. For 256^2 images, we increase δ for both pyramids and ellipses to 0.3 and 0.5 respectively and keep α and γ at their previous settings. For 512^2 images, we keep α and δ same as they are for 256^2 images and increase the range for γ to discrete uniform $[1,\ 3]$. The MATLAB code used to generate these images are given in Appendix I.

Figures 4.3 and 4.4 show sequences of images containing pyramids at different time points for the first two parameter settings, whereas the sequences of images containing ellipses evolving over time for the first two parameter settings are shown in Figures 4.5 and 4.6.

Opening granulometry using a disk SE was applied to the foreground of one stack of 256^2 ellipse images consisting of 100 layers (each layer represents one time point) for parameters $\alpha = 0.8$, $\delta = 0.5$, and γ of 1 or 2. A sequence of these images is shown in Figure 4.6. Figure 4.7 contains some of the images, the corresponding image volume dropped and the pattern spectrum.

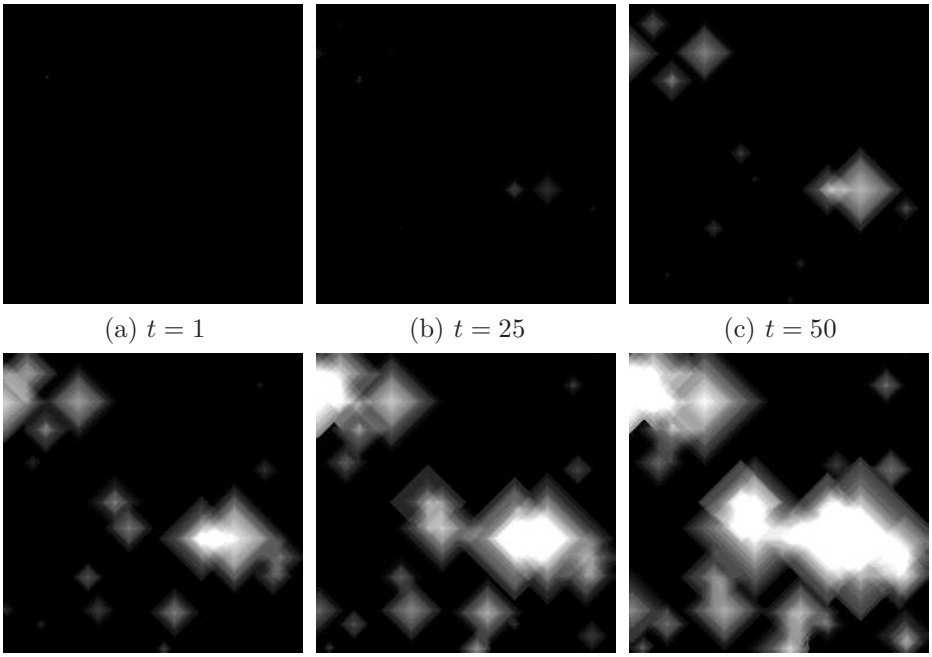
Although granulometry is a useful tool for analysing the size and shape of image content, it does not consider the spatial arrangement of the image content (Zingman et al. (2007)). Applying granulometry on the background of an image does provide such spatial information. We can obtain the image background by



(a) $t = 1$ (b) $t = 25$ (c) $t = 50$

(d) $t = 70$ (e) $t = 85$ (f) $t = 100$

Figure 4.3: Evolution of 100^2 grey scale pyramid images at different time points for parameters $\alpha = 0.5$, $\delta = 0.1$, and γ as discrete uniform $[1,\ 2]$.

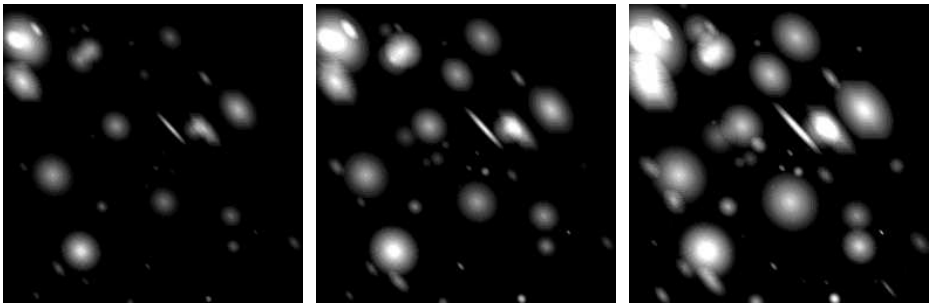


(a) $t = 1$ (b) $t = 25$ (c) $t = 50$

(d) $t = 70$ (e) $t = 85$ (f) $t = 100$

Figure 4.4: Evolution of 256^2 grey scale pyramid images at different time points for parameters $\alpha = 0.5$, $\delta = 0.3$, and γ as discrete uniform $[1,\ 2]$.

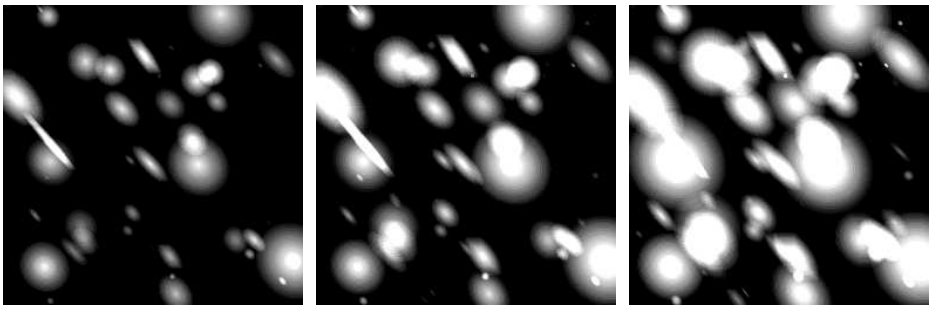
(a) $t = 1$ (b) $t = 25$ (c) $t = 50$



(d) $t = 70$ (e) $t = 85$ (f) $t = 100$

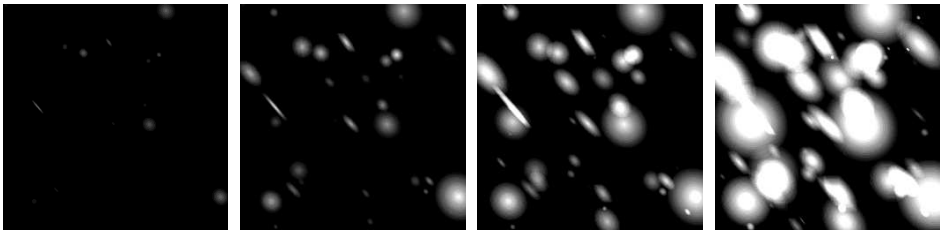
Figure 4.5: Evolution of 100^2 grey scale ellipse images at different time points for parameters $\alpha = 0.8$, $\delta = 0.3$, and γ as discrete uniform $[1,\ 2]$.

(a) $t = 1$ (b) $t = 25$ (c) $t = 50$

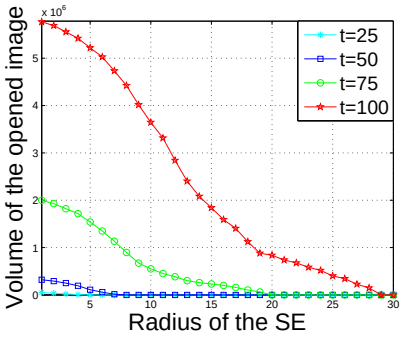


(d) $t = 70$ (e) $t = 85$ (f) $t = 100$

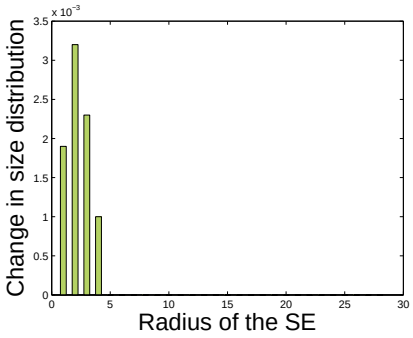
Figure 4.6: Evolution of 256^2 grey scale ellipse images at different time points for parameters $\alpha = 0.8$, $\delta = 0.5$, and γ as discrete uniform $[1,\ 2]$.



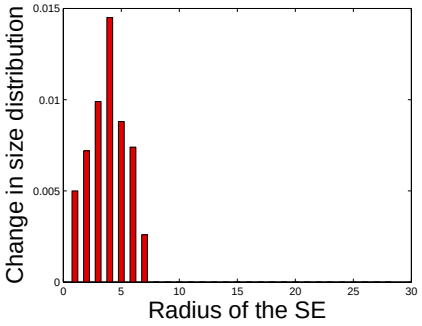
(a) Image at $t = 25$ (b) Image at $t = 50$ (c) Image at $t = 75$ (d) Image at $t = 100$



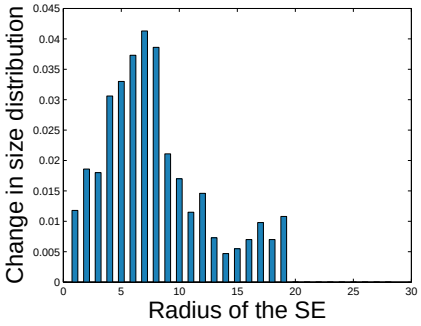
(e) Image volume dropped



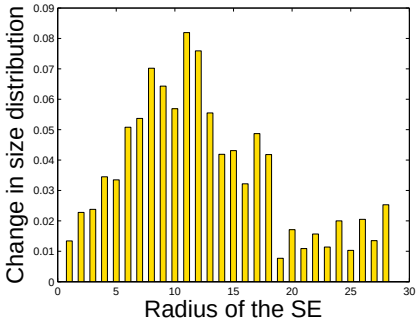
(f) PS at $t = 25$



(g) PS at $t = 50$



(h) PS at $t = 75$



(i) PS at $t = 100$

Figure 4.7: A sequence of grey scale 256^2 ellipse images, the corresponding granulometric size distribution and pattern spectrum using a disk SE.

taking 255 minus image intensity for binary (0/255) images or 8-bit grey scale images.

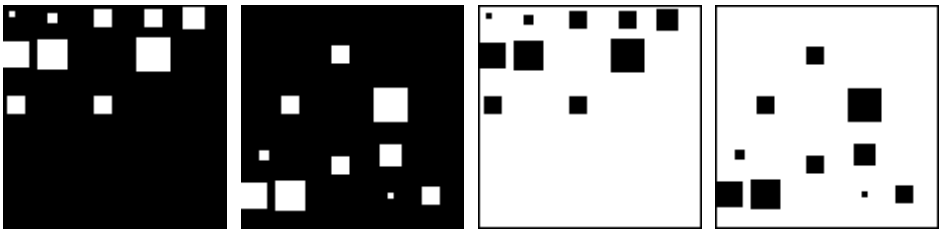
We illustrate this in Figure 4.8, where (a) and (b) are two binary images of squares of size 111^2 containing 10 squares of different widths ranging from 1 to 8. The base size of a square is 2×width+1. The number of objects, total number of foreground pixels and the object sizes and shapes are identical in both images, but they have different spatial arrangements. Their pattern spectra arising from an opening granulometry on the foreground, using a square SE, are identical (Figure 4.8(e)). The bars in the pattern spectrum correspond to the distinct object sizes dropping out of the image. For example, there is one very small square of base size 3 (width 1) which is removed by opening with a square of size 4, and there are 4 squares of base size 9 (width 4) so the highest bar is observed at 9 as they were removed by opening using a square of size 9. In general, applying granulometry using any SE on the foreground of (a) and (b) will produce equal pattern spectra. However, applying the same granulometry on the background of each image (Figure 4.8(c)-(d)) using the same SE, generates different pattern spectra, as shown in Figure 4.8(f). Since the pattern spectra are different their moments are different, hence providing different sets of information.

Similarly, Figure 4.9 shows foreground and background pattern spectra from images of disks, using a disk SE. Clearly foreground and background granulometries provide different but complementary sets of information and the granulometric moments of both may be useful for texture classification (Chen and Dougherty (1994)).

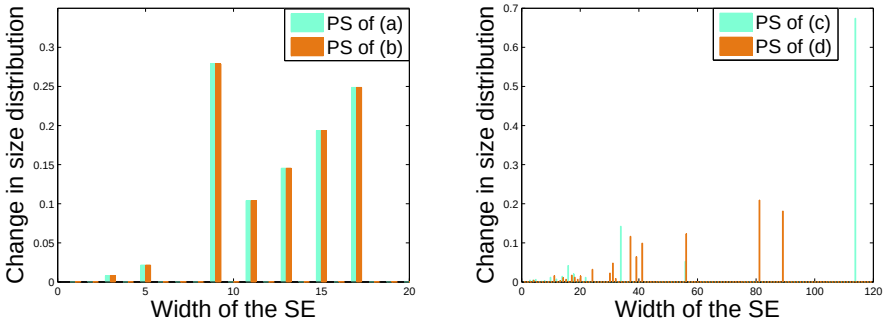
In the following section we apply granulometry on the foreground as well as the background of the pyramid images and ellipse images.

4.3 PS Moments and Evolution Time

Granulometry was applied on the foreground and the background images separately, using six different SEs, namely a square, disk, horizontal line, vertical line, and lines at 45° and 135°. Corresponding pattern spectra were generated from each SE and the first four granulometric PS moments (mean, sd, skewness, and kurtosis) were computed from each pattern spectrum. Average PS moments were then obtained at each time point by averaging over 100 simulations, each one consisting of a single image at each time point $t = 1, \ldots, 100$.

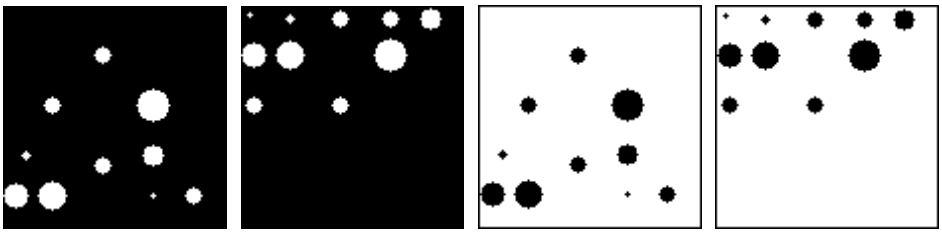


(a) Foreground image of squares (b) Foreground image of squares (c) Background of (a) (d) Background of (c)

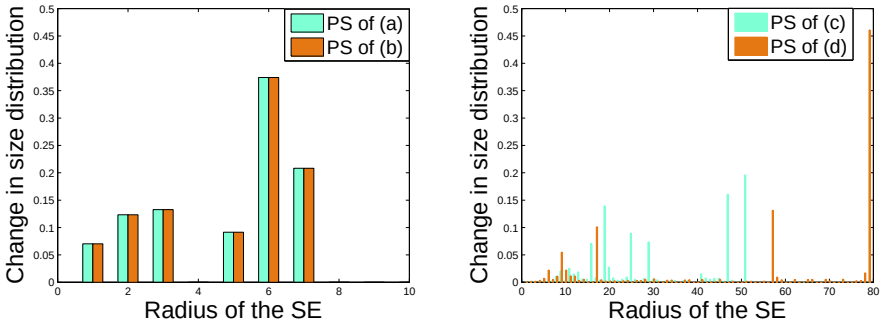


(e) Foreground PS (f) Background PS

Figure 4.8: Effect of granulometry using a square SE on the foreground and background of two binary images of size 111^2, containing squares of width 1 to 8 (base length=2×width+1). Object pixels are shown as white in (a)–(d).



(a) Foreground image of disks (b) Foreground image of disks (c) Background of (a) (d) Background of (c)



(e) Foreground PS (f) Background PS

Figure 4.9: Effect of granulometry using a disk SE on the foreground and background of two binary images of size 111^2, containing disks of different radii ranging from 1 to 8. Object pixels are shown as white in (a)–(d).

4.3.1 Foreground PS moments and evolution time

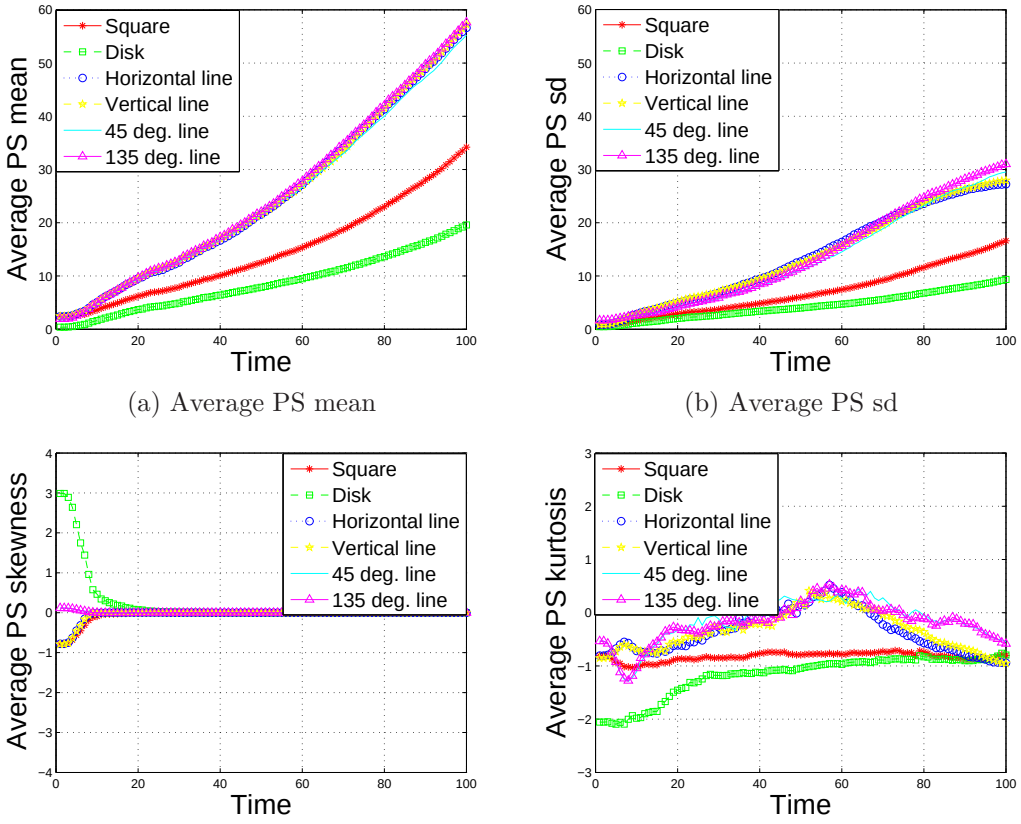
First of all, granulometry was applied on the foreground of 100 stacks of pyramid images of size 100^2, which were generated using parameters $\alpha = 0.5$, $\delta = 0.1$, and γ was chosen from a discrete uniform distribution on the range [1, 2]. The PS moments obtained by averaging over 100 simulations of the image stack were stored in four matrices of size 100×6, where rows represent time steps and columns represent SEs, one matrix for each moment. The average PS moments were plotted against time, to identify any relationship with time.

Figure 4.10 represents the average foreground PS moments for the pyramid images of size 100^2, which shows that the first two moments (mean and sd) using each SE increase with evolution time. The average PS means from all line SEs coincide and have higher magnitude than for the square and disk SE, and the PS mean from the disk has the lowest value throughout the whole evolution period. The same pattern is noticed for PS sd for all SEs. However, the average PS sds for horizontal and vertical line SEs are the same, and the average PS sds for lines at 45° and 135° are the same.

There is no useful relationship between PS skewness and evolution time. If there is a symmetric distribution of differently sized objects in the image, the PS of that image will be symmetric, leading to near zero skewness. Having no object at the beginning, the process starts adding new objects and updating existing ones as time goes on. So in the earlier stages, a few objects appear, each of different size, which forces the PS to be either positively skewed or negatively skewed. If more image volume is dropped in the earlier stages of opening than the later stages, the PS becomes positively skewed. The PS becomes negatively skewed when the opposite occurs. The process is designed so that we are not controlling either the number of objects or the size of the objects with time, but these change randomly. After evolution time 10, the PS became symmetric, hence all SEs produced near 0 skewness, and so PS skewness is not informative about time after that point.

Average PS kurtosis increases over time for a square and disk SE, although the rate of increase is different for each SE, but for any line SE it decreases after time 50. Again there is no clear trend observed between PS kurtosis and evolution time.

Similarly, average PS moments were computed for the pyramid images of size 256^2 and 512^2, each yielding four 100×6 matrices of moments as before, and average moments were plotted against time to identify any relationship with time.



(a) Average PS mean

(b) Average PS sd

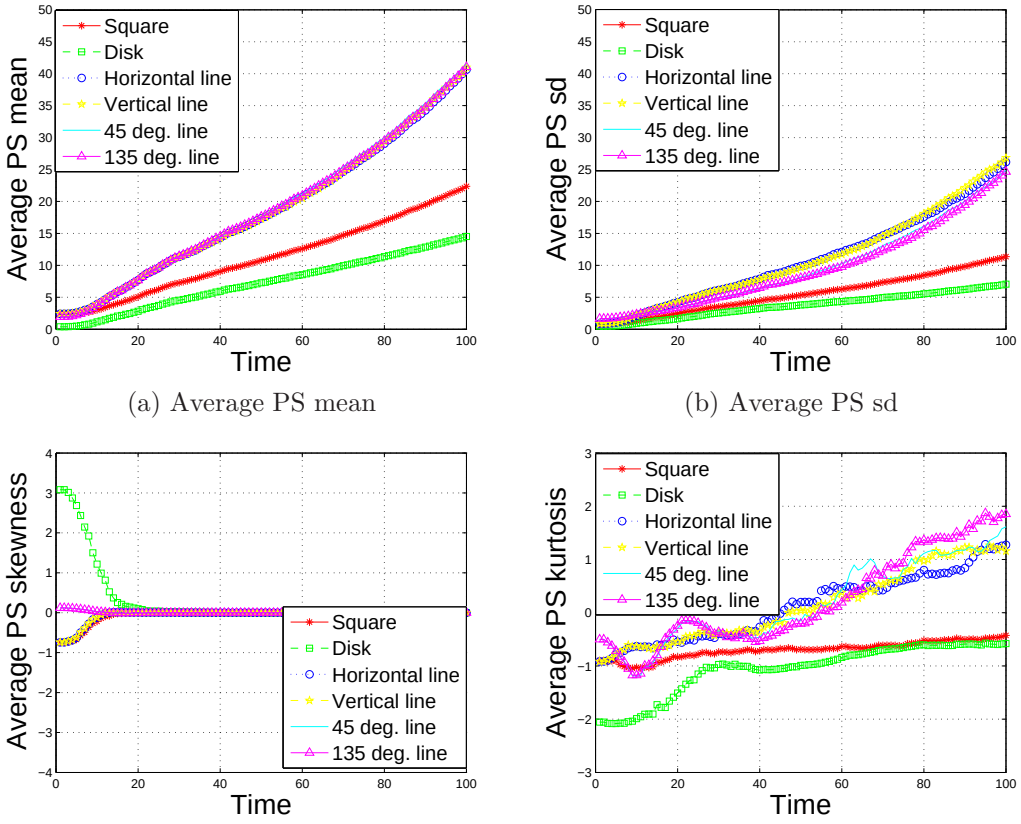
(c) Average PS skewness

(d) Average PS kurtosis

Figure 4.10: Plots of average foreground PS moments (averaged over 100 simulations) against evolution time, using six different SEs, for the 100^2 pyramid images at each time point.

Figure 4.11 shows the average foreground PS moments for the pyramid images of size 256^2. The relationships between any moments and evolution time are very similar to those observed for the 100^2 pyramid images. Here again mean and sd using all SEs increase over time. Any of the line SEs produces a higher mean and sd, while the PS mean and sd from a disk SE are smaller, and the mean and the sd from square SE are between the moments for the line SEs and disk SE. Again skewness becomes near-zero for a square and disk SE through the whole evolution period, and for any line SEs it becomes near-zero just after $t = 15$. Kurtosis now shows a more regular relationship with time than before, but the curves are jagged, especially for any line SE.

Finally, the relationships of the foreground moments of the 512^2 pyramid images with time are shown in Figure 4.12. Similar patterns are observed as in Figures 4.10 and 4.11, with the only exception being that average PS kurtosis generally increases over time and the lines are smoother than before, although the rate of increase is different for each SE. In general, only PS mean and sd



(a) Average PS mean

(b) Average PS sd

(c) Average PS skewness

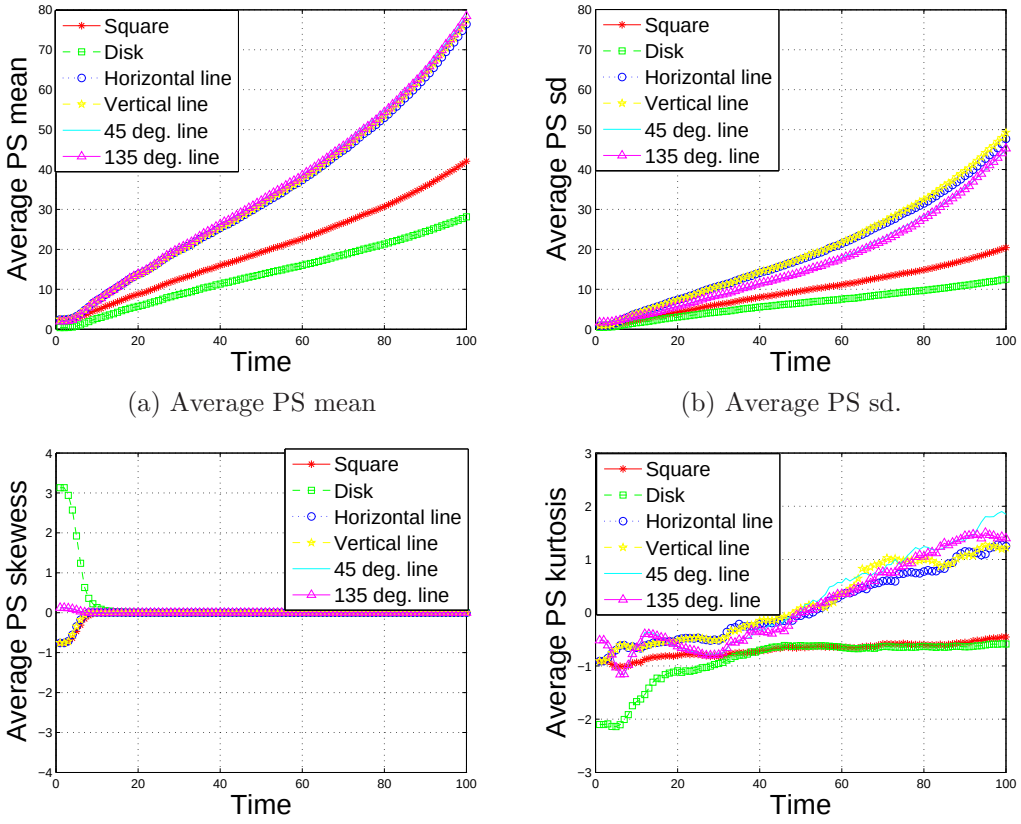
(d) Average PS kurtosis

Figure 4.11: Plots of average foreground PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 256^2 pyramid images at each time point.

provide useful relationships with evolution time, but the skewness and kurtosis do not look useful. Therefore, we used only the PS means and sds from each SE to model evolution time in Chapter 5.

Granulometry was also applied to the foreground of the ellipse image stacks of size 100^2 and 256^2 using the same SEs. We have not used 512^2 ellipse images, as the granulometric computation on larger images is more time consuming and we cannot expect improved relationships of the moments with time, since the PS moments from the 512^2 pyramid images showed very similar patterns with time as those of the 256^2 images (Figures 4.11 and 4.12).

Figure 4.13 shows the relationship of the PS moments with time, which shows that both average foreground PS mean and sd for all SEs clearly increase over time. Average PS skewness for all SEs is near-zero after evolution time 10 except for a disk SE for which the PS is symmetric after time 40. Average PS kurtosis slightly increases throughout the whole evolution period for a disk SE, but does not show any trend for the other SEs. Figure 4.14 represents the correspond-



(a) Average PS mean

(b) Average PS sd.

(c) Average PS skewness

(d) Average PS kurtosis

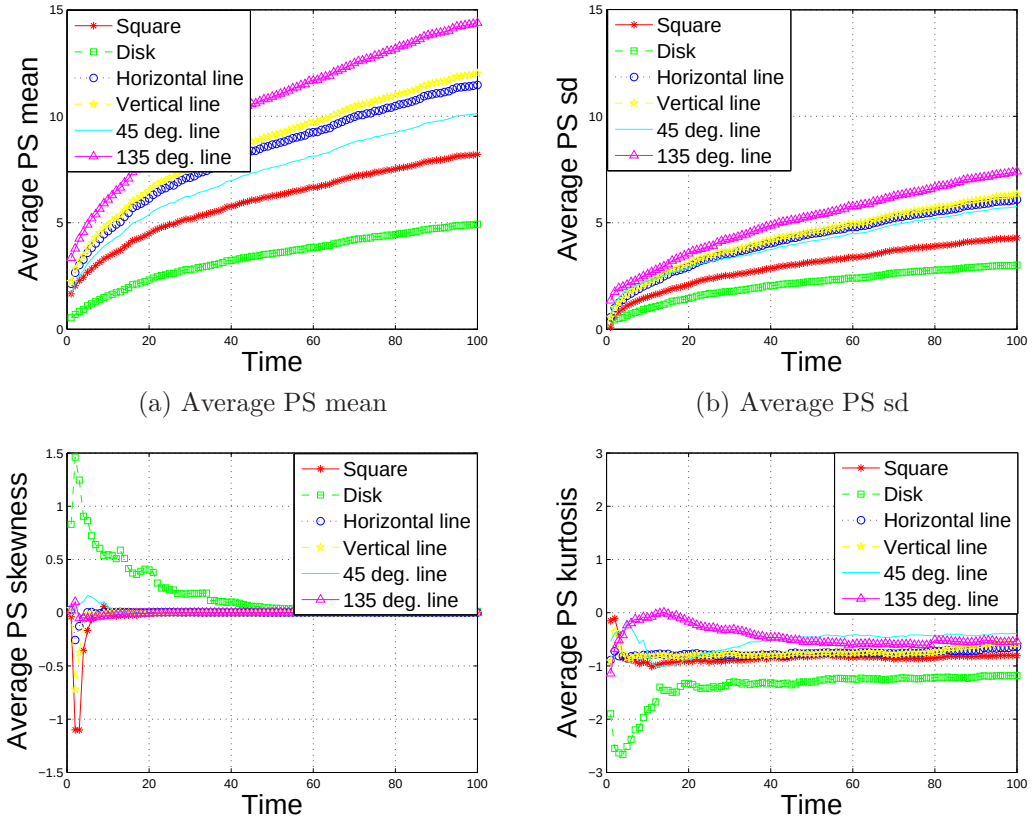
Figure 4.12: Plots of average foreground PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 512^2 pyramid images at each time point.

ing foreground granulometric moments for 256^2 ellipse images. Both average foreground PS mean and sd for all SEs clearly increase over time. Average PS skewness for all SEs is near-zero after evolution time 20, but average PS kurtosis does not show such a clear relation with time.

4.3.2 Background PS moments and evolution time

As the size of the images has little effect on the foreground PS moments relationship with time, we only consider the 100^2 and 256^2 images when computing the background PS moments for both pyramid and ellipse images.

The background granulometries of the same sets of pyramid images, each of size 100^2, were also calculated using the above SEs and the first four average PS moments are plotted against time in Figure 4.15. The background PS means for all SEs clearly decrease as a function of time. Average PS mean for a square SE decreases faster with time than for the other SEs. The average PS means for the



(a) Average PS mean

(b) Average PS sd

(c) Average PS skewness

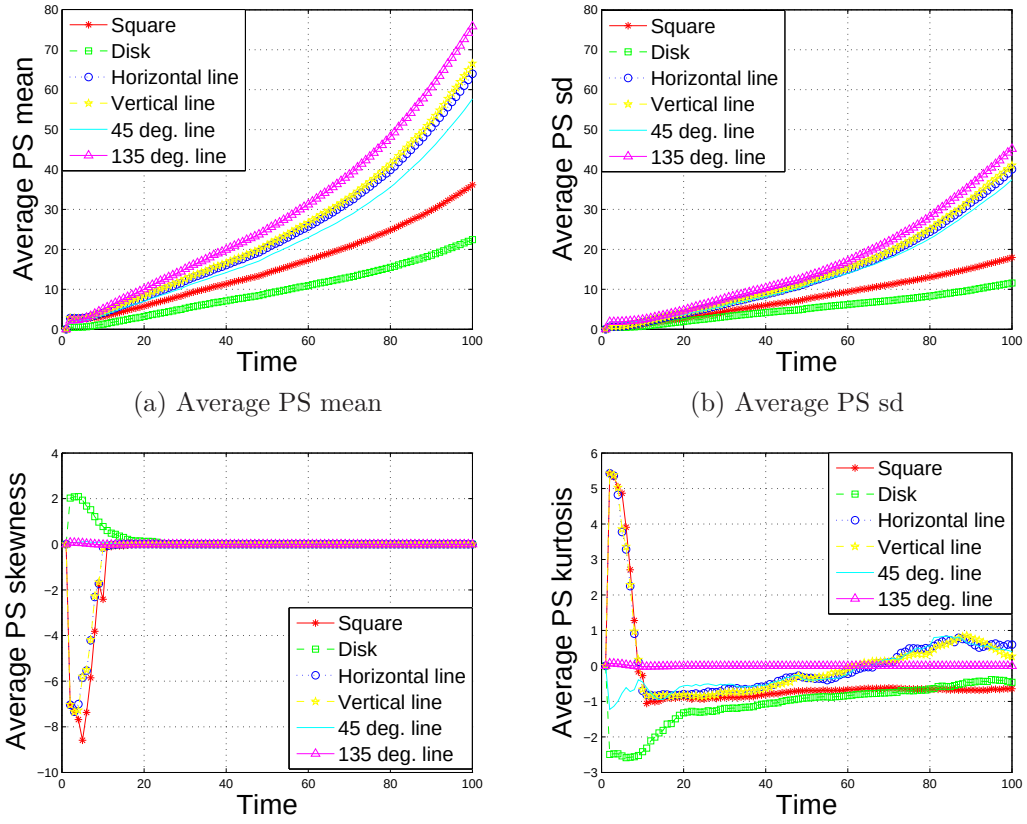
(d) Average PS kurtosis

Figure 4.13: Plots of the average foreground PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 100^2 ellipse images at each time point.

horizontal and vertical line SEs are the same, and the average PS means for lines at 45° and 135° are the same. Average PS means for a disk SE are lower than for any other SEs through the whole evolution period for both sets of images.

Average PS sd for any line SE increases with time, although the rate of increase is higher for a horizontal or a vertical line SE. A curved relationship with time is observed for the PS sd for a square and a disk SE, especially for the square SE. The PS sds for horizontal and vertical line SEs are linearly related to time. For the 45° and 135° line SEs, the PS sd is almost constant at a level around 34 up to $t = 26$ and moves upwards slowly after that. Larger negative skewness is observed for the horizontal and vertical line SEs at the beginning of the evolution but for the other SEs skewness is almost zero for the entire evolution period. All SEs produced near-zero kurtosis, except for the horizontal and vertical lines which produced very high kurtosis at the beginning of the evolution time.

Figure 4.16 represents the corresponding results for the background of the 256^2 pyramid images. It shows almost identical relationships as in Figure 4.15.



(a) Average PS mean

(b) Average PS sd

(c) Average PS skewness

(d) Average PS kurtosis

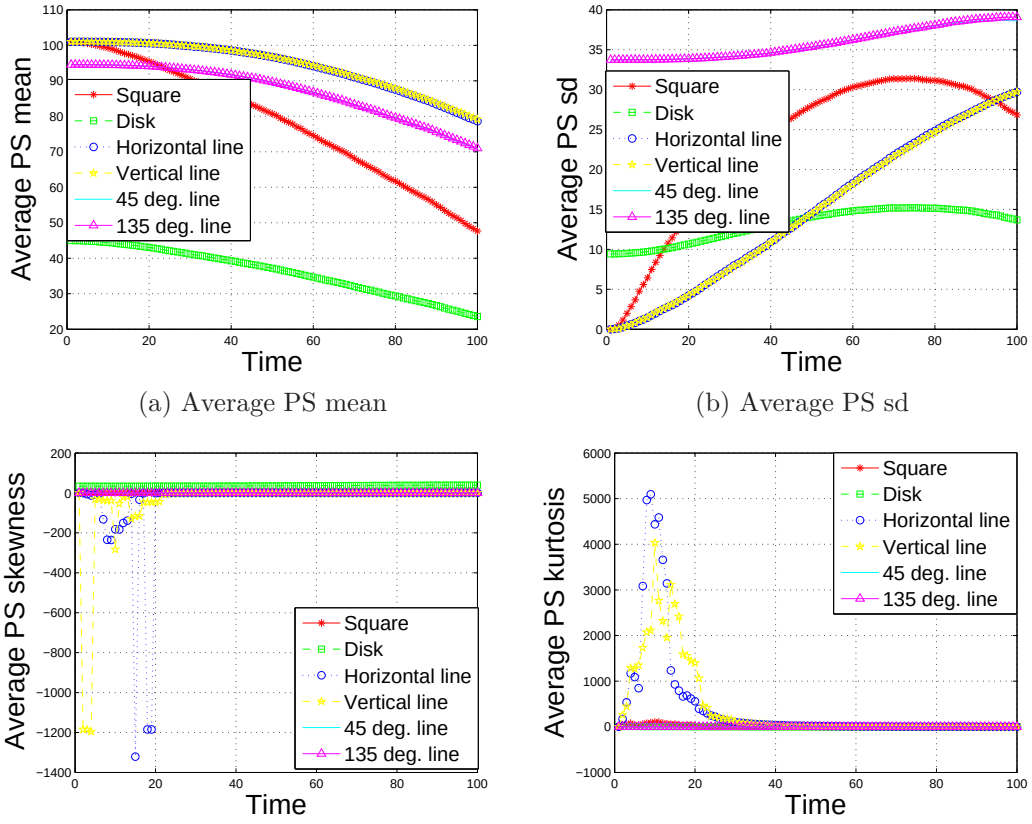
Figure 4.14: Plots of the average foreground PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 256^2 ellipse images at each time point.

Both average PS skewness and kurtosis do not show a clear relationship with time, hence do not provide useful texture information for classification.

Similarly granulometry was used on the background of the ellipse images of size 100^2 and 256^2. Average PS moments are plotted against evolution time in Figures 4.17 and 4.18. A clearly decreasing time trend is observed for the average PS background mean for all SEs, whereas a clear curvilinear relation is found for average PS sd with time. Again, both PS skewness and kurtosis are near-zero after evolution time 20, so we do not use these in the prediction process.

4.3.3 Nature of PS moments

To examine the variation in the PS moments, the average foreground PS moments, their maximum, minimum and 95% confidence limits for the 256^2 pyramid images are plotted in Figure 4.19, for the six different SEs. The confidence intervals are very narrow and close to the average moments over the whole evolution time,



(a) Average PS mean

(b) Average PS sd

(c) Average PS skewness

(d) Average PS kurtosis

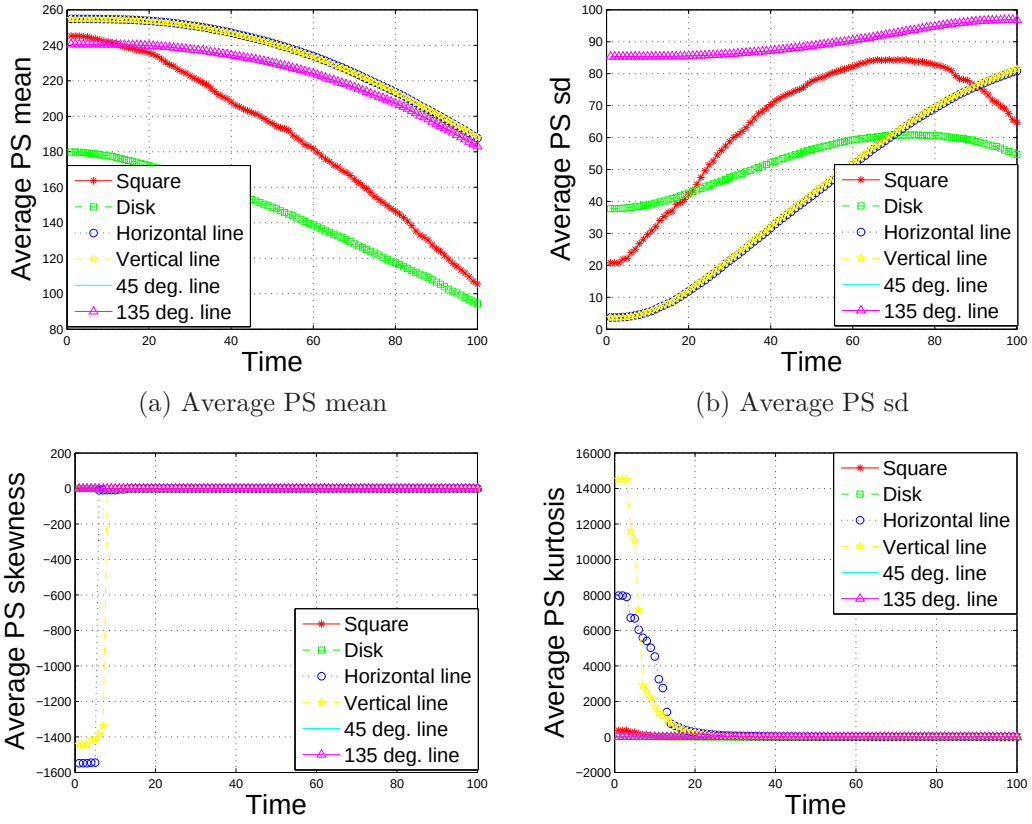
Figure 4.15: Plots of the average background PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 100^2 pyramid images at each time point.

while the range of the PS moments is wider. A similar situation is observed in the corresponding figures for the ellipse images, shown in Figure 4.21.

Corresponding results for the background PS moments from the 256^2 pyramid and ellipse images are shown in Figures 4.20 and 4.22. The confidence limits are again very close to the average moments but the moments at a specific time point vary considerably. Wider variability of the PS moments may adversely affect the classification results and may explain why the classification results are not as good as we would expect (see Chapter 5).

4.3.4 Principal component analysis of the PS moments

As we have seen above that only the PS means and sds provide useful information regarding evolution time, not skewness and kurtosis, we now use only the first two granulometric moments to relate to evolution time. So our interest lies in combining the information from the first two granulometric moments obtained



(a) Average PS mean

(b) Average PS sd

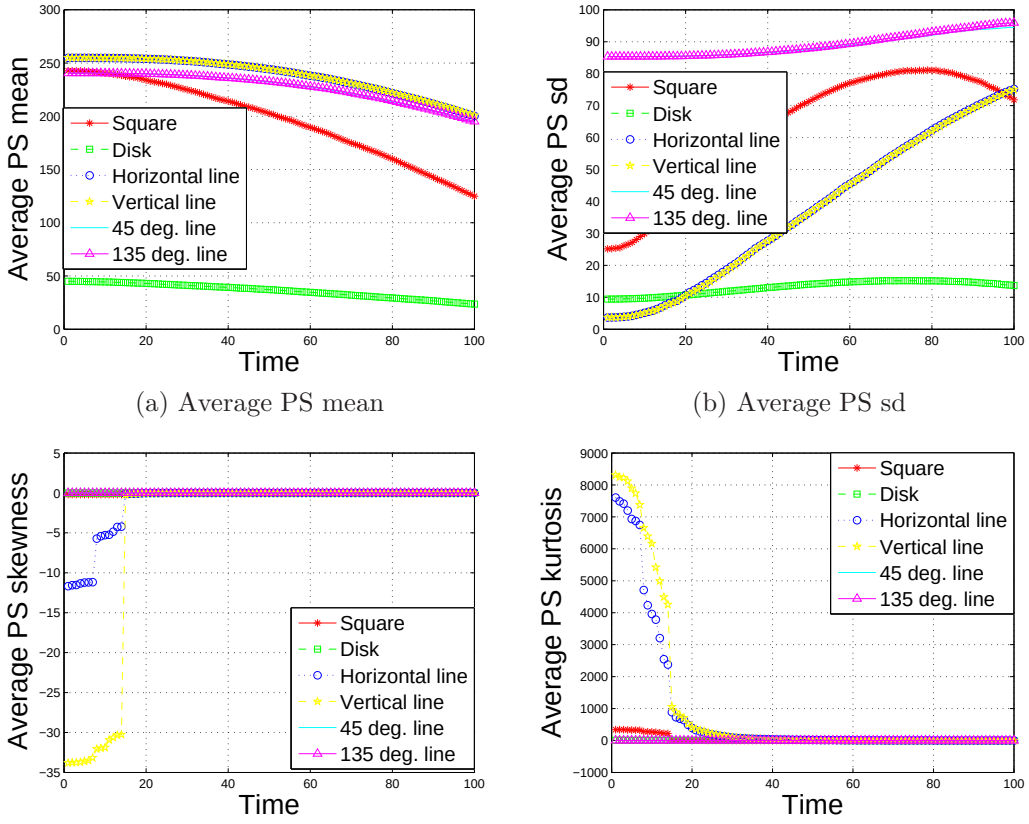
(c) Average PS skewness

(d) Average PS kurtosis

Figure 4.16: Plots of the average background PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 256^2 pyramid images at each time point.

using the six SEs. Therefore the average PS moment data become a 100×12 matrix, where rows correspond to time, the first six columns represent the PS means and the last six represent the PS sds for each SE. Having such a large feature set, we consider principal component analysis (PCA), which can identify a small number of independent linear combinations (principal components) of the set of feature variables that retain a high proportion of the information in the original variables, and also show which, if any, are the more important variables in the data.

PCA was applied to the moments of the 256^2 pyramid images and ellipse images. For the pyramid images, the first principal component (PC) explains 99.75% of the variation in the foreground moments, together with the second PC it explains 99.98% and the first three PCs explain 99.99%. The magnitude of the moments is larger for a line SE at any of the four angles, leading to larger coefficients for those SEs. So the moments were also normalised before applying PCA. In this case, the first PC explains 99.45% of the variation, the



(a) Average PS mean

(b) Average PS sd

(c) Average PS skewness

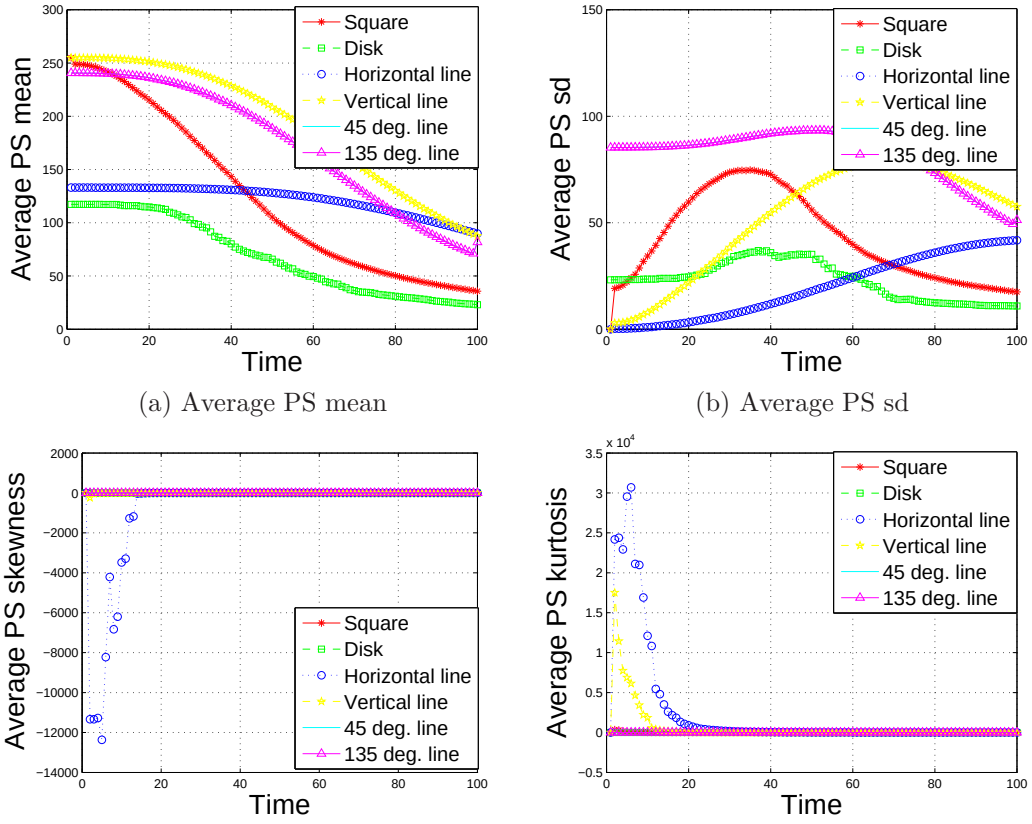
(d) Average PS kurtosis

Figure 4.17: Plots of the average background PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 100^2 ellipse images at each time point.

first two explain 99.98% and the first three explain 99.99%. The coefficients of the first 3 PCs for both foreground and background PS moments (with and without normalisation) are shown in Table 4.1.

For the foreground moments with normalisation, the coefficients are almost identical in the first PC. In PC2, the PS sd from a line at 135° has the highest coefficient (0.532) and the second strongest corresponds to the PS sds from the disk (-0.454) and line at 45° (0.455). In PC3, the PS sd from a line at 0° has the strongest coefficient (-0.531). However using PCA without normalisation, in the first PC the PS means from any of the line SEs have the higher coefficients. The PS sd from a line at 135° has the highest coefficient (0.588) in PC2. The PS sd from a line at 45° has the highest coefficient (0.538) in PC3.

For the background moments, the first 3 PCs explain 95.58% of the variation when PCA was applied with normalisation, but without normalisation the first 3 PCs express 97.04% of the variation in the moments. The PS mean from a line at 135° has the strongest coefficient (-0.311) in PC1, whereas the PS mean from



(a) Average PS mean

(b) Average PS sd

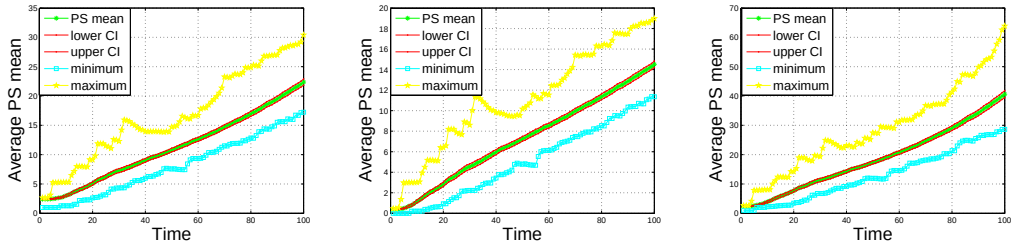
(c) Average PS skewness

(d) Average PS kurtosis

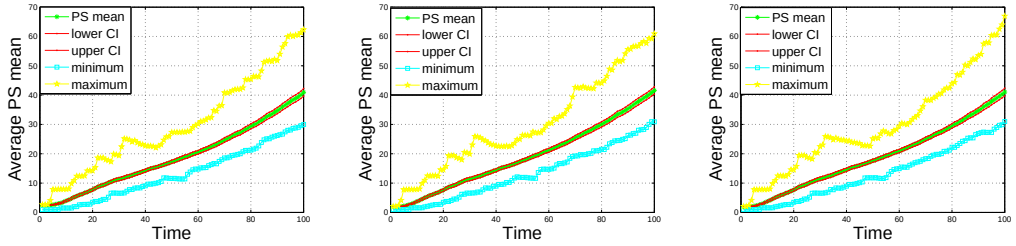
Figure 4.18: Plots of the average background PS moments (averaged over 100 simulations) against evolution time using six different SEs, for the 256^2 ellipse images at each time point.

a disk SE and the PS sds from a 90° and 135° line have the strongest coefficients in PC2 and PC3 respectively, using PCA on the normalised moments. Without normalisation, the PS mean from a square SE has the highest coefficient both in PC1 (0.580) and PC3 (0.792), while the PS mean from a disk SE has the strongest coefficient (-0.830) in PC2.

For the ellipse images, the first 3 PCs explain 98.08% of the variation in the foreground PS moments. The principal components scores of the first 3 PCs with and without normalisation are shown in Table 4.1. The PS mean from a square SE and sd from a horizontal line SE have the strongest coefficient (0.293), when PCA was applied with normalisation, although all the coefficients are similar. In PC2, the PS sd from a disk SE has the strongest coefficient (-0.504), while in PC3 the PS mean from a line at 135° has the strongest coefficient (-0.574). The first 3 PCs without normalisation explain 98.94% variation of the moments. In PC1, a 45° line has the highest coefficient of 0.458 for the PS sd. In PC2 and PC3, the PS mean and sd from a 135° line SE have the strongest coefficients of

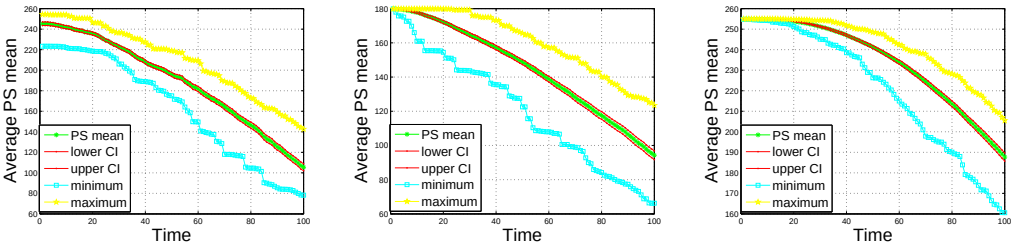


(a) Square SE (b) Disk SE (c) Horizontal line SE

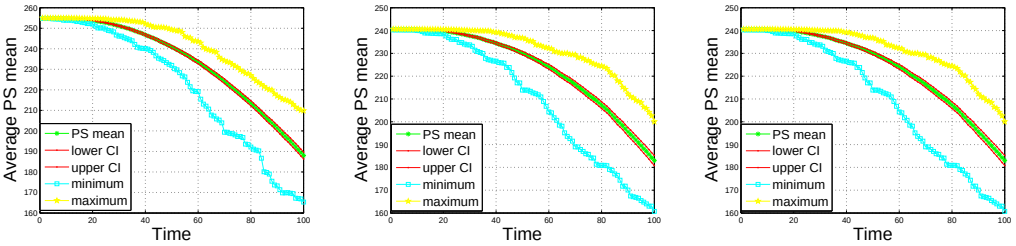


(d) Vertical line SE (e) Line at 45° SE (f) Line at 135° SE

Figure 4.19: Plots of average foreground PS means, maximum and minimum and 95% confidence intervals versus time, for the 256^2 pyramid images.



(a) Square SE (b) Disk SE (c) Horizontal line SE

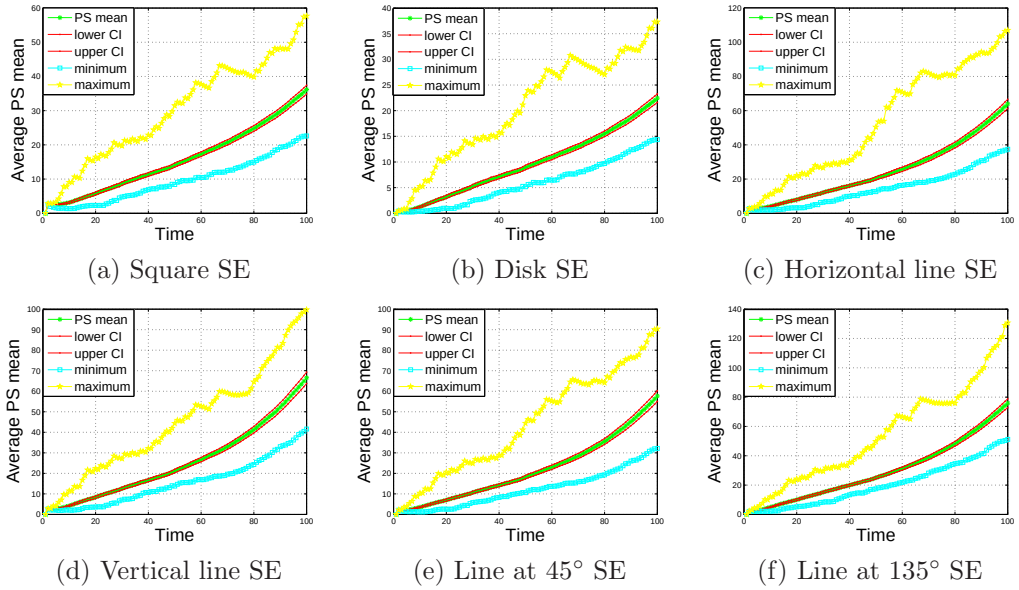


(d) Vertical line SE (e) Line at 45° SE (f) Line at 135° SE

Figure 4.20: Plots of average background PS means, maximum and minimum and 95% confidence intervals versus time, for the 256^2 pyramid images.

0.627 and -0.642 respectively.

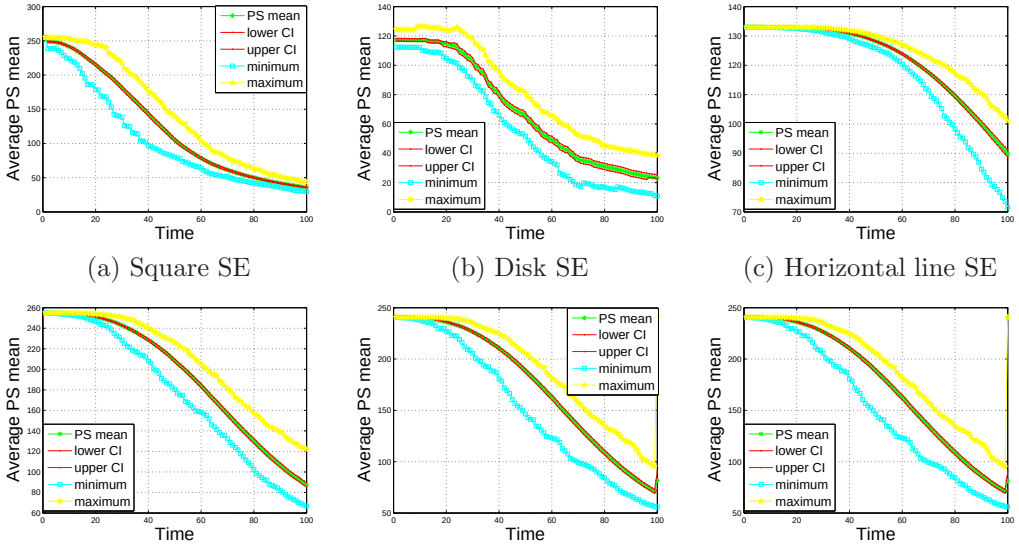
Using PCA on the background PS moments from the 256^2 ellipse images, the first 3 PCs explain 97.40% and 99.13% of the variation with and without normalisation respectively. The coefficients for both cases are shown in Table 4.1. With normalisation, the PS sd from a 0° and 45° line SE produced the highest coefficient of 0.328. The PS sd and mean from a disk SE produced the strongest



(a) Square SE (b) Disk SE (c) Horizontal line SE

(d) Vertical line SE (e) Line at 45° SE (f) Line at 135° SE

Figure 4.21: Plots of average foreground PS means, maximum and minimum and 95% confidence intervals versus time, for the 256^2 ellipse images.



(a) Square SE (b) Disk SE (c) Horizontal line SE

(d) Vertical line SE (e) Line at 45° SE (f) Line at 135° SE

Figure 4.22: Plots of average background PS means, maximum and minimum and 95% confidence intervals versus time, for the 256^2 ellipse images.

coefficients (-0.517 and -0.694) in PC2 and PC3 respectively. Without normalisation the PS mean from a square SE has the highest coefficients (0.550 and 0.540) in PC1 and PC2. In the third PC, the PS mean from a disk SE has the highest coefficient (0.847).

From this analysis, the line SEs are more informative than the square and disk SEs for the foreground moments of the pyramid images, while for the background

images the square, disk and line SEs are the most informative in different PCs. For the ellipse images, in some PCs the square SE is the most informative while in other cases the disk or line SEs appeared to be best. In general no single SE is better overall than the others for all sets of moments. Therefore, we conclude that choosing a suitable SE largely depends on the images and objects in the images of interest.

4.4 Conclusion

In this chapter we have generated synthetic images of differently sized objects and computed granulometric PS moments from them using multiple SEs. We investigated possible relationships between the average PS moments and time, and found that both the foreground and background PS mean and sd always show a relationship with time for both the pyramid and ellipse images. We also computed the PCs of the PS moments. In the next chapter we will model the PS means and sds in terms of evolution time, and we also examine the usefulness of the PCs to relate them to evolution time.

Table 4.1: Principal component scores for average foreground and background PS mean and sd using 6 SEs for the size 256^2 images.

PS moments from the pyramid images							
	Foreground Moments	Coefficients (normalisation)			Coefficients (without normalisation)		
	SE	PC1	PC2	PC3	PC1	PC2	PC3
PS means	square	0.289	-0.158	0.371	0.224	-0.210	0.366
	disk	0.288	-0.393	0.205	0.163	-0.351	0.146
	line at 0°	0.290	0.008	-0.145	0.411	-0.033	-0.198
	line at 90°	0.290	-0.057	-0.028	0.401	-0.167	0.002
	line at 45°	0.289	-0.074	0.083	**0.422**	-0.215	0.202
	line at 135°	0.290	-0.026	-0.083	0.418	-0.106	-0.091
PS sds	square	0.289	-0.195	0.117	0.112	-0.126	0.066
	disk	0.288	-0.454	-0.145	0.072	-0.178	-0.042
	line at 0°	0.289	0.255	**-0.531**	0.258	0.310	-0.519
	line at 90°	0.289	0.112	-0.449	0.241	0.111	-0.405
	line at 45°	0.288	0.455	0.505	0.221	0.494	**0.538**
	line at 135°	0.287	**0.532**	0.104	0.222	**0.588**	0.164
	Background Moments	Coefficients (normalisation)			Coefficients (without normalisation)		
PS means	square	0.305	-0.040	0.111	**0.580**	-0.114	**0.792**
	disk	-0.220	**-0.666**	-0.155	-0.203	**-0.830**	0.064
	line at 0°	0.297	-0.044	0.167	0.349	-0.021	-0.094
	line at 90°	-0.237	-0.560	0.088	-0.090	-0.221	-0.002
	line at 45°	0.304	-0.209	0.098	0.272	-0.222	-0.273
	line at 135°	**-0.311**	-0.099	-0.087	-0.343	-0.152	0.239
PS sds	square	0.305	-0.190	0.083	0.270	-0.201	-0.261
	disk	-0.310	-0.105	-0.083	-0.343	-0.161	0.245
	line at 0°	0.296	-0.251	0.250	0.227	-0.234	-0.222
	line at 90°	-0.284	0.082	**0.621**	-0.055	0.014	0.021
	line at 45°	0.296	-0.251	0.250	0.227	-0.234	-0.222
	line at 135°	-0.284	0.082	**0.621**	-0.055	0.014	0.021
PS moments from the ellipse images							
	Foreground Moments	Coefficients (normalisation)			Coefficients (without normalisation)		
	SE	PC1	PC2	PC3	PC1	PC2	PC3
PS means	square	**0.293**	0.065	0.151	0.219	-0.017	0.083
	disk	0.290	0.311	0.255	0.115	0.031	0.084
	line at 0°	0.292	0.137	0.185	0.141	0.002	0.046
	line at 90°	0.286	0.407	0.284	0.074	0.031	0.049
	line at 45°	0.292	0.145	-0.247	0.391	0.411	0.105
	line at 135°	0.281	0.388	**-0.574**	0.254	**0.627**	0.160
PS sds	square	0.292	-0.309	0.174	0.401	-0.440	0.127
	disk	0.284	**-0.504**	0.288	0.258	-0.490	0.144
	line at 0°	**0.293**	-0.036	0.081	0.353	-0.043	0.313
	line at 90°	0.286	-0.014	0.082	0.236	-0.010	0.384
	line at 45°	0.291	-0.209	-0.219	**0.458**	-0.011	-0.500
	line at 135°	0.282	-0.385	-0.481	0.280	0.012	**-0.642**
	Background Moments	Coefficients (normalisation)			Coefficients (without normalisation)		
	SE	PC1	PC2	PC3	PC1	PC2	PC3
PS means	square	0.297	0.314	0.029	**0.550**	**0.540**	0.074
	disk	0.216	-0.327	**-0.694**	0.091	-0.378	**0.847**
	line at 0°	0.308	0.252	-0.046	0.262	0.150	0.151
	line at 90°	0.267	-0.286	-0.400	0.046	-0.142	0.127
	line at 45°	0.316	-0.101	0.092	0.091	-0.129	-0.084
	line at 135°	-0.321	-0.150	0.006	-0.109	0.002	-0.005
PS sds	square	0.326	0.043	-0.003	0.421	-0.251	-0.018
	disk	-0.229	**-0.517**	-0.022	-0.160	-0.423	-0.064
	line at 0°	**0.328**	0.049	0.039	0.439	-0.238	-0.133
	line at 90°	0.248	-0.414	0.415	0.067	-0.274	-0.310
	line at 45°	**0.328**	0.049	0.039	0.439	-0.238	-0.133
	line at 135°	0.248	-0.414	0.415	0.067	-0.274	-0.310

Chapter 5

Classification using Granulometries

In this chapter the granulometric moments calculated in Section 4.3 are used to model texture evolution time of the synthetic images, using a new regression approach which is developed in Section 5.3.

5.1 Modelling PS Moments

Since the parameters used to generate the synthetic texture images, i.e. the probability of adding a new object (α), the probability of updating an existing object (δ) and the rate at which the objects grow (γ) are all fixed over the period of time for which the process was observed for any one simulation, the PS moments are simply modelled as a function of evolution time rather than of the parameters.

At each time step, $t = 1, 2, \ldots, 100$, of the evolution process there are 100 images, as 100 simulations are used to calculate the sample average moments (averaged over the 100 simulations). We observed the relationships of the average PS moments, namely, mean, sd, skewness, and kurtosis for both the foreground and background of the pyramid and ellipse images, in Chapter 4. As skewness and kurtosis do not exhibit any clear relationship with time (see Section 4.3) we will not use them in model building, but use the foreground and background mean and sd for each of the different SEs. We model the foreground moments separately from the background ones.

Individual moments obtained from different SEs are expressed as a function of evolution time using regression modelling. The objective of regression analysis is to find a deterministic model which allows prediction of the values of a dependent variable (measured subject to error) from values of one or more independent

variables (assumed to be known and not subject to error). The general process of fitting data to a linear combination of variables is termed linear regression.

Let $Y_i(t)$, $i = 1, 2, \ldots, 6$ denote the average PS moment obtained by using the SE given by a square, disk, horizontal line, vertical line, or line at 45° and 135° respectively at time t. The simple linear regression model of $Y_i(t)$ on time t is

$$Y_i(t) = \beta_{i0} + \beta_{i1} * t + \xi_i, \tag{5.1}$$

where $t = 1, 2, \ldots, T$ represents evolution time, and ξ_i, $i = 1, \ldots, 6$, is an error term.

The quadratic regression is of the form:

$$Y_i(t) = \beta_{i0} + \beta_{i1} * t + \beta_{i2} * t^2 + \xi_i. \tag{5.2}$$

The cubic regression can be written as:

$$Y_i(t) = \beta_{i0} + \beta_{i1} * t + \beta_{i2} * t^2 + \beta_{i3} * t^3 + \xi_i. \tag{5.3}$$

Simple linear regression produces a straight line fit of the data and quadratic regression produces a fitted parabola, whereas cubic regression produces a fitted S shaped curve. Least squares is used to fit every model.

Assumptions about the error term: Some fundamental assumption are imposed on the disturbance term ξ_i, known as the Gauss-Markov assumptions, that are sufficient to guarantee that ordinary regression estimates will have good properties (Rawlings (1932)), namely.

- The errors ξ_i are independently and identically distributed random variables having an expected value of zero, i.e. $E(\xi_i) = 0$. This means that on average the errors balance out. This also implies that the disturbances associated with different observations are uncorrelated, i.e. $E(\xi_i, \xi_j) = 0$ if $i \neq j$.

- The disturbances ξ_i are homoscedastic: $E(\xi_i^2) = \sigma^2$, i.e. the variance of the disturbance is the same for each observation.

Serial correlation: When the observations are collected in successive periods of time, the disturbances associated with different observations may be correlated and the disturbance terms become autocorrelated. This property is known as serial correlation. With first-order serial correlation, errors in one time period are correlated directly with errors in the previous time period, and with *positive* or *negative* serial correlation, errors in one time period are positively correlated or negatively correlated with errors in the next time period respectively.

Durbin-Watson test: The most popular test for serial correlation is the Durbin-Watson test (Gardiner (2001)). The Durbin-Watson statistic is:

$$DW = \frac{\sum_{t=2}^{T}(\hat{\xi}_t - \hat{\xi}_{t-1})^2}{\sum_{t=1}^{T}\hat{\xi}_t^2}, \tag{5.4}$$

where $t = 1, 2, \ldots, T$ is the number of time periods.

The DW statistic will lie in the 0-4 range, with a value near 2 indicating no first-order serial correlation. When successive values of ξ_t are close to each other, the DW statistic will be low (below 2), indicating the presence of positive serial correlation. Positive serial correlation is associated with DW values below 2 and negative serial correlation with DW values above 2. The Durbin-Watson test has the null hypothesis that the autocorrelation of the disturbances is 0. The test is significant if $DW < Dl$ or $DW > Du$ and is inconclusive when $Dl < DW < Du$, where Dl and Du are lower and upper critical values of the test statistic at the specified significance level.

Some measures of goodness of fit of these models are:

Mean squared error: The mean squared error (mse) is a measure of prediction error, i.e. mse is the average squared vertical distance of a data point from a fitted curve or prediction. The smaller the mse, the closer the fit is to the data. The root mean square error (rmse) is the square root of mse, which is directly interpretable as it has the same units as the data (Gardiner (2001)).

Coefficient of determination: The coefficient of determination R^2 is an important measure of goodness of fit of a regression model. It is defined as the ratio of the sum of squares due to regression to the total sum of squares, $R^2 = SS_{reg}/SS_{tot}$. The higher the R^2, the better the model fit to the data used to build it. For example if $R^2 = 0.9968$, the fitted model expresses 99.7% of the variation in the dependent variable.

Adjusted R^2: The R^2 of a model can be made larger simply by adding more predictors in the model even if they are not useful predictors. The adjusted R^2, denoted by R^2_{adj}, allows for this and may actually decrease because the decrease in summed square error (SSE) may be more than offset by the corresponding decrease in the error degrees of freedom (df). The adjusted R^2 is $R^2_a = 1 - \frac{SSE/df_E}{SST/df_T}$, where SST is the total sum of squares (Rawlings (1932)).

5.2 Modelling Foreground PS Moments

We investigated the presence of serial correlation in all fitted models using different SEs for both the pyramid and ellipse images with different parameter settings. Similar results were observed for both pyramid and ellipse images. Here we have shown only the results for the average foreground PS moments (mean and sd) for the pyramid image of size 256^2 for all SEs using fitted straight line, quadratic and cubic regression models.

Using parameters $\alpha = 0.5$, $\delta = 0.3$, and $\gamma =$ discrete uniform $[1,\ 2]$ initially, 100 stacks of 256^2 pyramid images, each having 100 layers (one layer for each time point), were generated and the average foreground PS means and sds were calculated using each of the six SEs. The simple least squares regression given by equation (5.1) was fitted to each of the 6 sets of PS means and sds.

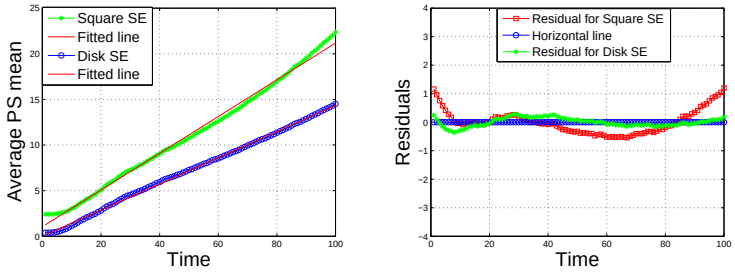
5.2.1 Modelling foreground PS mean

Figure 5.1(a) shows average foreground PS means for a square and a disk SE plotted against time, with their fitted regression line. The models fit very well for both sets of means. The upper lines in the graph correspond to the average PS mean for a square and the respective fitted line, whereas the lower lines represent the average PS mean and fitted line for a disk SE. Figure 5.1(b) shows the residuals for both models in (a). There is clearly curvature present in the residual plots. We employed the Durbin-Watson test on every fitted model to test if there is serial correlation present.

Figure 5.1(c) shows the average PS moments for all line SEs with their fitted regression lines and their corresponding residuals are shown in (d). Again all of the models fit well, but curvature in the residual plots suggests that quadratic regression may be more appropriate. The pattern of runs of positive then negative residuals also suggests serial correlation. This is not surprising, as the data were collected as the images evolve through time.

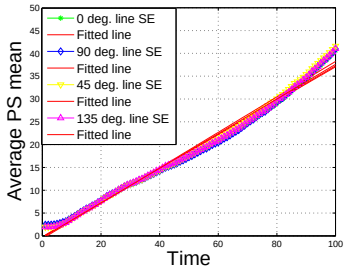
The quadratic regression model (equation (5.2)) was fitted also, with results shown in Figures 5.2. Quadratic regressions fit the data better than the straight line regressions. However the residual plots still exhibit curvature in most cases. The curvature in the residual plots suggests that applying higher order polynomial would be more appropriate and any problem caused by the presence of serial correlation may reduce.

Figure 5.3 represents the results of cubic regression (equation (5.3)). All models fit the corresponding data extremely well and the presence of any serial

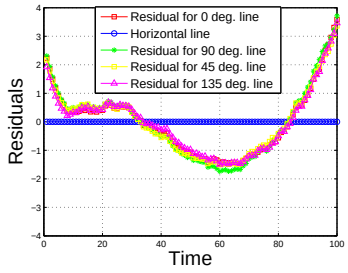


(a) Mean for square and disk SE

(b) Residual plot of (a)

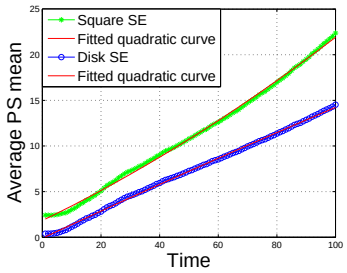


(c) Mean for all line SEs

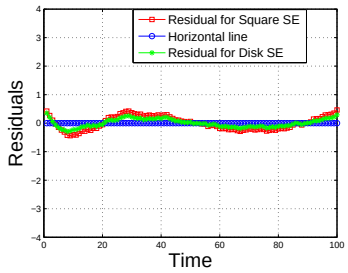


(d) Residual plot of (c)

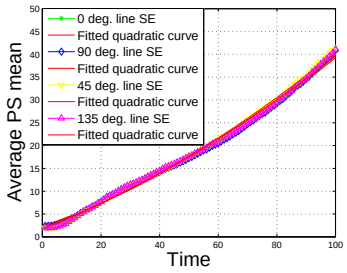
Figure 5.1: Plots of average foreground PS mean versus time t, with fitted linear regression lines (solid lines) and residual plots for the 256^2 pyramid images generated using parameters $\alpha = 0.5$, $\delta = 0.3$ and $\gamma =$ discrete uniform $[1,\ 2]$.



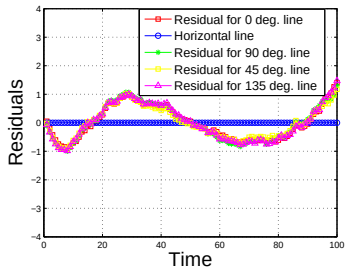
(a) Mean for square and disk SE



(b) Residual plot of (a)

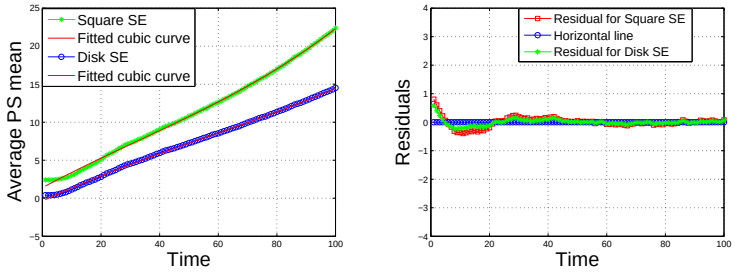


(c) Mean for all line SEs



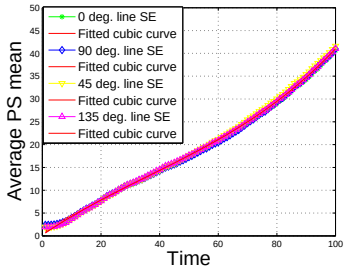
(d) Residual plot of (c)

Figure 5.2: Plots of average foreground PS mean versus time t, with fitted quadratic regression curves (solid lines) and residual plots for the 256^2 pyramid images generated using parameters $\alpha = 0.5$, $\delta = 0.3$ and $\gamma =$ discrete uniform $[1,\ 2]$.

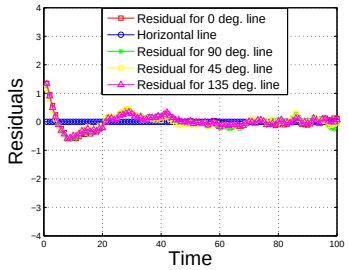


(a) Mean for square and disk SE

(b) Residual plot of (a)



(c) Mean for all line SEs



(d) Residual plot of (c)

Figure 5.3: Plots of average foreground PS mean versus time t, with fitted cubic regression curves (solid lines) and residual plots for the 256^2 pyramid images generated using parameters $\alpha = 0.5$, $\delta = 0.3$ and $\gamma =$ discrete uniform $[1,\ 2]$.

correlation is reduced, especially for square and disk SEs, but serial correlation is noticeable in the models fitted using moments from a line SE. We applied the Durbin-Watson test to all six PS mean models from the pyramid images and computed test statistics and the associated p-values (Table 5.1). The results indicate that positive serial correlation is present in every model.

The highest Durbin-Watson test statistic among the models is 0.1793 with p-value $3.7319e^{-20}$ for the cubic regression model using the PS mean from a 45° line SE, but all are highly significant. The presence of positive serial correlation is unavoidable as the synthetic image at any time point is built directly from the image at the previous time point. Serial correlation will not affect the unbiasedness or consistency of ordinary least square estimators, but it does affect their efficiency (Everitt and Dunn (2001)).

5.2.2 Modelling foreground PS standard deviation

Figure 5.4 represents the six sets of average PS sds with the fitted linear regression lines and associated residual plots. As for the PS means, straight line regression models fit the data quite well, but the residual plots indicate curvature and presence of serial correlation, so we also fitted quadratic regression models.

The results from quadratic regression are shown in Figure 5.5. These models

Table 5.1: Durbin-Watson test statistics and associated p-values for the six foreground PS mean models using the 256^2 pyramid images, one for each SE.

	Straight line		Quadratic		Cubic	
Model	DW	p-value	DW	p-value	DW	p-value
Mean from square SE	0.03	2.15e-23	0.06	9.10e-23	0.11	1.15e-21
Mean from disk SE	0.07	2.01e-22	0.08	3.66e-22	0.15	1.02e-20
Mean from 0° line	0.02	1.18e-23	0.03	2.59e-23	0.14	5.90e-21
Mean from 90° line	0.01	1.12e-23	0.03	2.66e-23	0.16	1.28e-20
Mean from 45° line	0.02	1.17e-23	0.04	3.36e-23	0.18	3.73e-20
Mean from 135° line	0.02	1.26e-23	0.03	2.57e-23	0.17	2.64e-20

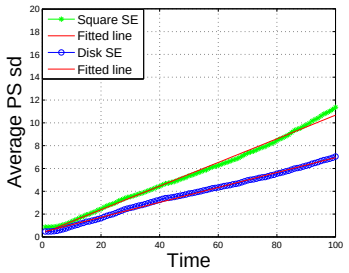
fit the data better than the straight lines and lessen the curvature in the residual plots, but serial correlation is still apparent.

Figure 5.6 represents the same sets of PS sds with fitted cubic regression curves and residual plots. All models fit the data very well, and the presence of serial correlation is not so severe, as judged from the residual plots.

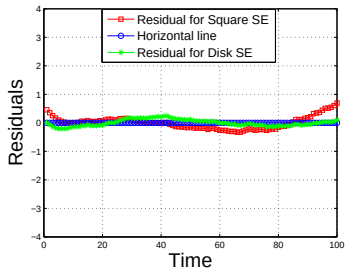
The Durbin-Watson test statistics and corresponding p-values shown in Table 5.2 confirm the presence of positive serial correlation in every model fitted to the PS sds. This is inevitable owing to the nature of the data. Time series modelling would allow for the serial correlation but would make prediction of evolution time much more difficult. We therefore continue to use regression modelling as the fit of the cubic regression model is generally very good.

Table 5.2: Durbin-Watson test statistics and associated p-values for the six foreground PS sd models using the 256^2 pyramid images, one for each SE.

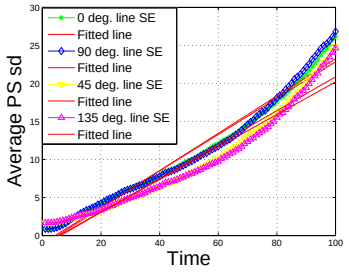
	Straight line		Quadratic		Cubic	
Model	DW	p-value	DW	p-value	DW	p-value
Sd from square SE	0.03	2.51e-23	0.05	8.59e-23	0.19	7.57e-20
Sd from disk SE	0.04	4.55e-23	0.06	1.29e-22	0.13	4.11e-21
Sd from 0° line	0.01	1.04e-23	0.03	3.03e-23	0.32	1.92e-17
Sd from 90° line	0.01	9.21e-24	0.03	2.62e-23	0.29	6.08e-18
Sd from 45° line	0.01	8.45e-24	0.057	9.64e-23	0.43	2.57e-15
Sd from 135° line	0.01	8.83e-24	0.03	2.60e-23	0.18	3.25e-20



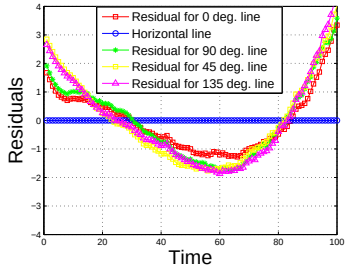
(a) Sd for square and disk SE



(b) Residual plot of (a)

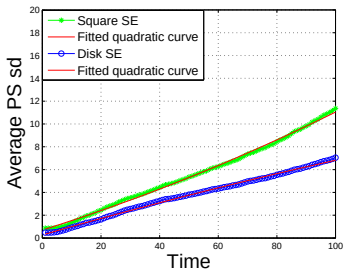


(c) Sd for all line SEs

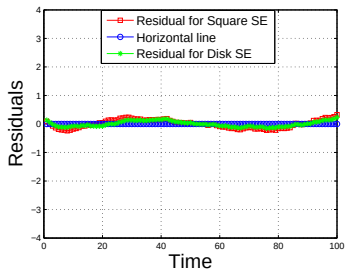


(d) Residual plot of (c)

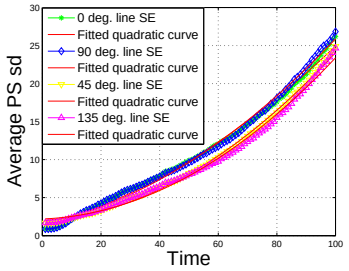
Figure 5.4: Plots of average foreground PS sd versus time t, with fitted linear regression lines (solid lines) and residual plots for the 256^2 pyramid images generated using parameters $\alpha = 0.5$, $\delta = 0.3$ and $\gamma =$ discrete uniform $[1,\ 2]$.



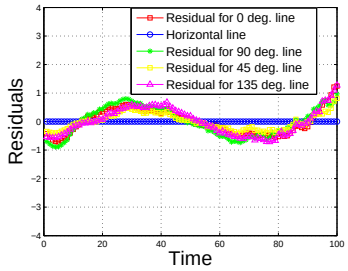
(a) Sd for square and disk SE



(b) Residual plot of (a)

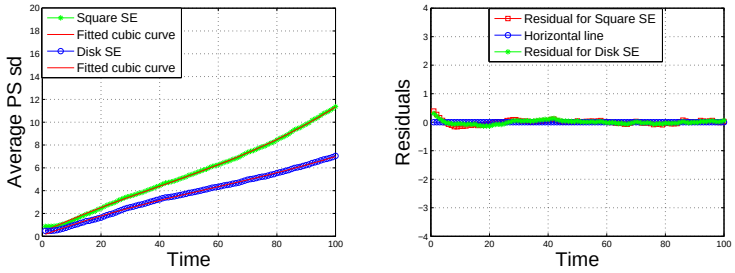


(c) Sd for all line SEs



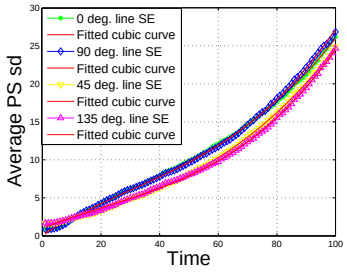
(d) Residual plot of (c)

Figure 5.5: Plots of average foreground PS sd versus time t, with fitted quadratic regression curves (solid lines) and residual plots for the 256^2 pyramid images generated using parameters $\alpha = 0.5$, $\delta = 0.3$ and $\gamma =$ discrete uniform $[1,\ 2]$.

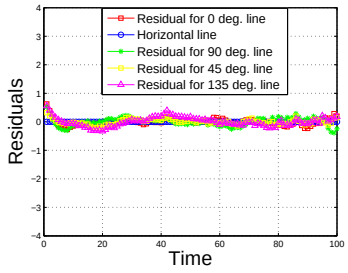


(a) Sd for square and disk SE

(b) Residual plot of (a)



(c) Sd for all line SEs



(d) Residual plot of (c)

Figure 5.6: Plots of average foreground PS sd versus time t, with fitted cubic regression curves (solid lines) and residual plots for the 256^2 pyramid images generated using parameters $\alpha = 0.5$, $\delta = 0.3$ and $\gamma =$ discrete uniform $[1,\ 2]$.

5.3 New Regression-based Classifier

Regression analysis provides a straightforward way to estimate the response variable for any known value of the regressor, but when the interest centres on estimating an unknown value of the regressor corresponding to an observed value of the response variable, this is known as calibration. So calibration is a reverse process to regression, which can be handled either by the classical approach or by the inverse approach (Krutchkoff, 1967). In the classical approach, the fitted regression line is solved for the regressor, namely, if the fitted line is

$$Y(t) = \hat{\beta}_0 + \hat{\beta}_1 * t, \tag{5.5}$$

the corresponding calibration equation becomes

$$\hat{t} = \frac{Y(t) - \hat{\beta}_0}{\hat{\beta}_1}, \tag{5.6}$$

where $t = 1, 2 \ldots, T$, and $Y(t)$ is the observed average PS moment at time point t. The inverse calibration approach suggests fitting a regression line as

$$T = \gamma_0 + \gamma_1 * Y(t), \tag{5.7}$$

so once the moments for any image are available, the corresponding time state can be estimated from the calibration equation

$$\hat{T} = \hat{\gamma}_0 + \hat{\gamma}_1 * Y(t). \tag{5.8}$$

Solutions given by equations (5.6) and (5.8) will not be the same.

As in this case t is error free and $Y(t)$ is subject to error, the relation between t and $Y(t)$ given by equation (5.8) violates one of the important assumptions of regression analysis, which states that the regressor should be free of measurement error.

5.3.1 Combined straight line model

Let $Y_i(t)$, $i = 1, 2, \ldots, p$, and $t = 1, 2, \ldots, T$ be the average of the i^{th} PS moment for the t^{th} time point. Each PS moment is modelled as a function of time t. First each moment is modelled as a function of time using straight line regression of the form:

$$\begin{aligned}
\mathbf{Y_i} &= \begin{bmatrix} Y_i(1) \\ Y_i(2) \\ \vdots \\ Y_i(100) \end{bmatrix} = \begin{bmatrix} \beta_0^{(i)} + \beta_1^{(i)} t_1 \\ \beta_0^{(i)} + \beta_1^{(i)} t_2 \\ \vdots \\ \beta_0^{(i)} + \beta_1^{(i)} t_{100} \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_{100} \end{bmatrix} \\
&= \begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ \vdots & \vdots \\ 1 & t_{100} \end{bmatrix} \begin{bmatrix} \beta_0^{(i)} \\ \beta_1^{(i)} \end{bmatrix} + \varepsilon = \mathbf{A}\beta^{(i)} + \varepsilon,
\end{aligned}$$

assuming 100 time points $t_1, t_2, \ldots, t_{100}$, have been observed for each moment and each such set of equations involves the same design matrix $\mathbf{A}$. Substituting estimates $\mathbf{Y_i}$ on the left hand side of any one such equation yields numerous different estimates of t, one from each equation if they are solved separately: $\hat{Y}_i(t) = \hat{\beta}_0^{(i)} + \hat{\beta}_1^{(i)} t$. We can either rearrange the equation to solve for t given the moment (inverse prediction) or use calibration. In either case we can simply average the predictions for t.

A different approach is to use all p fitted models involving t, one per moment, at once. For example, if the number of moments is $p = 4$, we could use the mean,

sd, skewness, and kurtosis as $\hat{\mathbf{Y}}_\mathbf{1}$, $\hat{\mathbf{Y}}_\mathbf{2}$, $\hat{\mathbf{Y}}_\mathbf{3}$, $\hat{\mathbf{Y}}_\mathbf{4}$. We have in general

$$\begin{bmatrix} \hat{Y}_1(t) \\ \hat{Y}_2(t) \\ \vdots \\ \hat{Y}_p(t) \end{bmatrix} = \begin{bmatrix} \hat{\beta}_0^{(1)} + \hat{\beta}_1^{(1)} t \\ \hat{\beta}_0^{(2)} + \hat{\beta}_1^{(2)} t \\ \vdots \\ \hat{\beta}_0^{(p)} + \hat{\beta}_1^{(p)} t \end{bmatrix} \tag{5.9}$$

or $\hat{\boldsymbol{Y}} = \hat{\boldsymbol{B}}\boldsymbol{T}$, where

$$\hat{B} = \begin{bmatrix} \hat{\beta}_0^{(1)} & \hat{\beta}_1^{(1)} \\ \hat{\beta}_0^{(2)} & \hat{\beta}_1^{(2)} \\ \vdots & \vdots \\ \hat{\beta}_0^{(p)} & \hat{\beta}_1^{(p)} \end{bmatrix}$$

is a $p \times 2$ matrix and $\boldsymbol{T}$ is a 2×1 vector $(1 \quad t)^T$.

Pre-multiplying by $\hat{\boldsymbol{B}}^T$ gives $\hat{\boldsymbol{B}}^T\hat{\boldsymbol{Y}}$=$\hat{\boldsymbol{B}}^T\hat{\boldsymbol{B}}\boldsymbol{T}$, so the left hand side is of dimensions $(2 \times p) \times (p \times 1)$ and the dimensions are $(2 \times 2) \times (2 \times 1)$ on the right. Therefore we can solve for $\boldsymbol{T}$ as $(\hat{\boldsymbol{B}}^T\hat{\boldsymbol{B}})^{-1}\hat{\boldsymbol{B}}^T\hat{\boldsymbol{Y}}$ as in the original model fitting, but the problem is that this does not guarantee that the first entry of $\boldsymbol{T}$ is 1 as it should be. Therefore instead write

$$\begin{bmatrix} \hat{Y}_1(t) \\ \hat{Y}_2(t) \\ \vdots \\ \hat{Y}_p(t) \end{bmatrix} = \begin{bmatrix} \hat{\beta}_0^{(1)} \\ \hat{\beta}_0^{(2)} \\ \vdots \\ \hat{\beta}_0^{(p)} \end{bmatrix} + \begin{bmatrix} \hat{\beta}_1^{(1)} \\ \hat{\beta}_1^{(2)} \\ \vdots \\ \hat{\beta}_1^{(p)} \end{bmatrix} \begin{bmatrix} t \end{bmatrix},$$

or $\hat{\boldsymbol{Y}} = \hat{\boldsymbol{\beta}}_\mathbf{0} + \hat{\boldsymbol{\beta}}_\mathbf{1} t$, so $(\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0}) = \hat{\boldsymbol{\beta}}_\mathbf{1} t$, in which $(\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0})$ is $p \times 1$, as is $\hat{\boldsymbol{\beta}}_\mathbf{1}$, and t is scalar, and then pre-multiply by any $1 \times p$ vector, e.g. $\hat{\boldsymbol{\beta}}_\mathbf{1}{}^T$ or $(\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0})^T$, to get t=$\hat{\boldsymbol{\beta}}_\mathbf{1}{}^T(\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0})/(\hat{\boldsymbol{\beta}}_\mathbf{1}{}^T\hat{\boldsymbol{\beta}}_\mathbf{1})$, in the first case, or t=$(\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0})^T(\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0})/((\hat{\boldsymbol{Y}}\text{-}\hat{\boldsymbol{\beta}}_\mathbf{0})^T\hat{\boldsymbol{\beta}}_\mathbf{1})$, in the second case.

We will use only the PS mean and sd, so $p = 2$ in Equation (5.9).

If p straight line models are solved separately for t, we get p solutions for t namely, $t = \frac{\hat{Y}_1(t) - \hat{\beta}_0^{(1)}}{\hat{\beta}_1^{(1)}}$, ..., $t = \frac{\hat{Y}_p(t) - \hat{\beta}_0^{(p)}}{\hat{\beta}_1^{(p)}}$. Taking the average of the p solutions we get

$$
\begin{aligned}
t &= \frac{1}{p}\sum_{i=1}^{p}\frac{\hat{Y}_i(t)-\hat{\beta}_0^{(i)}}{\hat{\beta}_1^{(i)}} \\
&= \left[\frac{\hat{Y}_1(t)-\hat{\beta}_0^{(1)}}{p\hat{\beta}_1^{(1)}}+\ldots+\frac{\hat{Y}_p(t)-\hat{\beta}_0^{(p)}}{p\hat{\beta}_1^{(p)}}\right] \\
&= \left[\frac{1}{p\hat{\beta}_1^{(1)}},\ldots,\frac{1}{p\hat{\beta}_1^{(p)}}\right]\begin{bmatrix}\hat{Y}_1(t)-\hat{\beta}_0^{(1)}\\ \vdots \\ \hat{Y}_p(t)-\hat{\beta}_0^{(p)}\end{bmatrix} \\
&= \left[\frac{1}{p\hat{\beta}_1^{(1)}},\ldots,\frac{1}{p\hat{\beta}_1^{(p)}}\right]\left[\hat{\boldsymbol{Y}}-\hat{\boldsymbol{B}}_{\mathbf{0}}\right]. \qquad (5.10)
\end{aligned}
$$

The solution from the combined model was ($\hat{\boldsymbol{Y}}$-$\hat{\boldsymbol{\beta}}_{\mathbf{0}}$)= $\hat{\boldsymbol{\beta}}_{\mathbf{1}}t$. Pre-multiplying by any $1 \times p$ vector, $\boldsymbol{A} = (a_1, \ldots, a_p)$ and simplifying to make it comparable with Equation (5.10) we get

$$
\begin{aligned}
t &= \frac{A}{A\hat{\boldsymbol{\beta}}_{\mathbf{1}}}\left[\hat{\boldsymbol{Y}}-\hat{\boldsymbol{\beta}}_{\mathbf{0}}\right] \\
t &= \left[\frac{a_1}{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(1)}},\ldots,\frac{a_p}{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(p)}}\right]\left[\hat{\boldsymbol{Y}}-\hat{\boldsymbol{\beta}}_{\mathbf{0}}\right]. \qquad (5.11)
\end{aligned}
$$

The solutions of t given by Equations (5.10) and (5.11) will coincide if and only if $\frac{a_1}{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(1)}} = \frac{1}{p\hat{\beta}_1^{(1)}}$, ..., $\frac{a_p}{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(1)}} = \frac{1}{p\hat{\beta}_1^{(p)}}$, that is if $a_1 = \frac{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(1)}}{p\hat{\beta}_1^{(1)}}$, ..., $a_p = \frac{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(p)}}{p\hat{\beta}_1^{(p)}}$. Therefore only using $\boldsymbol{A} = [\frac{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(1)}}{p\hat{\beta}_1^{(1)}},\ldots,\frac{\sum_{i=1}^{p}a_i\hat{\beta}_1^{(p)}}{p\hat{\beta}_1^{(p)}}]$ will yield an estimate of t which is equivalent to the average of the p estimates of t from the p separate models.

5.3.2 Combined quadratic model

As we have observed in Section 5.2 that a higher order polynomial fits the data better, if we use quadratic polynomial regression to relate the mean and sd (for a single SE) to time, the above becomes

$$
\begin{bmatrix}\hat{Y}_1(t)\\ \hat{Y}_2(t)\end{bmatrix} = \begin{bmatrix}\hat{\beta}_0^{(1)}+\hat{\beta}_1^{(1)}t+\hat{\beta}_2^{(1)}t^2\\ \hat{\beta}_0^{(2)}+\hat{\beta}_1^{(2)}t+\hat{\beta}_2^{(2)}t^2\end{bmatrix} \qquad (5.12)
$$

or $\quad \hat{\boldsymbol{Y}} = \hat{\boldsymbol{B}}\boldsymbol{T}$, where

$$\hat{\boldsymbol{B}} = \begin{bmatrix} \hat{\beta}_0^{(1)} & \hat{\beta}_1^{(1)} & \hat{\beta}_2^{(1)} \\ \hat{\beta}_0^{(2)} & \hat{\beta}_1^{(2)} & \hat{\beta}_2^{(2)} \end{bmatrix}$$

is a 2×3 matrix and $\boldsymbol{T}$ is a 3×1 vector $(1, \quad t, \quad t^2)$.

Equation (5.12) can be rewritten as

$$\begin{bmatrix} \hat{Y}_1(t) \\ \hat{Y}_2(t) \end{bmatrix} = \begin{bmatrix} \hat{\beta}_0^{(1)} \\ \hat{\beta}_0^{(2)} \end{bmatrix} + \begin{bmatrix} \hat{\beta}_1^{(1)} & \hat{\beta}_2^{(1)} \\ \hat{\beta}_1^{(2)} & \hat{\beta}_2^{(2)} \end{bmatrix} \begin{bmatrix} t \\ t^2 \end{bmatrix}$$

or

$$\begin{bmatrix} \hat{Y}_1(t) - \hat{\beta}_0^{(1)} \\ \hat{Y}_2(t) - \hat{\beta}_0^{(2)} \end{bmatrix} = \begin{bmatrix} \hat{\beta}_1^{(1)} & \hat{\beta}_2^{(1)} \\ \hat{\beta}_1^{(2)} & \hat{\beta}_2^{(2)} \end{bmatrix} \begin{bmatrix} t \\ t^2 \end{bmatrix},$$

$$\text{or} \quad \left[\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}} \right] = \hat{\boldsymbol{\beta}}\boldsymbol{T}.$$

The left hand side is of dimension 2×1 and the right hand side is of order $(2 \times 2) \times (2 \times 1) = (2 \times 1)$.

Using the PS mean and sd obtained using 6 different SEs, the left hand side becomes 12×1 and the right hand side becomes $(12 \times 2) \times (2 \times 1)$. Pre-multiplying both sides of the above expression by $(\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})^T$, which is of order 1×12 produces

$$\begin{aligned} (\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})^T_{(1\times 12)} (\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})_{(12\times 1)} &= (\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})^T_{(1\times 12)} \hat{\boldsymbol{B}}_{(12\times 2)} \boldsymbol{T}_{(2\times 1)} \\ \text{or} \qquad K' &= \boldsymbol{K}_{(1\times 2)} \boldsymbol{T}_{(2\times 1)}. \end{aligned} \tag{5.13}$$

The left hand side is a scalar and the right hand side is $(1 \times 2) \times (2 \times 1)$, so we end up with a quadratic equation of the form $K(2)t^2 + K(1)t - K' = 0$ to solve for t. The root of such an equation can be calculated using the conventional formula

$$t = \frac{-K(1) \pm \sqrt{(K(1)^2 + 4K'K(2))}}{2K(2)}.$$

Since our goal is to predict time, we choose the positive root. If both roots are positive we choose the smallest one and in the case of no positive root we predict time as the first time point.

This model can be built using only the foreground average PS mean and sd obtained by using each of the 6 SEs, and for the background average PS mean

and sd a similar model can be built to predict the time state, or we could use both at once.

5.3.3 Combined cubic model

In a similar way a cubic regression model can be developed to relate PS moments to time. A simplified form of this relationship is as follows:

$$\left[\ \hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}}\ \right] = \left[\ \hat{\boldsymbol{\beta}}_{\mathbf{1}} \quad \hat{\boldsymbol{\beta}}_{\mathbf{2}} \quad \hat{\boldsymbol{\beta}}_{\mathbf{3}}\ \right] \begin{bmatrix} t \\ t^2 \\ t^3 \end{bmatrix}$$

$$\text{or} \quad \left[\ \hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}}\ \right] = \hat{\boldsymbol{B}}\boldsymbol{T}.$$

In this case the left hand side is 12×1 and the right hand side is $(12 \times 3) \times (3 \times 1)$. Pre-multiplying by $(\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})^T$, we get

$$(\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})^T_{(1\times 12)} (\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})_{(12\times 1)} = (\hat{\boldsymbol{Y}} - \hat{\boldsymbol{\beta}}_{\mathbf{0}})^T_{(1\times 12)} \hat{\boldsymbol{B}}_{(12\times 3)} \boldsymbol{T}_{(3\times 1)}$$

$$\text{or} \qquad K' = \boldsymbol{K}_{(1\times 3)} \boldsymbol{T}_{(3\times 1)}. \tag{5.14}$$

Now equation (5.14) can be written as $K(3)t^3 + K(2)t^2 + K(1)t - K' = 0$. A real root of such a cubic equation can be found by dividing the equation by $K(3)$ and letting $t = s - K(2)/3K(3)$. The reduced equation will be of form $s^3 + As + B = 0$, known as a depressed cubic equation. One of the roots of the depressed cubic equation is $s = u - v$ where $u^3 - v^3 = B$ and $3uv = A$. This can be proved by replacing A, B and s in the equation. Once we get s, then t the real root of the original equation can be obtained. However, this procedure does not always guarantee a real root. A more robust procedure for finding a real root of a cubic equation, as given in Tuma and Walsh (1998), is described here:

- Firstly calculate $p = (3k(1)/k(3) - (k(2)/k(3))^2)/3$ and $q = (2(k(2)/k(3))^3 - 9k(2)k(1)/k(3)/k(3) + 27(-k')/k(3))/27$.
- Then calculate the discriminant D in terms of p and q as $(p/3)^3 + (q/2)^2$.
- If $D > 0$ or $D = 0$, calculate u and v as $u = (-q/2 + \sqrt{D})^{(1/3)}$ and $v = (-q/2 + \sqrt{D})^{(1/3)}$.
- Three transformed roots in these cases are $y1 = u + v$, $y2 = -(u + v)/2 + i(u - v)\sqrt{3}/2$ and $y3 = -(u + v)/2 - i(u - v)\sqrt{3}/2$.

- If $D < 0$, the transformed roots are: $y1 = 2\sqrt{(|p|/3)\cos(\varphi/3)}$, $y2 = -2\sqrt{(|p|/3)\cos((\varphi+\pi)/3)}$ and $y3 = -2\sqrt{(|p|/3)\cos((\varphi-\pi)/3)}$, where $\varphi = acos(-q/2.\sqrt{(|p^3|/27)})$.

- The real roots of the original cubic equation are calculated as $x = y - k(2)/k(3)/3$, i.e. $x1 = y1 - k(2)/k(3)/3$, $x2 = y2 - k(2)/k(3)/3$ and $x3 = y3 - k(2)/k(3)/3$.

To find the cubic roots we use either the *roots* function in Matlab or the method given in Tuma and Walsh (1998) and available as FORTRAN or Matlab code at http://www.ece.umd.edu/∼nsw/ench250/cubiceq.htm. We use the following procedure for choosing a root as the predicted time:

- If $D < 0$, there are 3 distinct real roots and we choose the smallest positive one.

  • If none of the roots are positive, the method fails to predict time and we choose the first (smallest) time point as the prediction.

- If $D = 0$, there are three real roots of which at least two are equal and we choose the smallest positive one.

- If $D > 0$, there are one real and two complex roots, and we choose the real one if it is positive, otherwise we choose the first time point as the prediction.

This procedure gave sensible results in all cases examined in the training stage.

5.3.4 Assessing accuracy of prediction

The prediction abilities of the models are assessed using various error rates as well as mean absolute error. Type k error (as used in McKenzie (2004)) measures the proportion of images not classified to within k units of their actual time point, and is defined as

$$E_{(k)} = \frac{1}{n}\sum_{i=1}^{n} I_k(t^i_{pred} - t^i_{act}),\ k = 0, 1, 2 \tag{5.15}$$

where n is the number of images for which the time is to be predicted, t^i_{pred} and t^i_{act} are respectively the (rounded) predicted time and actual state of time of

image j, and the indicator function I is such that

$$I_k(x) = \begin{cases} 1 & \text{if } |x| > k \\ 0 & \text{otherwise.} \end{cases}$$

The prediction ability of the models (5.9), (5.12), and (5.14) are assessed separately for the foreground as well as for the background PS moments.

Mean absolute error: Mean absolute error (MAE) of the prediction time is also calculated. This is an average deviation of the predicted time from the actual time, i.e.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |t^i_{pred} - t^i_{act}|, \tag{5.16}$$

where t_{pred} and t_{act} are respectively the rounded predicted time and the actual time state, and n is the total number of images for which the time is predicted.

5.4 Prediction using the Regression Approach

In this section we make use of the first two PS moments computed from different sizes 100^2, 256^2 and 512^2 of the pyramid images using all 6 SEs to predict the evolution state of individual image by means of the regression approach. The aim is to investigate whether the performance of the PS moments increases with increasing image size.

The PS moments data for any set of images is a 10000×12 matrix, where rows correspond to time states $t = 1, 2, \ldots, 100$ for 100 simulations and the first 6 columns contain PS means and the last 6 contain PS sds from the 6 SEs. Moments are averaged over the 100 simulations (there are 100 images at each time point), so the data is a matrix of size 100×12.

At this stage we built each combined regression model, i.e. straight line (5.9), quadratic (5.13) and cubic (5.14) regression models using the average PS moments (100 images at each time point) and the moments for each single image are used to predict the time. We assess the performance of straight line, quadratic and cubic regression models using the foreground as well as the background PS moments separately. At this stage all of the data is used to fit the models and we predicted the time for all of the available images at each time.

5.4.1 Prediction for 100^2 images

The 100^2 pyramid images were generated using parameters $\alpha = 0.5$, $\delta = 0.1$, and γ as discrete uniform [1, 2] in Section 4.2. Figure 5.7 shows some of those images at their final stage of evolution at the growth time step 100. These are seen to be quite variable in appearance.

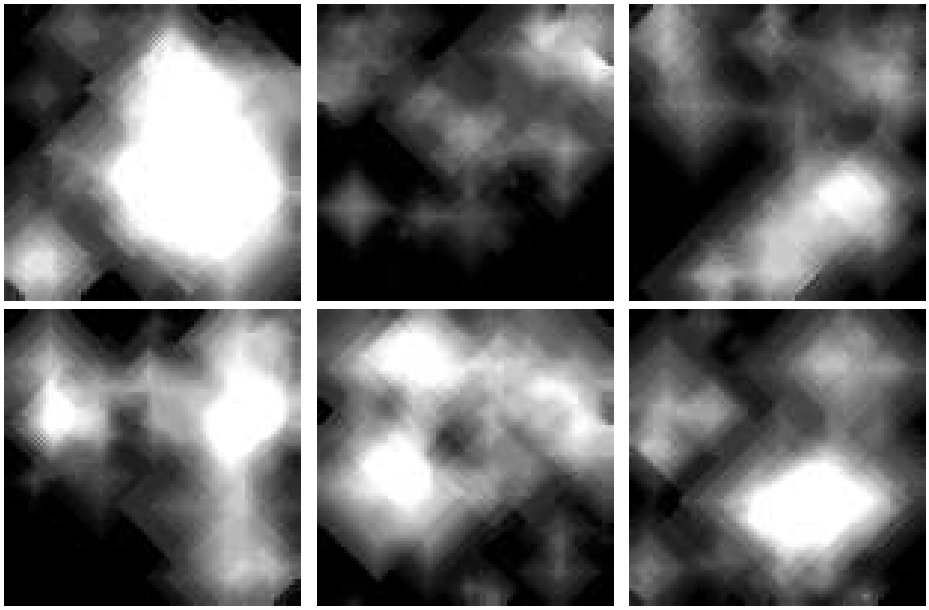
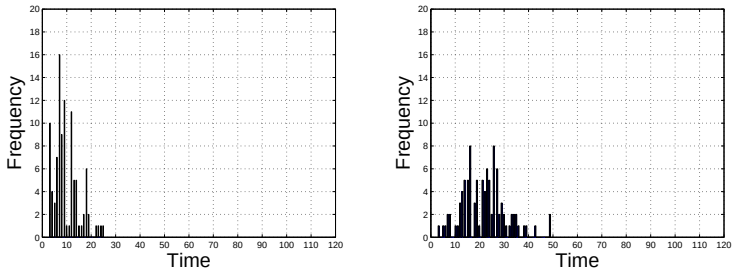


Figure 5.7: Final pyramid images of size 100^2 for different simulations, using $\alpha = 0.5$, $\delta = 0.1$, and γ= discrete uniform [1, 2], at time 100.

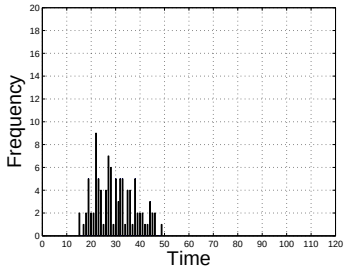
We obtained the predicted times for 100 different simulations, using straight line, quadratic and cubic regression, but we plotted the predicted times from cubic regression of the foreground PS moments at every tenth time point in Figure 5.8. The predicted times are centred at the actual time point, but are quite widely spread. Very often the predicted times are more than 20 units away from the actual time point, therefore the predicted time exceeds 100 (the largest actual time state) when the actual time is 80 or higher, so in general prediction is poor. Clipping the upper limit of the predicted time to 100 might decrease the error rate, but we did not do that in this chapter as evolution time in principle could be higher than 100.

The foreground and background moments were used separately for model fitting and predicting time, and the different types of error rates and MAEs as given in equations (5.15) and (5.16) are shown in Table 5.3. The type 0 error rate for the straight line, quadratic and cubic regressions using foreground PS moments are 96.1%, 95.7% and 95.2%, and 94.7%, 94.0% and 94.2% using background PS moments. The MAEs for the foreground moments are 8.389, 8.137 and 7.550,

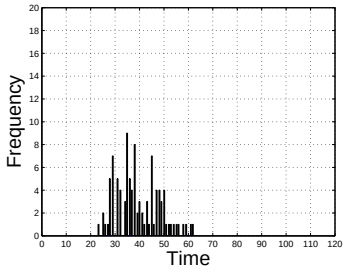


(a) $t = 10$

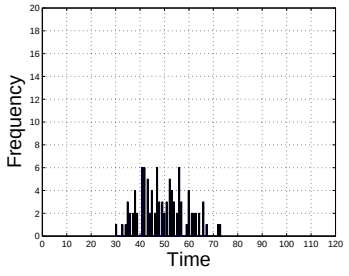
(b) $t = 20$



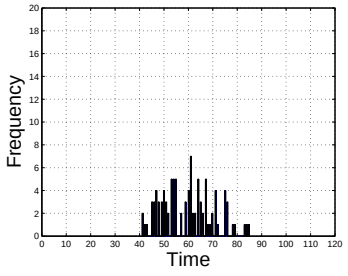
(c) $t = 30$



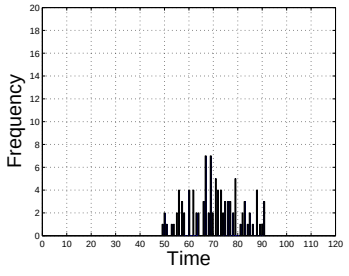
(d) $t = 40$



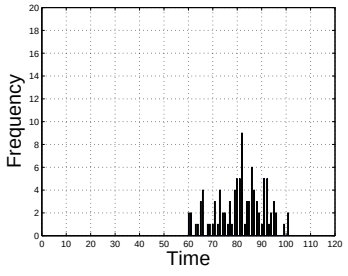
(e) $t = 50$



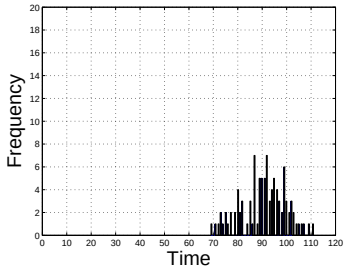
(f) $t = 60$



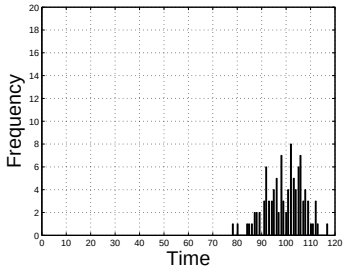
(g) $t = 70$



(h) $t = 80$



(i) $t = 90$



(j) $t = 100$

Figure 5.8: Histograms of the predicted times for the 100^2 pyramid images using cubic regression modelling of the first two foreground PS moments from all 6 SEs, for actual times $t = 10,\ 20,\ \ldots,\ 100$.

while much lower MAEs were obtained from the background moments, i.e. 5.769, 5.344 and 5.460 for straight line, quadratic and cubic regression respectively. The background PS moments produced lower error rates and MAEs for any model than the foreground moments. Cubic regression works slightly better for the foreground moments but quadratic regression works better for the background moments.

Figure 5.9 shows the MAEs, type 0, type 1, and type 2 error rates for all 3 models and the cubic regression provides slightly lower error rates than the straight line and quadratic regression models. In terms of MAE, the quadratic and cubic regression models are equally effective, especially after evolution time 10 for the foreground moments and for the background moment between times 20 and 80.

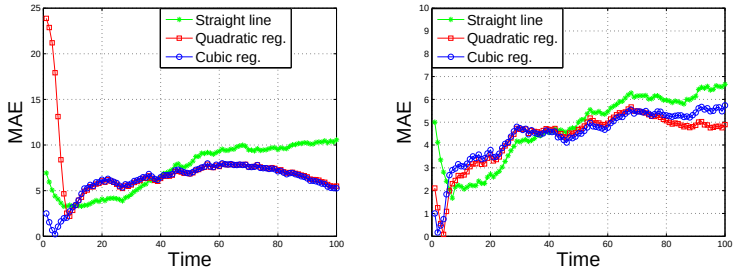
Table 5.3: Average classification error rates and MAEs from prediction of time for all of the 100^2 pyramid images, using the first 2 PS moments from 6 SEs.

Error rate	Foreground PS moments		
	Straight line	Quadratic reg.	Cubic reg.
Type 0	0.961	0.957	0.952
Type 1	0.885	0.870	0.854
Type 2	0.813	0.793	0.764
MAE	8.389	8.137	7.550
Error rate	Background PS moments		
	Straight line	Quadratic reg.	Cubic reg.
Type 0	0.947	0.940	0.942
Type 1	0.835	0.818	0.820
Type 2	0.724	0.696	0.702
MAE	5.769	5.344	5.460

5.4.2 Prediction for 256^2 images

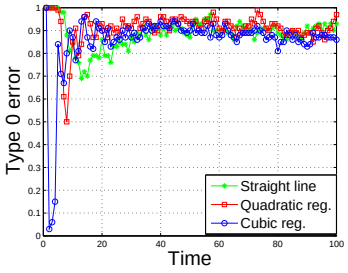
We then applied granulometry on the foreground of the 256^2 pyramid images stacks generated using parameters $\alpha = 0.5$, $\delta = 0.3$, and γ as discrete uniform $[1,\ 2]$. Some images at their final stage of evolution are shown in Figure 5.10.

Again evolution times were predicted separately for the foreground and background PS moments from straight line, quadratic and cubic regression models, and the different error rates and MAEs were computed. The predicted times for the foreground moments for 256^2 pyramid images using cubic regression are shown as frequency histograms in Figure 5.11 at every tenth time point. Again

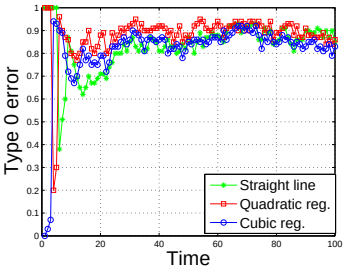


(a) MAE

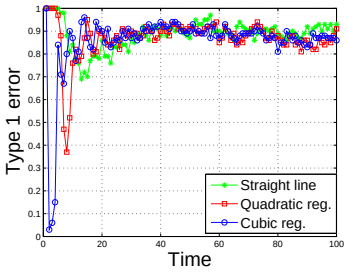
(b) MAE



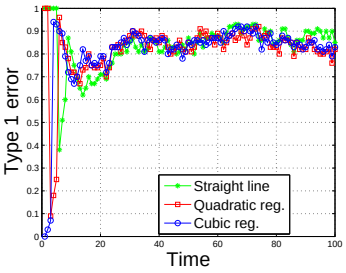
(c) Type 0 error



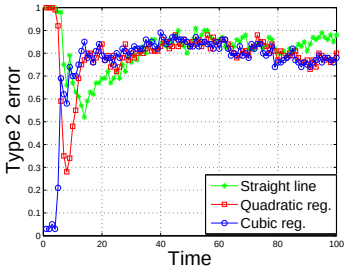
(d) Type 0 error



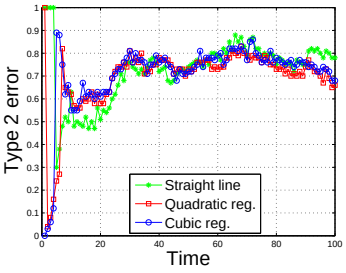
(e) Type 1 error



(f) Type 1 error



(g) Type 2 error



(h) Type 2 error

Figure 5.9: MAE (a)-(b), type 0 error (c)-(d), type 1 error (e)-(f) and type 2 error (g)-(h) for the first 2 foreground (left) and background PS moments (right) from the 100^2 pyramid images using all 3 regression models.

the predicted times are quite spread out, although they are centred at the actual time points.

The error rates and MAEs for all 3 models for the foreground and background PS moments from the 256^2 pyramid images are shown in Table 5.4 and Figure 5.12. Again the background moments produce lower error rates and MAEs compared to the foreground moments. Again for the foreground moments cubic

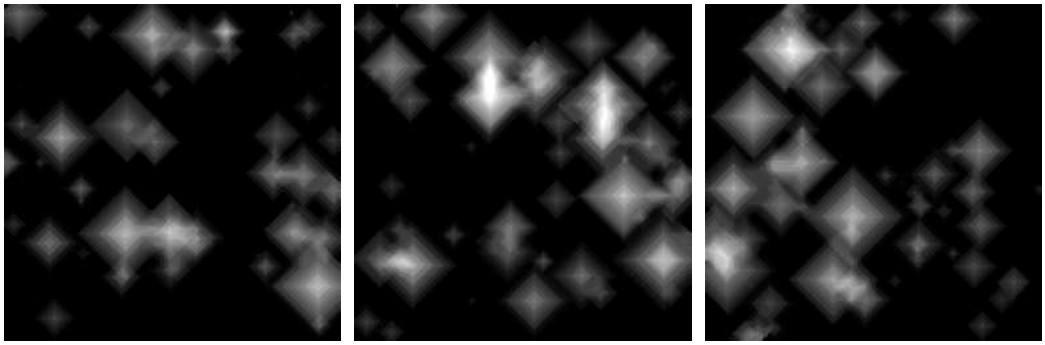


Figure 5.10: Some pyramid images of size 256^2 for different simulations, $\alpha = 0.5$, $\delta = 0.3$, and γ=discrete uniform [1, 2], at time 100.

regression produces lower error rates and MAEs than straight line and quadratic regression. For the background moments quadratic regression produces the lowest error rates but cubic regression produces the lowest MAEs.

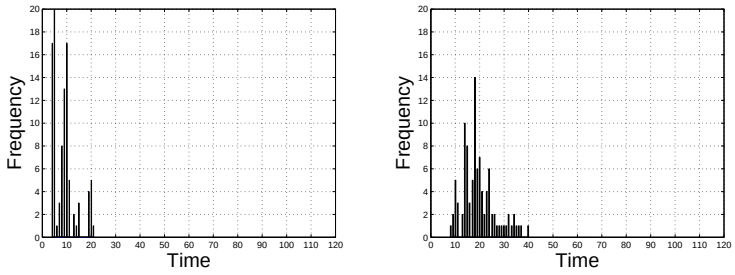
Table 5.4: Average classification error rates and MAEs from all the 256^2 pyramid images using the first 2 PS moments from 6 SEs.

Error rate	Foreground PS moments		
	Straight line	Quadratic reg.	Cubic reg.
Type 0	0.949	0.948	0.941
Type 1	0.852	0.841	0.834
Type 2	0.756	0.737	0.727
MAE	7.164	6.397	6.103
Error rate	Background PS moments		
	Straight line	Quadratic reg.	Cubic reg.
Type 0	0.918	0.906	0.942
Type 1	0.756	0.726	0.820
Type 2	0.605	0.568	0.577
MAE	4.015	3.952	3.637

5.4.3 Prediction for 512^2 images

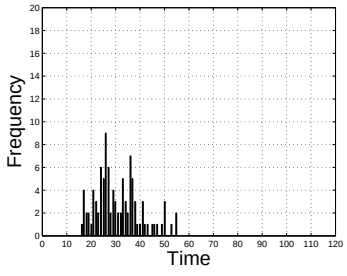
Pyramid images of size 512^2, generated using parameters $\alpha = 0.5$, $\delta = 0.3$, and γ as discrete uniform [1, 3] (with a increased rate of growth), are shown in Figure 5.13.

Although the error rates using the background PS moments from the 100^2 and 256^2 images were slightly lower than with the foreground PS moments, computing PS moments from the background of an image is more time consuming than from

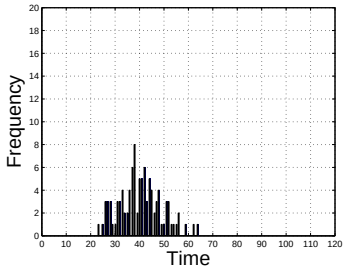


(a) $t = 10$

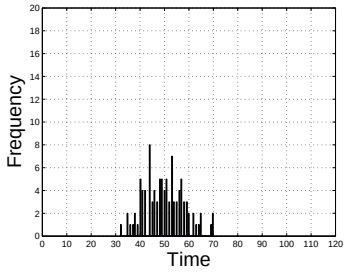
(b) $t = 20$



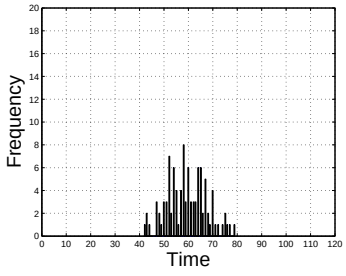
(c) $t = 30$



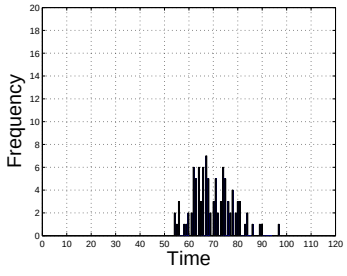
(d) $t = 40$



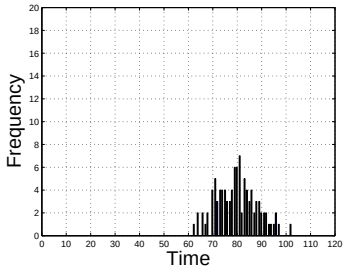
(e) $t = 50$



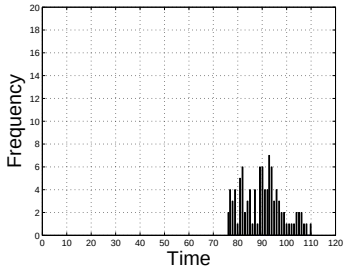
(f) $t = 60$



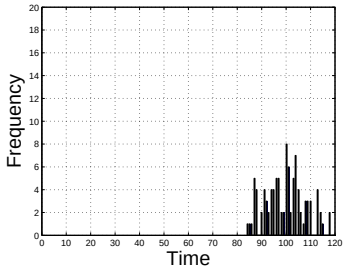
(g) $t = 70$



(h) $t = 80$

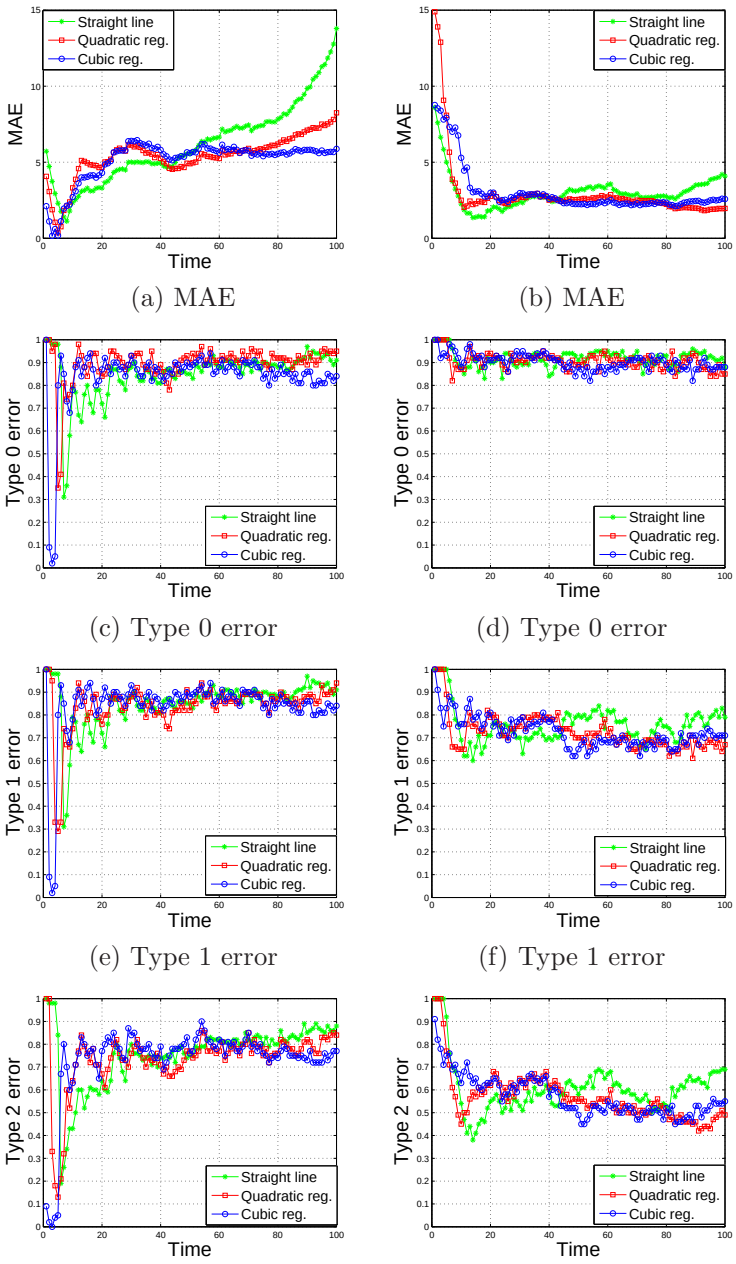


(i) $t = 90$



(j) $t = 100$

Figure 5.11: Histograms of the predicted times for the 256^2 pyramid images using cubic regression modelling of the first two foreground PS moments using all 6 SEs, for actual times $t = 10,\ 20,\ \ldots,\ 100$.



(a) MAE (b) MAE

(c) Type 0 error (d) Type 0 error

(e) Type 1 error (f) Type 1 error

(g) Type 2 error (h) Type 2 error

Figure 5.12: MAE (a)-(b), type 0 error (c)-(d), type 1 error (e)-(f) and type 2 error (g)-(h) for the first 2 foreground (left) and background PS moments (right) from the 256^2 pyramid images using all 3 regression models.

the foreground image. Therefore we computed PS moments using 6 SEs only from the foreground of the 512^2 images. Again all 3 regression models were used to predict evolution times. The predicted times from the cubic regression model at every 10^{th} time point are shown in Figure 5.14. Still the predicted times are wide spread.

The error rates and MAEs are shown in Table 5.5 and Figure 5.15. The overall

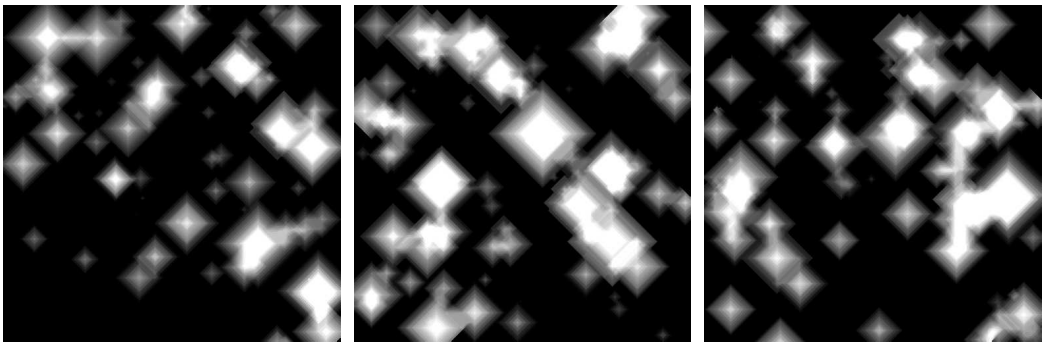


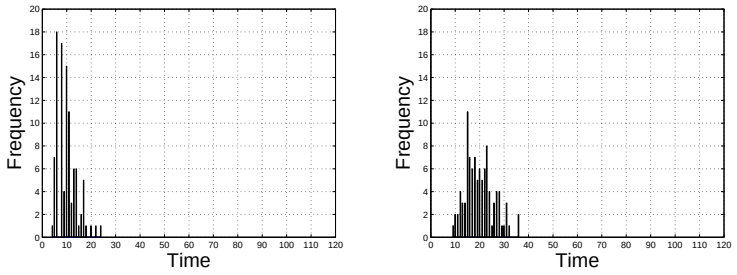
Figure 5.13: Some pyramid images of size 512^2 for different simulations, $\alpha = 0.5$, $\delta = 0.3$, and γ=discrete uniform [1, 3], at time 100.

Table 5.5: Average classification error rates and MAEs from all the 512^2 pyramid images using the first 2 PS moments from 6 SEs.

Error rate	Foreground PS moments		
	Straight line	Quadratic reg.	Cubic reg.
Type 0	0.943	0.935	0.927
Type 1	0.828	0.811	0.785
Type 2	0.714	0.683	0.651
MAE	6.016	5.188	4.864

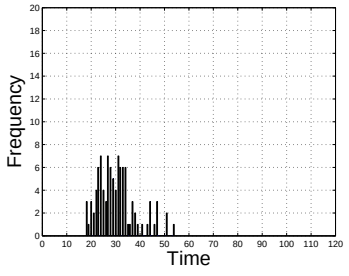
type 0 error rate for straight line, quadratic and cubic regression is 94.3%, 93.5% and 92.7% respectively and the corresponding MAEs are 6.016, 5.188 and 4.864. Comparing Tables 5.3, 5.4 and 5.5 it can be concluded that the accuracy of the PS moments in predicting time increases with size of the image. In general MAE decreases with increasing image size.

Because of this high prediction error we focused on the error distribution and investigated whether any adjustment can be made to the errors to increase the prediction ability of the models. Figure 5.16 represents the distribution of the prediction error (predicted time−actual time) at every 10^{th} time point from the cubic regression model using foreground PS moments from the 256^2 pyramid images. The prediction error for 100 different simulations is presented for specific time points. We can see that the errors are random and there is no positive or negative bias. However, only for a very few simulations does the predicted time coincide with the actual time.

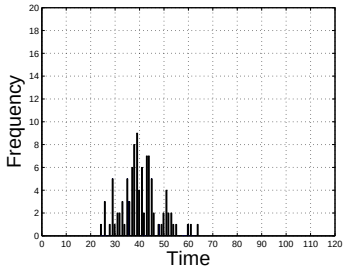


(a) $t = 10$

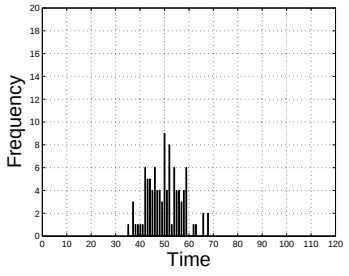
(b) $t = 20$



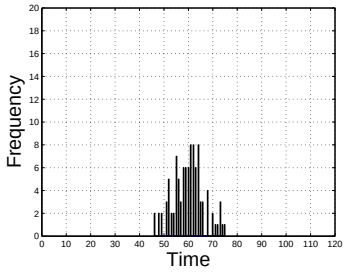
(c) $t = 30$



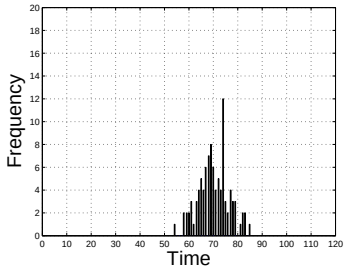
(d) $t = 40$



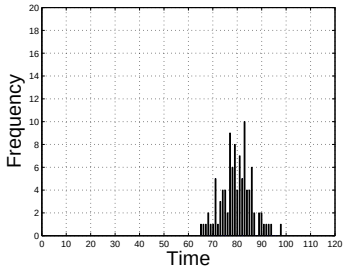
(e) $t = 50$



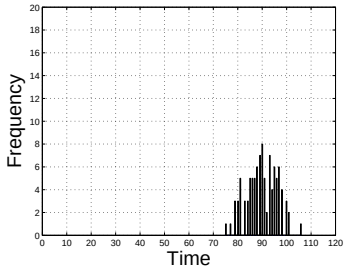
(f) $t = 60$



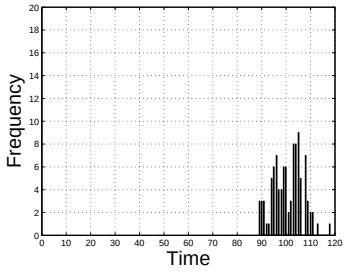
(g) $t = 70$



(h) $t = 80$

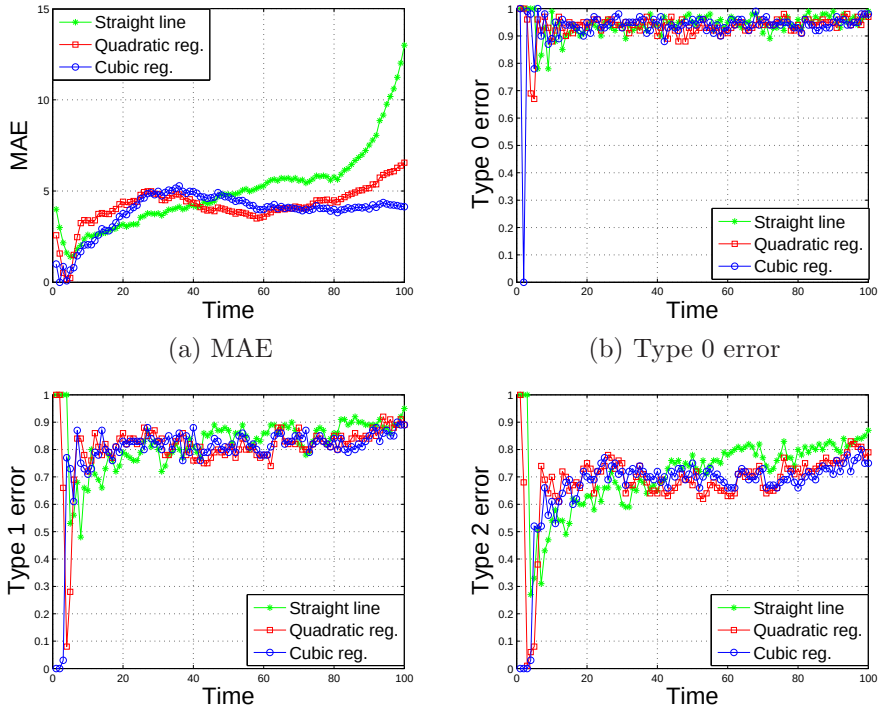


(i) $t = 90$



(j) $t = 100$

Figure 5.14: Histograms of the predicted times for the 512^2 pyramid images using cubic regression modelling of the first two foreground PS moments from all 6 SEs, for actual times $t = 10,\ 20,\ \ldots,\ 100$.



(a) MAE (b) Type 0 error

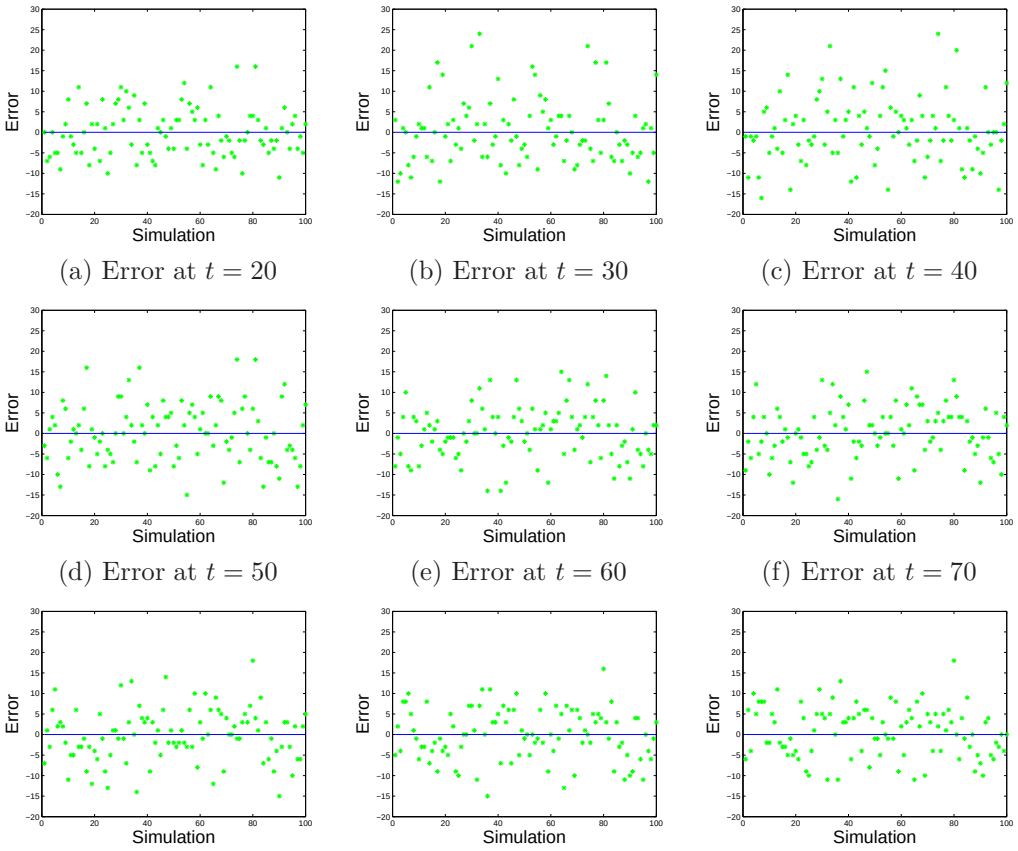
(c) Type 1 error (d) Type 2 error

Figure 5.15: MAE and type 0, type 1 and type 2 error for the first 2 foreground PS moments from all 6 SEs from the 512^2 pyramid images, using all 3 regression models.

5.4.4 Prediction using PCs

Rather than using the original PS moments for prediction, here we investigate the usefulness of PCA. The PCs of the normalised PS moments derived in Section 4.3.4 were modelled as a function of evolution time using regression models, and were used to predict time. Figure 5.17 shows the error rates using the first 2 PCs from foreground PS moments from all 6 SEs of the 256^2 pyramid images. We used all three regression models but none of them outperforms the previous moments-based results. Comparing the cubic regression results (blue lines) in Figure 5.17 with the REG results in Figure 5.18(a), (c), (e) and (g) (red lines), we can say that prediction is not improved by employing PCs rather than using the PS moments directly.

Computation of granulometric moments from the 512^2 pyramid images is time consuming, especially for a disk SE. Although the PS moments from the 512^2 pyramid images provide slightly lower MAE and error rates, we will now only use the PS moments from the 256^2 images in the other classifiers to compare the results with the corresponding regression models.

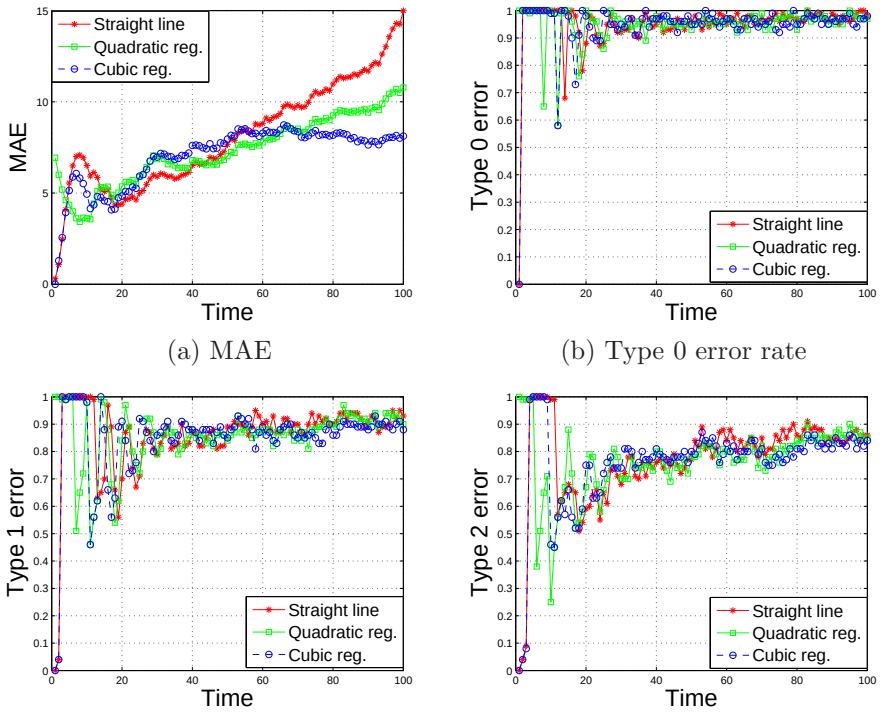


(a) Error at $t = 20$ (b) Error at $t = 30$ (c) Error at $t = 40$

(d) Error at $t = 50$ (e) Error at $t = 60$ (f) Error at $t = 70$

(g) Error at $t = 80$ (h) Error at $t = 90$ (i) Error at $t = 100$

Figure 5.16: Prediction error (predicted time−actual time) plotted against simulation number, using the cubic regression model with 70% of the 2 foreground PS moments using 6 SEs from the 256^2 pyramid images.



(a) MAE (b) Type 0 error rate

(c) Type 1 error rate (d) Type 2 error rate

Figure 5.17: Type 0, type 1 and type 2 error rates and MAEs using the first 2 PCs from 12 foreground PS moments (2 PS moments from 6 SEs) of the 256^2 pyramid images using all 3 regression models.

5.5 Prediction using Other Classifiers

In this section, the first 2 PS moments from the foreground and background of the 256^2 pyramid images as well as the 256^2 ellipse images using all 6 SEs are used in some other classifiers, i.e. a support vector machine (SVM), a feed forward neural network (FF-NNET) and linear discriminant analysis (LDA), and their classification accuracy is compared with that of the regression approach. Therefore each dataset is of size 10000×13, where the first 6 columns contain the PS means using 6 SEs, the next 6 columns consist of the PS sds using 6 SEs and the last column contains the actual time state. For all classifiers in this section, 70% of the moments were randomly sampled and used to train the classifiers and the rest of the moments were used for testing. We calculate training or test set error as the number of misclassifications divided by the total number of images in the training or test set, or as the mean absolute error. This was done 10 times and results were averaged to give overall performance of the method.

5.5.1 Results from SVM

Different types of kernel can be used, such as *linear*, *polynomial*, *radial basis* (also known as the Gaussian kernel), and *sigmoid* (tanh). A detailed description is given in Section 3.12. The radial basis kernel is by far the most popular kernel used in SVMs (Chellappa and Chatterjee (1985)), mainly because of its localised and finite response across the entire range of the real x-axis. R library e1071 is used here for SVMs (details are given in Appendix II(a)).

Although the generalisation ability of the SVM is relatively robust to variations in the parameter settings (Li (2009)), we tested several kernels and a wide range of the parameter values in the kernel function and the cost or regularisation parameter in training to ensure high accuracy. We investigated the performance of the linear, polynomial and radial basis kernels with the associated parameter values (in the last two cases). We used a grid search approach for finding an appropriate kernel and the optimum value of any kernel parameters and also the cost parameter, in terms of the training set error rates using a single training set of 70% of the PS moments randomly selected for each set of images.

PS moments from the pyramid images

Using the first 2 foreground PS moments of the 256^2 pyramid images, Table 5.6 contains error rates for different combinations of the value of the cost and the kernel parameter γ for the radial basis and the polynomial kernel. For the polynomial kernel, any value of the parameter η between 1 to 5 produced the same results but the default value 0 produced a higher error rate, and we used $\eta = 1$. Only $\gamma = 1$ and cost $= 1$ for the radial basis kernel gave 100% correct classification and $\gamma = 0.2$ and cost $= 100$ produced the second lowest error rate of 10%. None of the cases for the polynomial kernel produced 100% correct classification, although the error rates were lower in general for a higher cost in both kernels. Using the first 2 background PS moments, the radial basis kernel and polynomial kernel produced very similar error rates for different combinations of γ and cost to those shown here, so these results are not shown.

In the linear kernel there is no kernel parameter, so we considered only the cost parameter. Training set error rates corresponding to different values of the cost for the linear kernel are shown in Table 5.7, for both foreground and background PS moments from the pyramid images. Any cost of 20 or more produced 100% correct classification for both datasets. Therefore, a linear kernel with cost of 100 was used for both datasets.

Table 5.6: Training set error rates for different combinations of cost and γ for the radial basis kernel and the polynomial kernel with $\eta = 1$, using foreground PS moments from the 256^2 pyramid images. A 0 below means exactly 0. The values in bold are the best results.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.1	0.92	0.81	0.66	0.61	0.60	0.54	0.44	0.27	0.20	0.15	0.12
0.2	0.91	0.71	0.63	0.52	0.30	0.21	0.17	0.14	0.12	0.11	**0.10**
0.3	0.89	0.66	0.57	0.33	0.23	0.19	0.17	0.14	0.13	0.12	0.12
0.4	0.89	0.66	0.44	0.28	0.22	0.19	0.17	0.16	0.16	0.15	0.15
0.5	0.89	0.67	0.37	0.27	0.23	0.20	0.19	0.19	0.18	0.18	0.17
0.6	0.88	0.65	0.35	0.27	0.24	0.22	0.21	0.20	0.19	0.19	0.18
0.7	0.88	0.60	0.33	0.27	0.25	0.23	0.22	0.21	0.21	0.21	0.20
0.8	0.88	0.57	0.33	0.28	0.25	0.24	0.23	0.23	0.22	0.22	0.22
0.9	0.88	0.54	0.37	0.29	0.26	0.25	0.25	0.24	0.24	0.24	0.24
1	**0**	0.87	0.51	0.37	0.30	0.27	0.26	0.26	0.25	0.25	0.25
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.1	0.92	0.78	0.73	0.70	0.68	0.66	0.64	0.63	0.62	0.61	0.60
0.2	0.81	0.62	0.57	0.53	0.51	0.50	0.49	0.48	0.47	0.46	0.45
0.3	0.70	0.52	0.48	0.45	0.45	0.43	0.42	0.41	0.40	0.40	0.39
0.4	0.64	0.47	0.43	0.41	0.40	0.38	0.37	0.36	0.35	0.34	0.33
0.5	0.59	0 .43	0.40	0.37	0.35	0.35	0.33	0.32	0.31	0.31	0.29
0.6	0.55	0.40	0.37	0.34	0.32	0.30	0.29	0.29	0.28	0.26	0.26
0.7	0.51	0.37	0.36	0.35	0.30	0.27	0.26	0.26	0.25	0.24	0.23
0.8	0.49	0.35	0.33	0.32	0.30	0.25	0.25	0.23	0.24	0.22	0.22
0.9	0.46	0.32	0.30	0.29	0.24	0.23	0.22	0.21	0.21	0.20	0.20
1	0.45	0.30	0.25	0.25	0.23	0.22	0.22	0.21	0.21	0.21	0.19

PS moments from the ellipse images

Now we investigate the optimum kernel and its parameter values using the first 2 foreground and background PS moments using 6 SEs from the 256^2 ellipse images. Table 5.8 shows the error rates for the first 2 foreground PS moments using the radial basis and polynomial kernels. The error rate with any combination of the parameters is high (at least 16% for the radial basis except for $\gamma = 1$ and cost $= 1$ which gives 0% error, and 25% for the polynomial kernel). Different values of γ between 0 and 1 in steps of size 0.1 were tested for the polynomial kernel as well but are not all shown in the table. Any combination of γ and cost using the first 2 background PS moments also produced a high error rate, so they are not included here.

Table 5.7 shows the error using the linear kernel for the foreground and back-

Table 5.7: Training set error rates for different values of cost for the linear kernel using PS moments from the 256^2 pyramid and ellipse images.

	Pyramid images										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
Foreground	0.82	0.29	0	0	0	0	0	0	0	0	**0**
Background	0.80	0.25	0	0	0	0	0	0	0	0	**0**
	Ellipse images										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
Foreground	0.84	0.27	0	0	0	0	0	0	0	0	**0**
Background	0.59	0.09	0.09	0.01	0.01	0.01	0.01	0.01	0.01	0.01	**0.01**

ground PS moments from the ellipse images. The linear kernel produced 100% correct classification for any cost of 20 or above for the foreground PS moments but for the background moments a cost of 30 or more is best, with 1% error. However we used a cost of 100 for both set of moments.

Although many authors (e.g. Li et al. (2003), Chellappa and Chatterjee (1985), Chaplot et al. (2006)) achieved better classification accuracy using the radial basis kernel, here the linear kernel is the most appropriate kernel with any cost of 20 or more. The outstanding results achievable by SVM greatly depend on the choice of an appropriate kernel and its parameters. For example, SVM yielded 100% correct classification for the foreground PS moments from the pyramid images using a linear kernel with cost of 20 or more, but for a radial basis kernel only $\gamma = 1$ and cost = 1 produced 100% correct classification and the second best combination of $\gamma = 0.2$ and cost = 100 gave a 10% error rate. For a polynomial kernel, the optimum parameters are $\gamma = 1$ and cost = 100 with an error rate of 19%. Therefore, choosing the most appropriate kernel is important for best performance.

5.5.2 Results from LDA

Section 3.9 explains the LDA approach for binary and multi-class classification. LDA was employed by using function *lda* in R library MASS. We applied LDA to the first 2 foreground and background PS moments from the 256^2 pyramid and ellipse images. See Appendix II(b) for details of the R function.

Table 5.8: Training set error rates for different combinations of cost and γ for the radial basis kernel and the polynomial kernel with $\eta = 1$, using foreground PS moments from the 256^2 ellipse images.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.1	0.92	0.83	0.68	0.60	0.59	0.53	0.43	0.31	0.24	0.19	0.16
0.2	0.90	0.74	0.70	0.63	0.52	0.34	0.26	0.22	0.20	0.17	0.16
0.3	0.90	0.68	0.57	0.54	0.34	0.27	0.24	0.23	0.20	0.20	0.19
0.4	0.89	0.66	0.54	0.45	0.31	0.27	0.25	0.24	0.24	0.23	0.23
0.5	0.89	0.65	0.40	0.31	0.27	0.26	0.26	0.25	0.25	0.25	0.25
0.6	0.89	0.62	0.53	0.47	0.40	0.28	0.28	0.28	0.26	0.26	0.26
0.7	0.88	0.58	0.53	0.47	0.35	0.30	0.29	0.29	0.28	0.28	0.27
0.8	0.88	0.57	0.43	0.42	0.40	0.30	0.25	0.24	0.23	0.23	0.20
0.9	0.88	0.53	0.47	0.45	0.40	0.35	0.35	0.32	0.32	0.31	0.30
1	**0**	0.87	0.52	0.47	0.40	0.37	0.32	0.32	0.31	0.30	0.30
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.1	0.91	0.81	0.76	0.70	0.68	0.67	0.65	0.64	0.62	0.61	0.62
0.5	0.60	0 .45	0.41	0.40	0.38	0.36	0.35	0.33	0.32	0.32	0.31
0.9	0.48	0.31	0.29	0.29	0.28	0.27	0.27	0.26	0.26	0.25	0.25

5.5.3 Results from FF-NNET

The neural network classifier was applied in R, in the library *nnet*. It fits a feed-forward single hidden layer neural network (FF-NNET). The architecture of such networks is explained in Section 3.11 and details of the R function are in Appendix II(c). Of many possible activation functions, R allows only a logistic or linear activation function. Both were tested. A logistic activation function was used for the final computation but the use of a linear activation function instead had little effect on the results. Velten (2009) suggests scaling the input data to the range 0 to 1 for neural networks. It was found that scaling the data to range between 0 to 1 increased the overall error rate, whereas normalising the data (to 0 mean and unit variance) substantially decreased the overall error rate for the FF-NNET in some cases and often improved the results. So the data were normalised before using the FF-NNET. This made it comparable to use of SVM, as SVM normalises the data by default (though not normalising did not alter the SVM results much). Normalisation of the data is a part of the LDA computation, so further normalisation had no effect on the results. It was also found that normalising the input data to the regression method worsened the

results.

The optimum values of some important parameters in the FF-NNET, namely decay, rang and number of hidden units were chosen for each set of features. The value of rang specifies the range [-rang rang] from which the initial weights are randomly taken, and decay controls the decay of the weights with successive iterations of the fitting process and penalises over-fitting. The maximum number of iterations was set to 5000, to ensure that the optimisation problem converged.

We carried out a grid search approach to find the optimum value of rang and decay, using 7 hidden units (the maximum possible number of units for these features). Different values of decay, i.e. 10^{-4}, 10^{-3}, 10^{-2}, and 10^{-1} were used, and the values of rang were taken as 0.1, 0.3, 0.5, 0.7, 0.9 and $1/\max(|\boldsymbol{x}|)$ where $\boldsymbol{x}$ is the input data. The R default value of rang is 0.7 and a value of about 0.5 is recommended unless the inputs are large, in which case it should be chosen so that rang$\times \max(|\boldsymbol{x}|)$ is about 1, where $\boldsymbol{x}$ is the input data. For the pyramid images, the optimum pairs of values for decay and rang were (10^{-4}, 0.3) for the foreground PS moments and (10^{-4}, $1/\max(|\boldsymbol{x}|)$) for the background PS moments. For the ellipse images, the optimum values were (10^{-4}, $1/\max(|\boldsymbol{x}|)$) and (10^{-3}, $1/\max(|\boldsymbol{x}|)$) for the foreground and background PS moments respectively. Using the best values of decay and rang for each dataset we then again used a grid search to choose the number of hidden units. Seven units was the best choice for the pyramid or ellipse foreground or background PS moments. Table 5.9 shows the training set error rates for the foreground and background PS moments of the 256^2 pyramid and ellipse images. According to Fujita (1998), error decreases with the number of units in the hidden layer, which is true in our cases. The number of neurons in the hidden layer greatly affects the classification results using the background PS moments for the ellipse images, but for the other moments datasets the classification results did not vary much with the number of hidden neurons (all are poor).

We computed the different error rates and MAEs with the first two (mean and sd) foreground and background moments for the 256^2 pyramid images for all classifiers including the cubic regression model. MAE, type 0, type 1 and type 2 error rates from the same sets of pyramid images are shown in Table 5.10 and plotted against time in Figure 5.18. For both foreground and background moments, SVM attained 100% correct classification. Type 0 error rates for the regression approach, LDA and FF-NNET using the foreground moments are 94.5%, 93.8% and 91.1%, whereas these are 90.8%, 88.2% and 85.1% respectively for the background moments. Type 1 and type 2 errors are also high. For the other

Table 5.9: Training set error rates from a grid search approach for finding the optimum number of hidden neurons for FF-NNET, with the 256^2 images of pyramids and ellipses. The values in bold are the best values.

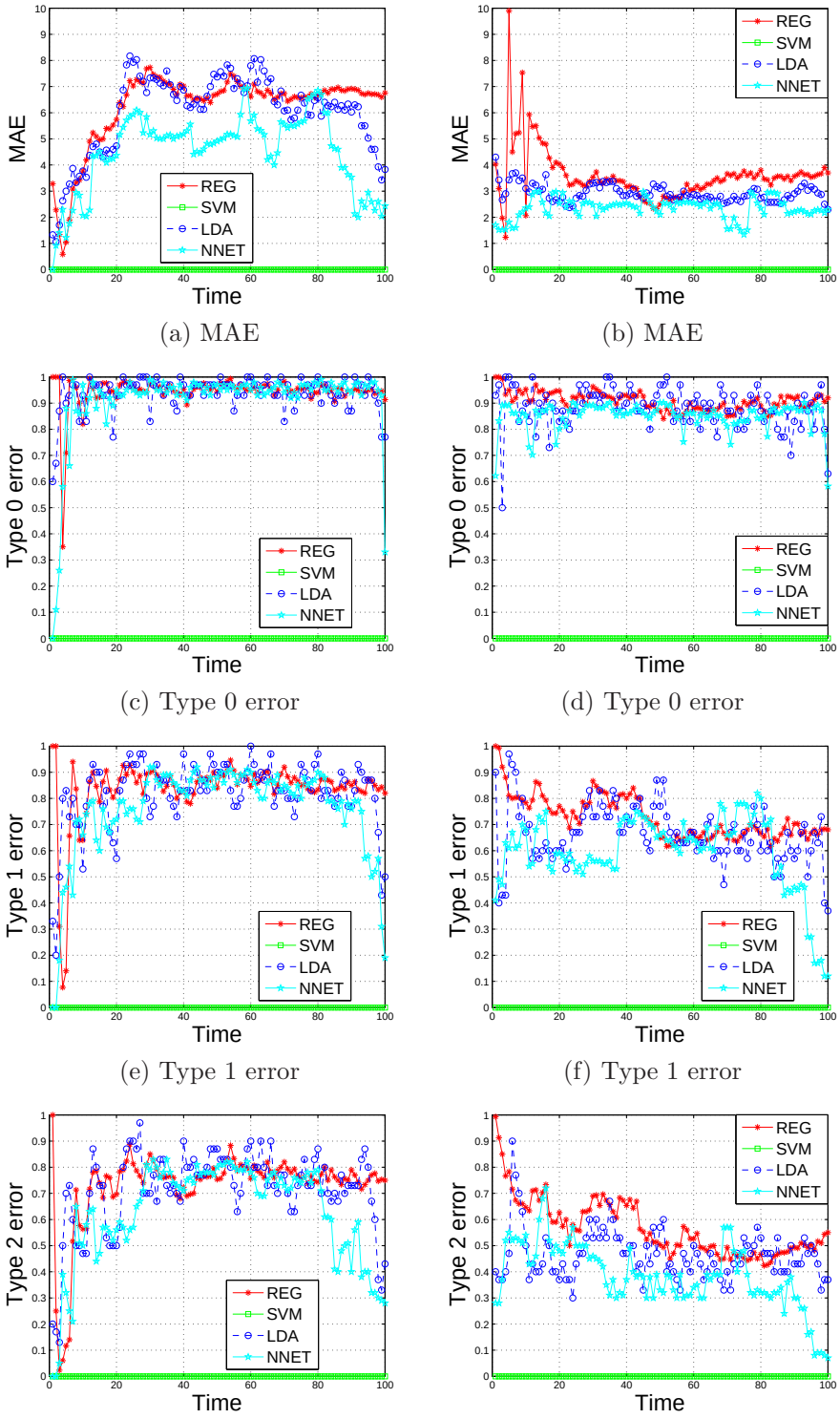
	Number of neurons						
PS moments	1	2	3	4	5	6	7
Foreground pyramid	0.94	0.94	0.93	0.93	0.93	0.92	**0.91**
Background pyramid	0.90	0.88	0.88	0.87	0.87	0.87	**0.85**
Foreground ellipse	0.90	0.90	0.90	0.90	0.89	0.89	**0.88**
Background ellipse	0.15	0.30	0.33	0.20	0.20	0.10	**0.08**

classifiers, error rates and MAEs for the background moments are lower than the corresponding foreground ones.

Table 5.10: Average test set classification error rates and MAEs for the 256^2 pyramid and ellipse images using PS moments; results are averaged over 10 runs.

	Pyramid images							
Error rate	Foreground PS moments				Background PS moments			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
Type 0	0.945	0.000	0.938	0.911	0.908	0.000	0.882	0.851
Type 1	0.837	0.000	0.819	0.758	0.727	0.000	0.661	0.594
Type 2	0.729	0.000	0.713	0.623	0.568	0.000	0.464	0.385
MAE	6.716	0.000	5.994	4.497	3.578	0.000	2.929	2.343
	Ellipse images							
Error rate	Foreground PS moments				Background PS moments			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
Type 0	0.937	0.000	0.926	0.886	0.903	0.002	0.881	0.123
Type 1	0.821	0.000	0.800	0.713	0.703	0.001	0.650	0.059
Type 2	0.703	0.000	0.692	0.560	0.520	0.000	0.444	0.023
MAE	5.716	0.000	5.670	3.781	3.328	0.002	2.706	0.229

We used all classifiers, including the regression approach, using the first two PS moments from all 6 SEs from the foreground and background separately of the 256^2 ellipse images, and computed the type 0, type 1, type 2 error rates and MAEs. The results are shown in Table 5.10 and Figure 5.19. Again SVM was 100% accurate for the foreground moments and 99.8% accurate for the background ones. For the foreground moments, the error rates for the regression classifier, LDA and the FF-NNET are 93.7%, 92.6% and 88.6%, while for the background moments these are 90.3%, 88.1% and 12.3% respectively. REG was



(a) MAE (b) MAE

(c) Type 0 error (d) Type 0 error

(e) Type 1 error (f) Type 1 error

(g) Type 2 error (h) Type 2 error

Figure 5.18: Mean absolute error (a)-(b), type 0 error (c)-(d), type 1 error (e)-(f), and type 2 error (g)-(h) for foreground (left) and background PS moments (right) using different classifiers for the 256^2 pyramid images.

rather erratic in performance when using the background moments to predict at later time points (Figure 5.19 (b)). At these points the predicted times were far away from the actual times (more than 30 units higher or 30 units lower) which contributed to the fluctuation of the MAE. (This was not due to any difficulty in finding a real root of the cubic polynomial used for prediction as there were no negative roots and only one real root in these cases, and for the synthetic images generally). Again background PS moments work better overall for all classifiers, especially for FF-NNET.

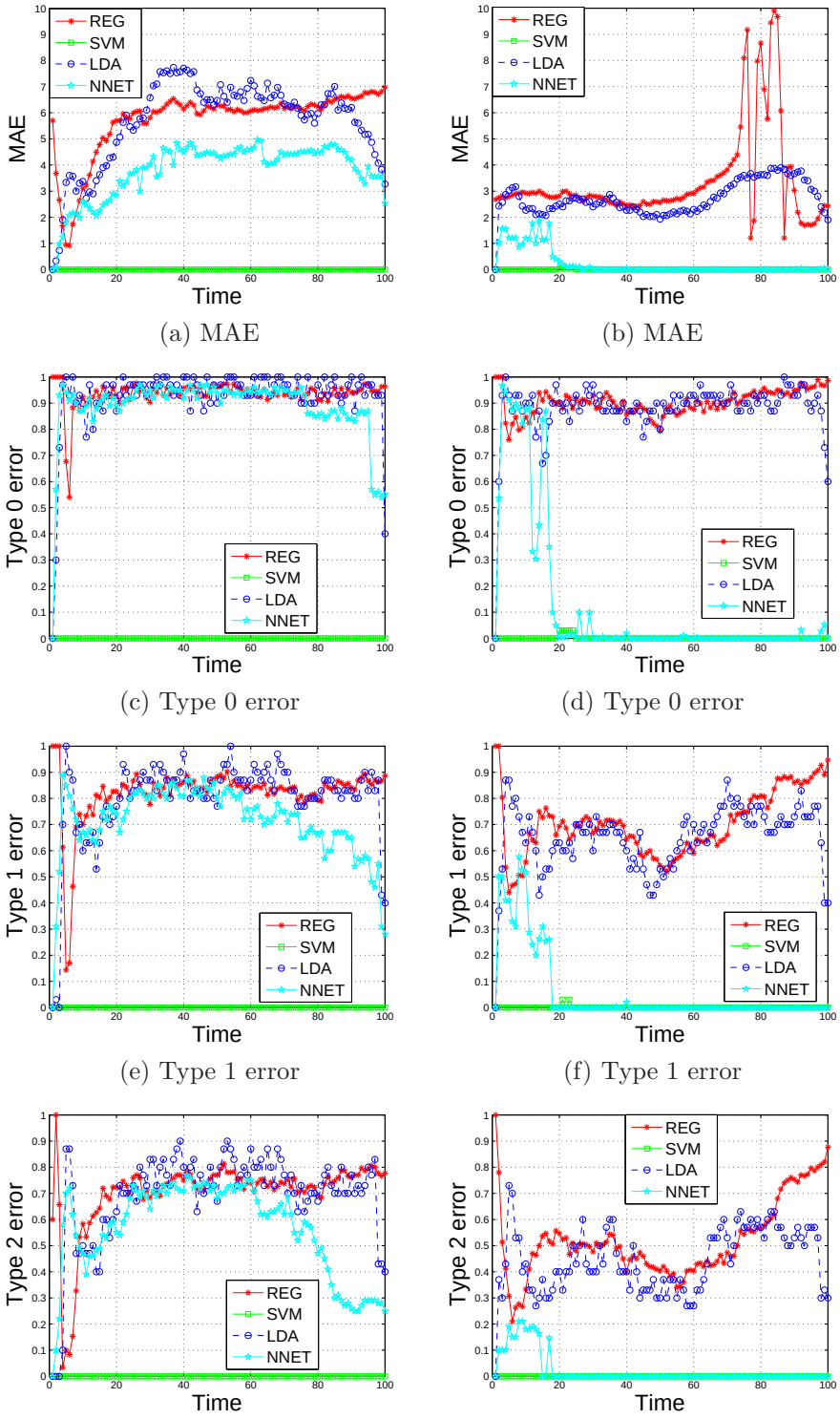
We also combined the foreground and background moments for each type of image and applied all the different approaches for classification using all of these moments (2 moments for each of the 6 SEs for both the foreground and background) but the results reflected the average of the results for the foreground and background moments separately.

Prediction using median PS moments

The regression modelling above used average PS moments (averaged over 100 simulations). In case the average PS moments were affected by unusual values of the moments, we checked whether using median PS moments instead of average PS moments in the regression modelling would provide improved results. We computed the foreground and background median PS moments for the 256^2 pyramid and ellipse images. The time trends look very similar to those of the corresponding average PS moments but in some cases are slightly smoother than for the average PS moments. We computed the error rates for all median PS moments sets using the regression approach. The error rates and MAEs are not very different from the corresponding average PS moments-based results (Table 5.11). Therefore, use of median PS moments is not beneficial over the average PS moments in the regression modelling.

Table 5.11: Average test set classification error rates and MAEs for the 256^2 pyramid and ellipse images using median PS moments, with the regression approach; results are the average of 10 runs.

Error rate	Pyramid foreground	Pyramid background	Ellipse foreground	Ellipse background
Type 0	0.946	0.902	0.937	0.905
Type 1	0.838	0.725	0.809	0.704
Type 2	0.731	0.565	0.686	0.520
MAE	6.313	6.269	5.696	3.165



(a) MAE (b) MAE

(c) Type 0 error (d) Type 0 error

(e) Type 1 error (f) Type 1 error

(g) Type 2 error (h) Type 2 error

Figure 5.19: Mean absolute error (a)-(b), type 0 error (c)-(d), type 1 error (e)-(f), and type 2 error (g)-(h), using foreground (left) and background PS moments (right) in different classifiers, for the 256^2 ellipse images.

5.6 Conclusions

None of the classifiers except SVM provides good classification rates for any of the PS moment feature sets. We believe that the possible reasons for this disappointing performance are largely due to the nature of the synthetic images and the way we built the regression model. In the simulated images, the adding of new objects and the updating of existing objects are random, so in some cases objects start to grow at earlier stages and in other cases they start to grow comparatively later. Consequently, the granulometric moments at a specific time differ considerably from one simulation to another, even with limited ranges for the parameters used to generate the images. Therefore the distribution of the predicted times for images from any one time point is wider than will give good classification accuracy. The coefficients of variation (CVs) for both foreground and background PS moments of the 256^2 pyramid and ellipse images are shown in Appendix II(d). The CVs are high, confirming the high variability of the PS moments.

We built the models using average granulometric moments and tested them using moments from single images, which can vary considerably from the average ones, but in practice there will only be one image available for a specific time point to be predicted. We did compare the results of classification based on modelling all the single moments from each training image available at each time point to evolution time, rather than the average moment from these images. Sometimes this was better, sometimes worse, but in general no real improvement was obtained in prediction accuracy using single moments rather than average ones.

Here we have evaluated our methodology of predicting evolution time of evolving synthetic texture images. Our methodology builds on and extends previous work to predict evolving time of corrosion images in Gray et al. (2006), Gray et al. (2005) and McKenzie et al. (2003). They also used a regression approach to relate granulometric moments to evolution time, but a key difference between the two methodologies is that they used multiple regression to relate moments to the underlying parameters used to generate the synthetic images, as the evolution of the artificial images depended explicitly on some evolution parameters, e.g. mean grain size, sd of grain size and number of grains, which were set up as a known function of time before generating the images. In our case the parameters are set at the beginning of the evolution period and do not change with time, so we developed a more generalisable approach of relating granulometric moments directly to evolution time. Relating moments to time directly makes more sense

for our synthetic images, as the parameters used to generate the images do not relate directly or explicitly to time.

The earlier work used the artificial image-based model to predict the evolution time of a new image, using the observed granulometric moments from that image. We developed and evaluated the model using synthetic images, but to apply it in practice the same approach is used on real training images to determine the appropriate model for test images of the same kind of texture, so our approach should be more robust.

In Chapter 6 we now apply the same methodology to real images of corrosion to classify them according to evolution time.

Chapter 6

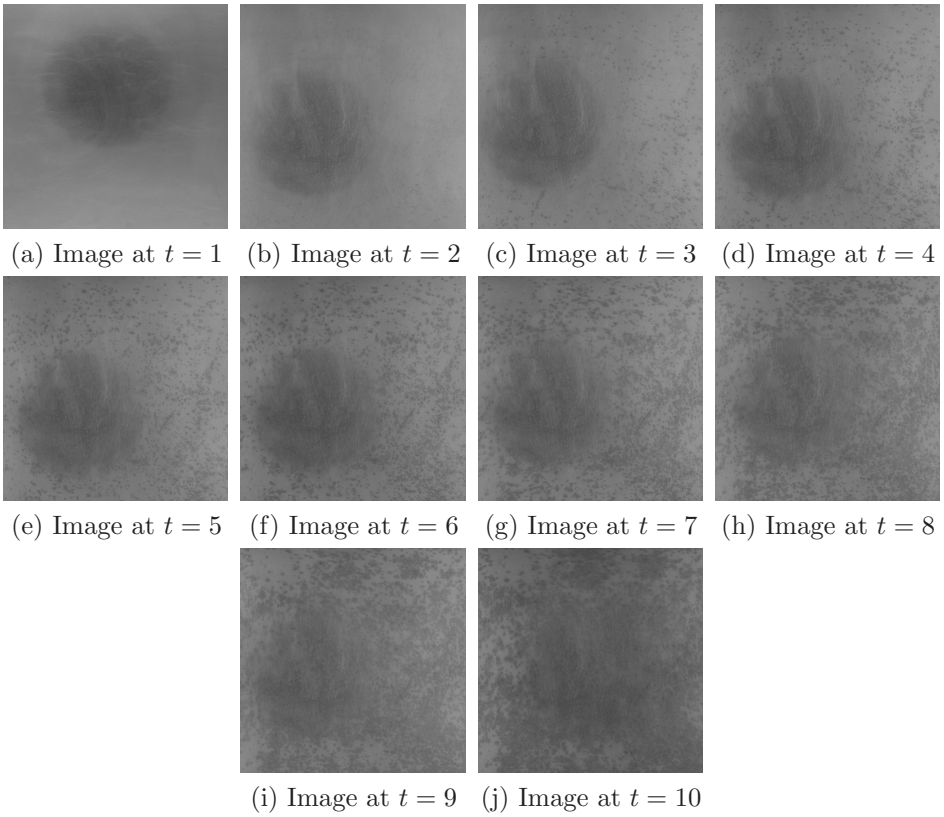
Classification of Corrosion Images

6.1 Corrosion Images

The corrosion images used here were also used in McKenzie et al. (2003), Gray et al. (2005) and Gray et al. (2006) where they described the development of parallel evolution functions as a method for texture classification of evolving textures, using granulometric features from these corrosion images. The images were generated in a laboratory setting by Jennifer McKenzie, for her PhD thesis (McKenzie (2004)). These are images of a steel plate, sprayed regularly with a dilute saline solution and left over a period of 10 days to corrode over time. Images were captured on a daily basis, starting with no corrosion on the first day of the experiment ($t = 1$) to the 10^{th} day ($t = 10$) when the plate was almost completely covered with corrosion texture and saved in TIFF image file format.

All images are of size 1400^2. The original colour images were converted to grey scale, as colour was found in McKenzie (2004) not to provide useful information for time classification of these images. A sub-set of these images is shown in Figure 6.1. The blurred area in the images results from reflection from the camera used to capture the images.

First we extracted sample images of size 256^2 from each of the 1400^2 grey scale images. A total of 10 non-overlapping sub-images were extracted from the bottom and right side of each images, to avoid the blurred area. So there are a total of 100 sample images, 10 for each time point ($t = 1, 2, \ldots, 10$). A sub-set of the sub-images is shown in Figure 6.2.



(a) Image at $t = 1$ (b) Image at $t = 2$ (c) Image at $t = 3$ (d) Image at $t = 4$

(e) Image at $t = 5$ (f) Image at $t = 6$ (g) Image at $t = 7$ (h) Image at $t = 8$

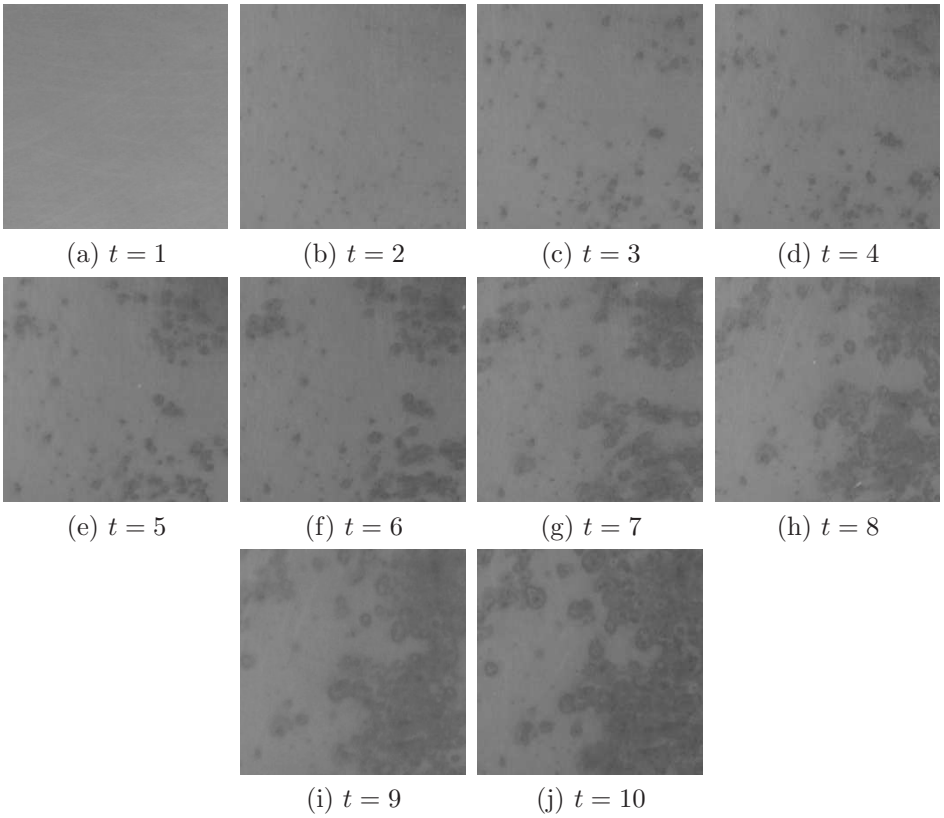
(i) Image at $t = 9$ (j) Image at $t = 10$

Figure 6.1: Grey scale corrosion images of size 1400^2 taken at 10 different time points.

6.2 Granulometry on Binary Corrosion Images

Initially we applied our methodology on the binary version of these images, applying granulometry using square and disk SEs. Although Otsu's thresholding provides optimum global thresholding (Gonzalez and Woods (2008)), in this case it does not work well to distinguish the blob-like texture from the background of the image. Hence, we selected the threshold empirically by experimenting with a wide range of values between 0 to 1 and it was found that the threshold value 0.45 emphasised the texture best. Still it is not very accurate, as in some cases it classified the image area as texture where there is no texture present at that location.

The binary versions of the images and the PS of the binary images using a disk SE are displayed in Figure 6.3. In the binary images, corrosion spots are presented as white and the background as black. The corrosion spots are not well preserved in the corresponding binary versions, especially at the top right corners after $t = 3$. From time $t = 6$ the right side of the images are classified as corroded regions but they are not fully corroded. For $t = 1$, there was no corrosion spot,



(a) $t = 1$ (b) $t = 2$ (c) $t = 3$ (d) $t = 4$

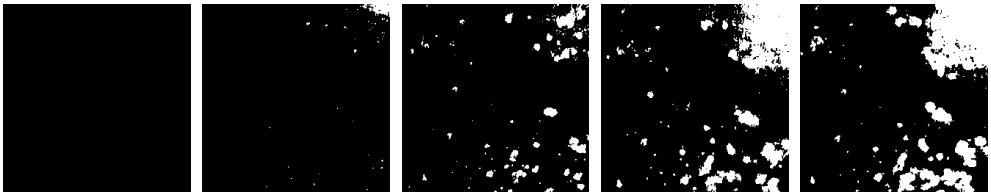
(e) $t = 5$ (f) $t = 6$ (g) $t = 7$ (h) $t = 8$

(i) $t = 9$ (j) $t = 10$

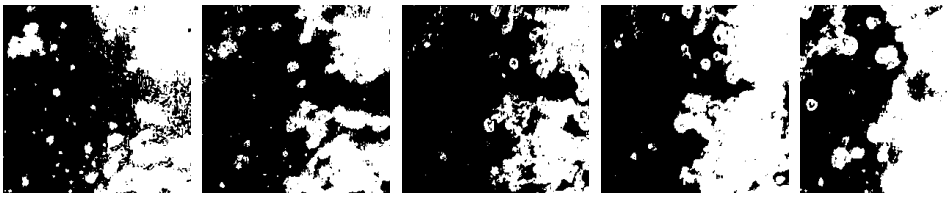
Figure 6.2: A sub-set of the extracted 256^2 grey scale corrosion images, one from each time point $t = 1$ to 10.

hence its binary version is completely blank, i.e. there is no object pixel which can be removed by opening granulometry and as a result a null PS is obtained. In general the PSs do not provide useful information regarding the shapes and sizes of the corrosion spots.

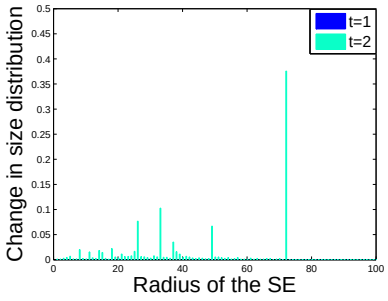
Granulometry using square and disk SEs separately was applied to the foreground and background of the thresholded images, and the first four PS moments from each SE were obtained, so using all of these moments for each image the moments data give a 100×8 matrix. Average moments were computed over the 10 sub-images at each time point. So the average moment data consists of 10 rows, one for each time and 8 columns from the four moments for each of the two SEs. The regression approach is used to model the foreground and background average PS moments to evolution time, and the fitted model is used to predict the time for each image. We consider foreground features first.



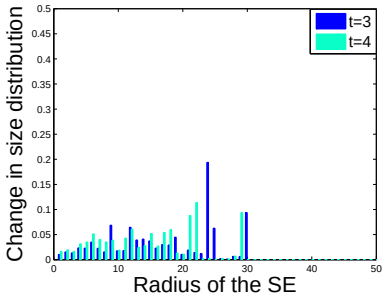
(a) $t = 1$ (b) $t = 2$ (c) $t = 3$ (d) $t = 4$ (e) $t = 5$



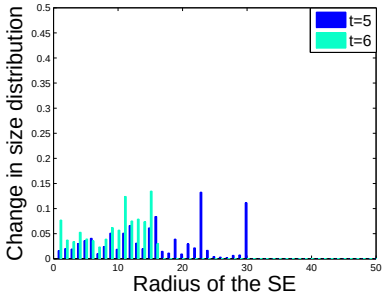
(f) $t = 6$ (g) $t = 7$ (h) $t = 8$ (i) $t = 9$ (j) $t = 10$



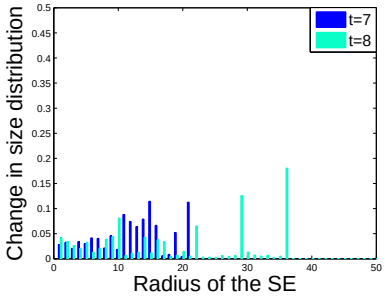
(k) PS of (a) and (b)



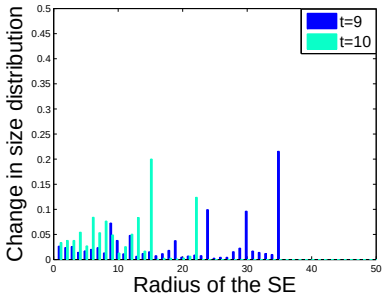
(l) PS of (c) and (d)



(m) PS of (e) and (f)



(n) PS of (i) and (j)



(o) PS of (k) and (l)

Figure 6.3: Binary form of one extracted sub-image from each of time points $t = 1$ to 10, and its PS using a disk SE.

6.2.1 Foreground PS moments of the binary images

The average foreground PS moments using the square and disk SE are plotted against time, with the best fitting cubic regression curves, in Figure 6.4. Both

mean and sd decrease with time, but are higher at time $t = 5$. Skewness from both SEs for times 4 to 10 shows an increasing trend but kurtosis does not show any clear relationship with evolution time. Therefore, only the first three moments were used in the prediction process. The dotted lines are the fitted cubic regression curves, which do not fit the data at all well. However, we are not interested in using higher order polynomial regression than cubic, and so we used the first three moments to predict time.

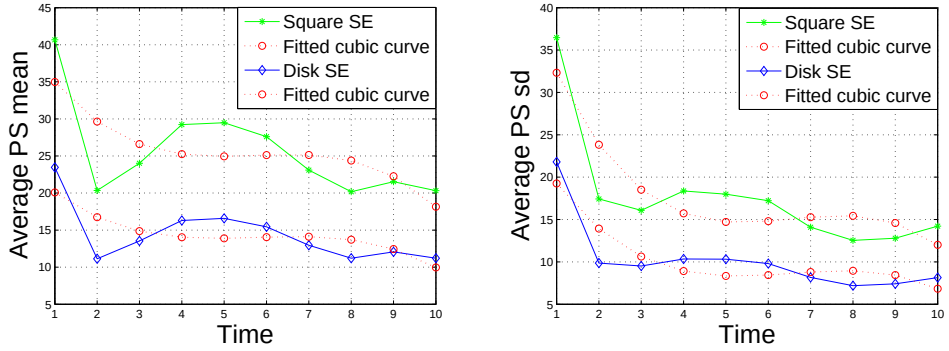
The predicted times are shown in Figure 6.5. The predicted time ranges from 1 to 11, whereas the actual time goes from 1 to 10. In general the prediction is very poor. For example there are 10 images at each time point, for which time is to be predicted, and for $t = 1$, 5 of them are predicted as time 1, 2 as time 2, 1 as time 3 and 2 as time 11. For $t = 10$, 1 image is predicted as time 2, 2 as time 3, 1 as time 9, 3 as 10 and 3 as time 11. The predicted times are either between 1 and 4 or between 9 and 11, whereas the actual time ranges from 1 to 10. None of the images were predicted between time 5 and 8. It was not clear why this is so.

This problem was not due to the root finding algorithm as there was always a real positive root, rather the algorithm found it harder to predict the time for the images in the middle stages of their evolution time. It was thought that the PS moments for the middle stage images are more variable compared to the earlier and later stages images, but sds and CVs of the PS moments did not confirm that.

6.2.2 Background PS moments of the binary images

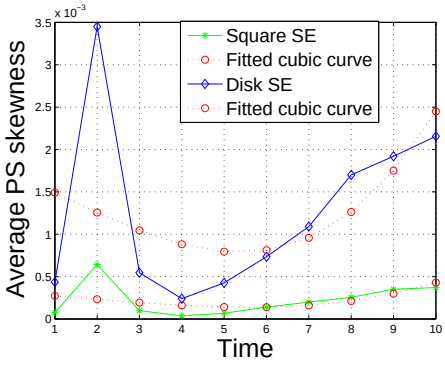
Figure 6.6 shows the average background PS moments against time with the fitted cubic curves using square and disk SEs. Neither the PS means nor the sds follow any regular pattern, as they increase with time in the earlier time points, then decrease and then increase again. Skewness from both SEs increases at the beginning and then decreases with time. Kurtosis decreases with time for both SEs. None of the cubic curves fitted very well. However, we use them all in the regression approach to predict evolution time.

The histograms of the predicted times are shown in Figure 6.7. The predicted times are worse than using the foreground moments (Figure 6.7). Again the predicted times are either between 1 and 3 or between 9 and 11, as for the foreground moments. Performance of the foreground and background moments for predicting time is computed in terms of error rates and MAEs, shown in Table 6.1. The foreground moments are better than the background moments

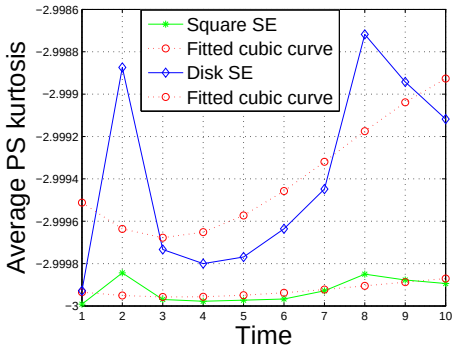


(a) Average PS mean

(b) Average PS sd

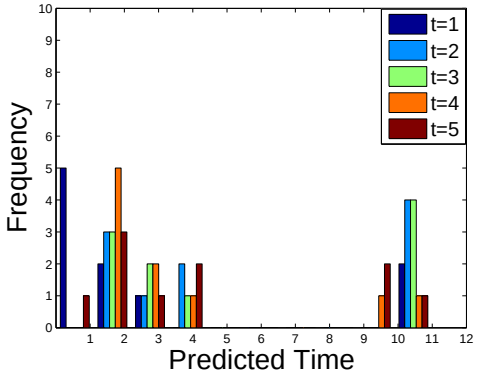


(c) Average PS skewness

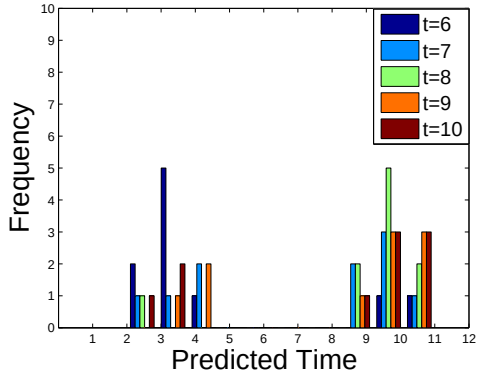


(d) Average PS kurtosis

Figure 6.4: Plots of the first four average foreground PS moments against time, for the binary corrosion images using square and disk SEs, with fitted cubic curves (dotted lines).



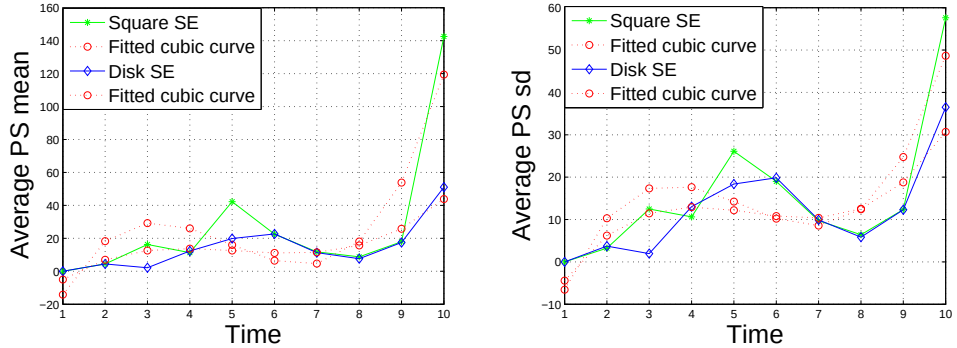
(a) Actual time t = 1 to 5



(b) Actual time t = 6 to 10

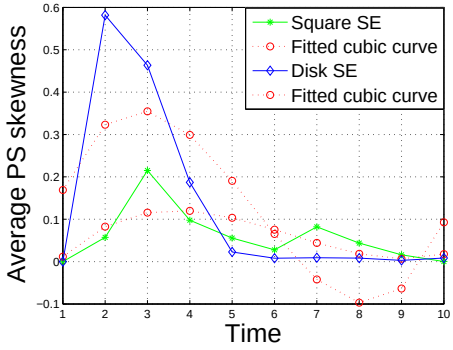
Figure 6.5: Frequency histograms of predicted time using the first three foreground PS moments of the binary corrosion images from square and disk SEs, using cubic regression, for all sub-images at each time point 1-5 (a) and 6-10 (b).

both in terms of error rates and MAEs, but both are very poor.

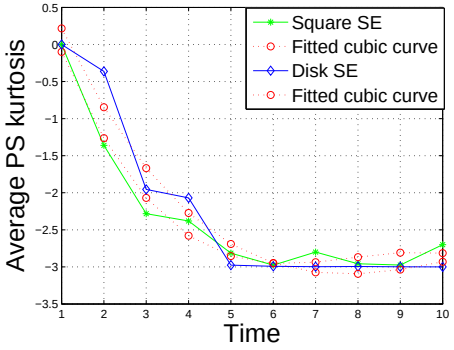


(a) Average PS mean

(b) Average PS sd

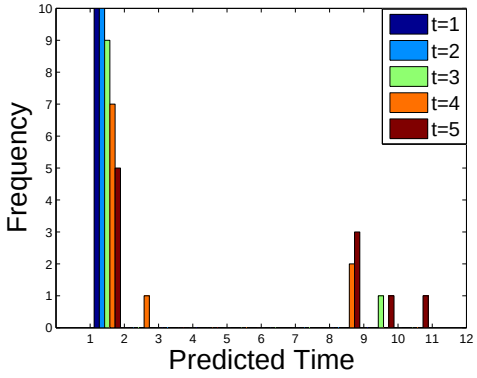


(c) Average PS skewness

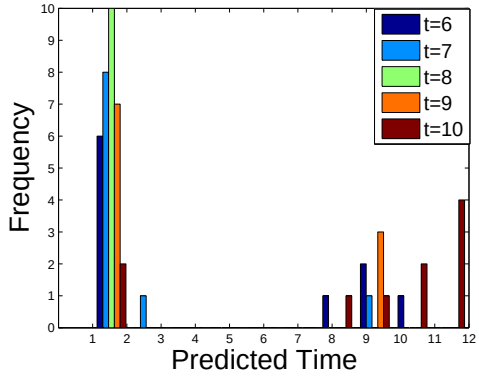


(d) Average PS kurtosis

Figure 6.6: Plots of the first four average background PS moments against time, for the binary corrosion images using square and disk SEs, with fitted cubic curves (dotted lines).



(a) Actual time t = 1 to 5



(b) Actual time t = 6 to 10

Figure 6.7: Frequency histograms of predicted time using the first four background PS moments of the binary corrosion images from square and disk SEs, using cubic regression, for all sub-images in each time point 1-5 (a) and 6-10 (b).

6.2.3 Other classifiers on the binary images

For the binary corrosion images, since the foreground PS moments provide lower classification error rates than the background ones, we use these moments (3

Table 6.1: Average test set error rates for the regression approach using 6 foreground and 8 background PS moments of the binary corrosion images using all sub-images; results are averaged over 10 runs.

Time	Foreground moments				Background moments			
	MAE	Type 0	Type 1	Type 2	MAE	Type 0	Type 1	Type 2
1	2.40	0.60	0.30	0.20	11.00	1.00	1.00	1.00
2	4.00	0.70	0.60	0.40	9.50	1.00	1.00	1.00
3	3.30	0.80	0.40	0.40	8.00	1.00	0.90	0.90
4	2.30	0.90	0.70	0.20	6.60	1.00	1.00	0.90
5	3.10	1.00	0.80	0.70	5.40	1.00	0.90	0.90
6	3.40	1.00	1.00	0.90	5.10	1.00	1.00	0.90
7	3.20	1.00	1.00	0.80	4.20	1.00	1.00	0.90
8	2.40	1.00	0.80	0.30	3.10	1.00	1.00	0.80
9	2.50	0.90	0.60	0.30	3.60	0.90	0.90	0.50
10	2.60	0.70	0.30	0.30	4.30	1.00	0.80	0.70
Overall	2.92	0.86	0.65	0.45	6.08	0.99	0.95	0.16

from the square and 3 from the disk) in the other classifiers, i.e. SVM, LDA and FF-NNET, and compare their performance with that of the regression approach.

For SVM, the different kernels and associated parameter values were tested using a grid search approach for a single training set consisting of 70% of the moments. Table 6.2 shows training set error rates for SVM using different kernels. For the radial basis kernel and the polynomial kernel the error rate was computed for different values of γ between 0.1 and 1 in steps of 0.1 (Table 6.2 shows error rates corresponding to three different values of γ with different costs, as the rest were very similar). A radial basis kernel with γ between 0.1 and 0.5 and any cost between 1 and 100 produced 3% error, and $\gamma = 0.9$ with any cost between 1 and 100 produced a higher error rate (10%). A polynomial kernel produced its lowest error rate of 10% with any cost between 50 and 100 and $\gamma = 0.1$. Other choices of parameter values produced higher error. Different values of η (the degree of polynomial) between 1 to 5 were experimented with and it was found that $\eta = 5$ yielded the lowest error rates. A linear kernel with any cost of 10 or above produced 100% correct classification. Therefore we used a linear kernel with cost of 100 for this feature set.

The data were normalised before using FF-NNET, to make FF-NNET comparable to SVM, and the values of decay and rang for FF-NNET were set to 10^{-4} and $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, respectively. Ten hidden units were used as this produced the lowest training set error rate (Table 6.3) and was the

largest number possible for these data.

Table 6.2: Training set error rates for different combinations of cost and γ for the radial basis kernel, the polynomial kernel with $\eta = 5$ and a linear kernel using 6 foreground moments from the binary corrosion images. The bold figure represents the kernel used in the final computation.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.5	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.9	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
	Polynomial kernel with $\eta = 5$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.70	0.30	0.13	0.10	0.13	0.10	0.10	0.10	0.10	0.10	0.10
0.5	0.53	0.23	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
0.9	0.57	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
	Linear kernel										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
	0.3	0	0	0	0	0	0	0	0	0	**0**

Table 6.3: Training set error rates from a grid search approach for finding optimum number of hidden neurons for FF-NNET using 6 foreground moments from the binary corrosion images.

	Number of units									
	1	2	3	4	5	6	7	8	9	10
Error rate	0.33	0.20	0.07	0.07	0	0	0	0	0	**0**

Table 6.4 shows average test set error rates and MAEs for all classifiers. SVM with a linear kernel produced only a 3% error rate, whereas the error rates for LDA and FF-NNET were 80% and 83% respectively. REG had an error rate of 86% (Table 6.1). MAEs are different from type 0 error for REG, LDA and FF-NNET as some predicted times were more than one unit away from the actual time, but this was not the case for SVM.

Table 6.4: Average test set error rates for SVM, LDA and FF-NNET using 6 foreground PS moments from the binary corrosion images using all sub-images; results are averaged over 10 runs.

Class	SVM				LDA			
	MAE	Type 0	Type 1	Type 2	MAE	Type 0	Type 1	Type 2
1	0.00	0.00	0.00	0.00	0.77	0.40	0.10	0.07
2	0.00	0.00	0.00	0.00	4.17	0.73	0.60	0.57
3	0.00	0.00	0.00	0.00	3.83	0.83	0.73	0.70
4	0.00	0.00	0.00	0.00	1.80	0.80	0.37	0.23
5	0.00	0.00	0.00	0.00	1.60	0.87	0.57	0.10
6	0.00	0.00	0.00	0.00	1.93	1.00	0.60	0.27
7	0.00	0.00	0.00	0.00	1.60	0.77	0.50	0.23
8	0.33	0.33	0.00	0.00	1.90	0.87	0.30	0.30
9	0.00	0.00	0.00	0.00	2.53	0.93	0.46	0.37
10	0.00	0.00	0.00	0.00	4.83	0.83	0.70	0.67
Overall	0.03	0.03	0.00	0.00	2.50	0.80	0.49	0.35

Time	FF-NNET			
	MAE	Type 0	Type 1	Type 2
1	2.17	0.53	0.33	0.27
2	4.60	0.87	0.67	0.60
3	3.33	0.73	0.70	0.50
4	2.13	0.90	0.53	0.27
5	1.90	0.93	0.57	0.30
6	1.50	0.77	0.47	0.17
7	2.23	0.90	0.73	0.33
8	3.10	0.80	0.67	0.53
9	3.30	1.00	0.70	0.57
10	5.20	0.90	0.87	0.77
Overall	2.95	0.83	0.62	0.43

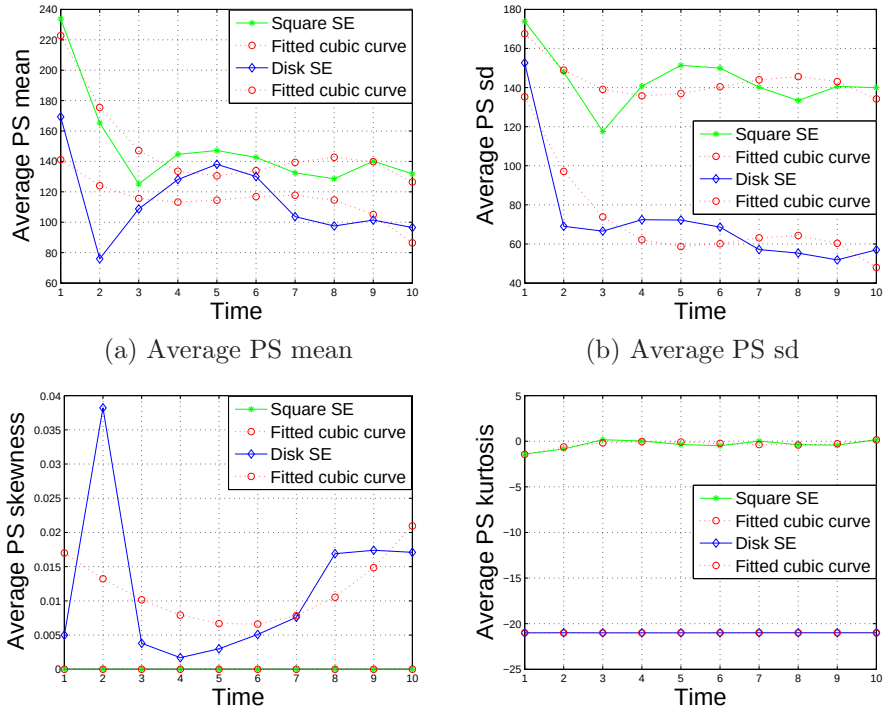
6.3 Granulometry on Grey Scale Corrosion Images

We now consider the foreground and background of the grey scale images directly. Granulometry using a square and disk SE of increasing size is applied to the foreground as well as the background of the 10 grey scale corrosion sub-images at each time point and the first four PS moments are calculated.

6.3.1 Foreground PS moments of the grey scale images

The average foreground PS moments are plotted against time in Figure 6.8. Both mean and sd decrease with time, but increase in the middle of the evolution period. The square SE produced near-zero skewness for the whole evolution period, whereas the disk produced skewness which does not follow any clear time trend. Kurtosis for the square SE is near-zero and for the disk SE it is almost constant (at -23.999). The dotted lines are the fitted cubic curves. These do not fit well to the first three moments. Although the fitted cubic has a good fit to kurtosis it will be of no use for time prediction as kurtosis is almost constant over time.

Using the first three PS moments from both SEs, we predicted the time using cubic regression, with the results shown in Figure 6.9. The predictions are poor, as before, especially for the middle states (actual time $t = 5 - 8$) of the evolution.



(a) Average PS mean

(b) Average PS sd

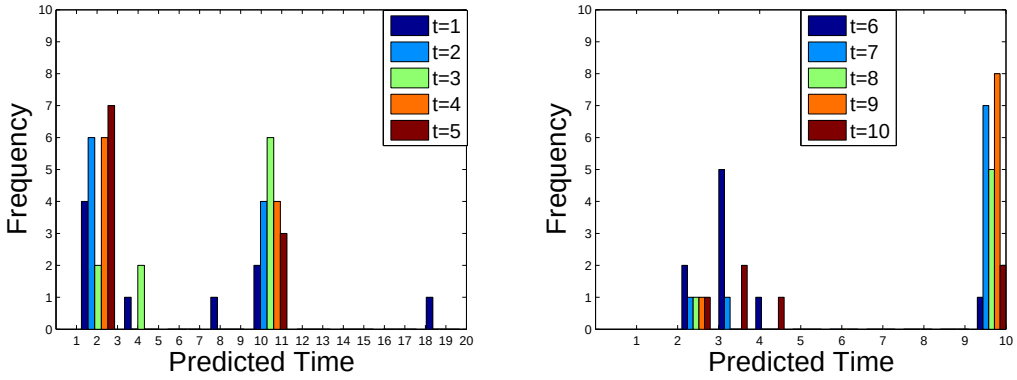
(c) Average PS skewness

(d) Average PS kurtosis

Figure 6.8: Plots of the average foreground PS moments against time, for grey scale images using square and disk SEs with the fitted cubic curves (dotted lines).

6.3.2 Background PS moments of the grey scale images

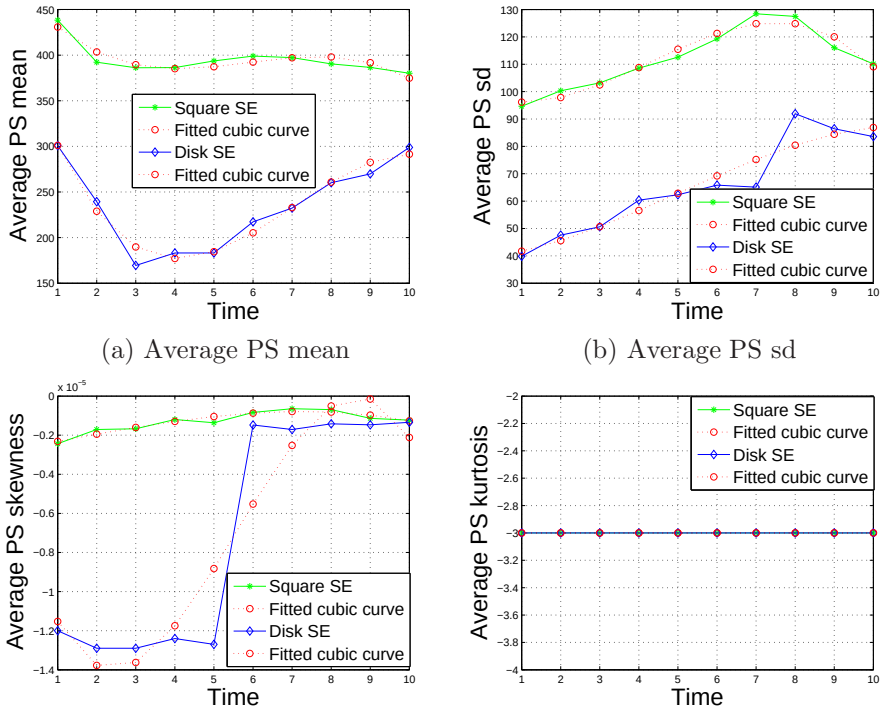
Figure 6.10 shows background PS moments against time using square and disk SEs from the grey scale corrosion images. The PS mean from the square SE



(a) Actual time t = 1 to 5 (b) Actual time t = 6 to 10

Figure 6.9: Frequency histograms of predicted time using the first three PS moments from square and disk SEs on the foreground of the grey scale corrosion images.

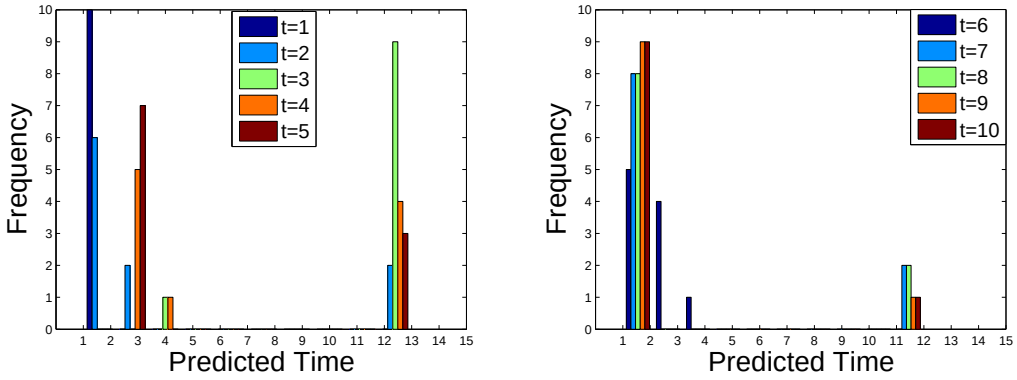
decreases slightly over time, whereas using the disk SE it decreases then increases. The sd from both SEs in general increases with time. Skewness from a square SE increases only slightly over time, whereas the increase is much more sudden for the disk SE. Kurtosis from both SEs is constant at -3. Hence, we use the first three moments from both SEs in the model building.



(a) Average PS mean (b) Average PS sd

(c) Average PS skewness (d) Average PS kurtosis

Figure 6.10: Plots of the average background PS moments against time, for grey scale images using square and disk SEs with the fitted cubic curves (dotted lines).



(a) Actual time t = 1 to 5 (b) Actual time t = 6 to 10

Figure 6.11: Frequency histograms of predicted time using the first three PS moments from square and disk SEs on the background of the grey scale corrosion images.

Figure 6.11 shows the predicted times for this moment set. These were again very disappointing, so we do not compare the performance of the regression approach with that of any other classifiers. Classification results for the foreground and background moments are summarised in Table 6.5. Foreground moments gave lower MAE, whereas background moments produced lower type 0 error. However, the MAEs and error rates are all rather high. Therefore, we look for further options to try to get improved prediction from the regression approach.

Table 6.5: Average test set error rates for the regression approach using 6 foreground and 6 background PS moments from the grey scale corrosion images, using all sub-images; results are averaged over 10 runs.

Time	Foreground moments				Background moments			
	MAE	Type 0	Type 1	Type 2	MAE	Type 0	Type 1	Type 2
1	7.00	0.90	0.60	0.60	0.10	0.10	0.00	0.00
2	4.00	1.00	0.40	0.40	2.20	0.40	0.20	0.20
3	4.60	0.80	0.60	0.60	8.90	1.00	0.90	0.90
4	3.60	1.00	0.80	0.40	3.70	0.90	0.40	0.40
5	3.10	1.00	1.00	0.50	3.60	1.00	1.00	0.30
6	3.40	1.00	1.00	0.90	3.40	1.00	1.00	0.90
7	3.40	1.00	1.00	1.00	5.00	1.00	1.00	1.00
8	2.80	1.00	1.00	0.50	5.60	1.00	1.00	1.00
9	1.70	1.00	0.20	0.10	6.60	1.00	1.00	1.00
10	3.20	0.80	0.40	0.40	7.80	1.00	1.00	0.90
Overall	3.68	0.95	0.70	0.54	4.69	0.84	0.75	0.66

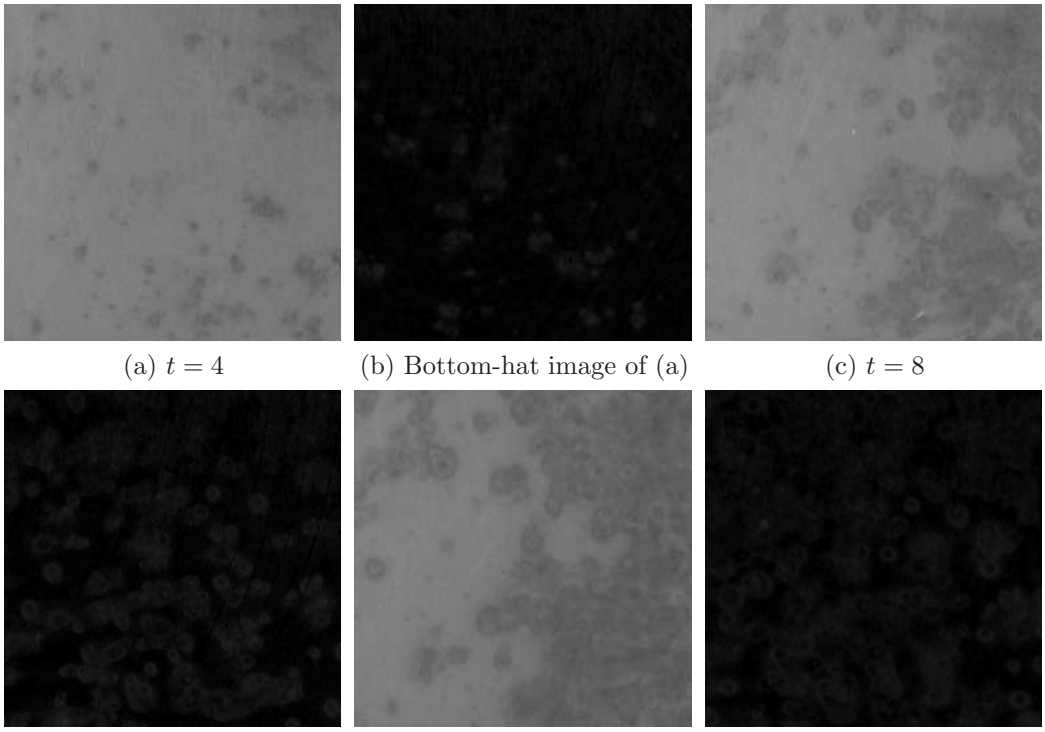
6.4 Granulometry on Transformed Corrosion Images

Pre-processing was investigated for better classification results, as the images were converted from colour images and there are a lot of intensity variations in the colour images. A hat transformation is applied at this stage for contrast enhancement of the images. In many cases hat transformation is an effective image pre-processing technique, e.g. it was used for optimum results in recognising vehicle number plates in Sulehria et al. (2007).

There are two types of hat operations, known as the top-hat and bottom-hat transformations, as discussed in Section 2.6. The top-hat operation is the result of subtraction of an opened image from the original, whereas the bottom-hat transform is defined as the closing of the image minus the original image. The top-hat operation suppresses the dark parts and highlights the bright parts of the image, and the bottom-hat operation highlights the dark features of the image. Since the texture in the corrosion images is darker than the background, the bottom-hat transformation is the appropriate one to use.

As the spots of corrosion in the images increase in size over time, it was felt that use of a disk SE of increasing size in the bottom-hat transformation would be appropriate. Disks with different radii were experimented with for each image and a suitable radius was chosen to represent properly the image objects (corrosion spots). From a wide range of radius values tested, it was found that disks of radius $6, 7, \ldots, 15$ were the most appropriate ones for times 1 through 10 respectively, as they preserved best the sizes and shapes of the textures. Once we found that a disk of radius 6 clearly represents the corrosion spots in the bottom-hat transformed image at time 1, for example, we were able to decide more quickly that a disk of radius 7 was suitable for the image at time 2 and so on. These disk sizes were determined at the training stage and were used for all images at each time point. However, in practice, if the evolution times of new test images were unknown similar exploration would be needed for each test image.

Figure 6.12 represents the grey scale corrosion images and their respective bottom-hat transformed images. Bottom-hat transformation highlighted the corrosion spots by making them lighter and the background evenly darker. Applying granulometry to these transformed images should make it possible to extract more meaningful information, leading to better classification results.



(a) $t = 4$ (b) Bottom-hat image of (a) (c) $t = 8$

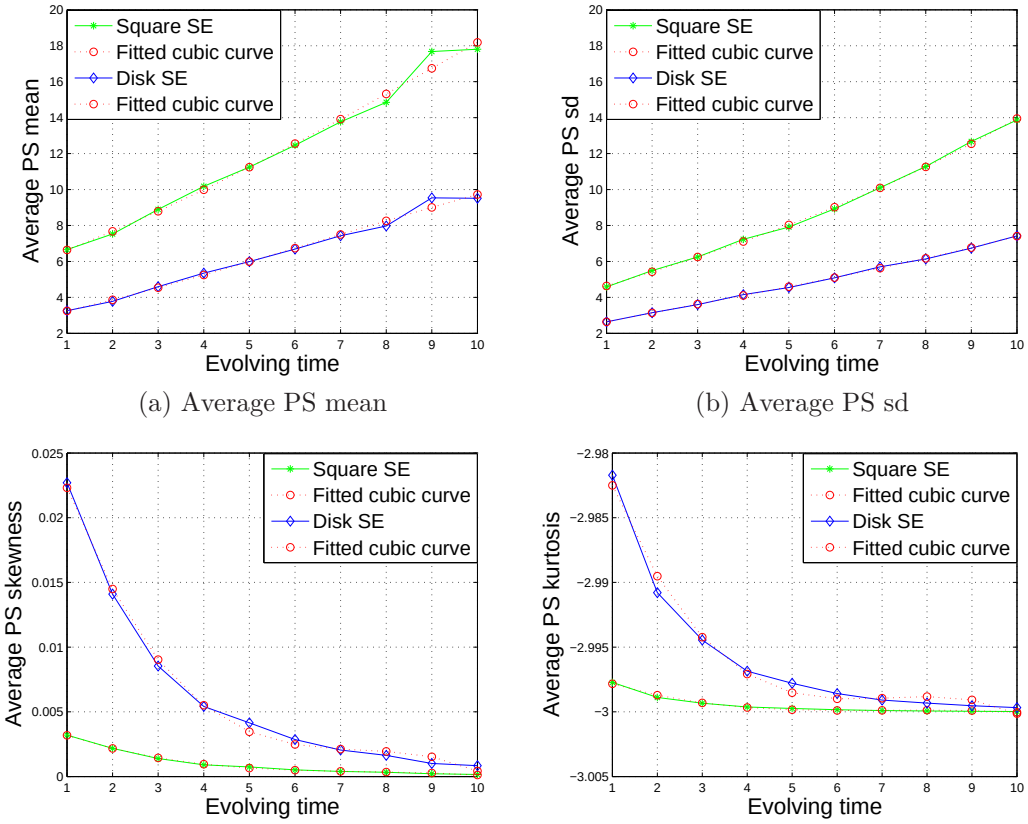
(d) Bottom-hat image of (c) (e) $t = 10$ (f) Bottom-hat image of (e)

Figure 6.12: Some corrosion sub-images of size 256^2 and their bottom-hat transformed images.

6.4.1 PS moments of the transformed images

We applied granulometry to the foreground of the bottom-hat transformed images using square and disk SEs and the first four granulometric moments of each pattern spectrum were computed for each image. The moments were then averaged over the 10 sub-images from each time point, to give a 10×8 matrix of moment data. The average moments are plotted against time in Figure 6.13, with fitted cubic curves. Average PS mean and sd using both SEs clearly increase with time, while skewness and kurtosis decrease in each case. The curves are now much smoother than for the un-transformed images, so that fitting cubics to these curves provided much better fits and should improve time prediction.

We computed the coefficients of variation (CVs) of the PS moments, which expresses the standard deviation as a percentage of the sample mean, for this dataset (see Appendix III). All 4 PS moments from both SEs have very low CVs, which reflects low variability among the moments at different time points. Using such moments we would expect good classification results. The CVs for both sets of synthetic images were very high (Appendix II(d)), and those sets of moments gave very high classification error.



(a) Average PS mean (b) Average PS sd

(c) Average PS skewness (d) Average PS kurtosis

Figure 6.13: Plots of average PS moments against evolution time, for the bottom-hat transformed corrosion images using square and disk SEs, with fitted cubic curves (dotted lines).

6.4.2 Using different classifiers on the transformed images

All 4 granulometric PS moments from both SEs are used as texture features for predicting the evolution time of the texture as they all show a strong relationship with time. Again we have implemented the regression approach as well as SVM, FF-NNET and LDA and we compare their accuracy. At this stage, for all classifiers 70% of the moments were randomly sampled and were used for training, and the rest used for testing, and the results averaged over 10 such runs.

Regression approach

A cubic polynomial regression was fitted using the first four average PS moments computed from both SEs (square and disk) to relate the PS moments to evolution time.

There are 10 sub-images for each time point and we selected 7 of them randomly to build the regression model and used 3 sub-images to predict the time

in each run. The confusion matrix of the predicted time for one such run is given in Table 6.6. The overall error rate was exactly 10% for this specific run. The regression approach correctly predicted the evolution time for all test images up to $t = 8$, 2 images were misclassified at $t = 9$ and only 1 at $t = 10$, giving 3 misclassified out of 30 test images (3 for each time).

Table 6.6: Confusion matrix of the predicted time for test set images using regression approach for the bottom-hat transformed images from a single run.

Actual time	Predicted time using regression approach									
	1	2	3	4	5	6	7	8	9	10
1	**3**	0	0	0	0	0	0	0	0	0
2	0	**3**	0	0	0	0	0	0	0	0
3	0	0	**3**	0	0	0	0	0	0	0
4	0	0	0	**3**	0	0	0	0	0	0
5	0	0	0	0	**3**	0	0	0	0	0
6	0	0	0	0	0	**3**	0	0	0	0
7	0	0	0	0	0	0	**3**	0	0	0
8	0	0	0	0	0	0	0	**3**	0	0
9	0	0	0	0	0	0	0	0	**1**	2
10	0	0	0	0	0	0	0	0	1	**2**

Then we obtained average results, by repeating the process 10 times and averaging overall error rate, type 0 error rate and MAE. Type 0 error and average MAE are identical for all time points as the misclassified times were only one unit away from the actual times (Table 6.8).

For these hat-transformed images we did not clip any predictions to the maximum observed time point of 10, so some may have been higher than that, and also some may have been predicted as 0. In the root finding algorithm, for these images, in training there were always three real roots, of which it was most appropriate to select the smallest one as the prediction.

Support vector machine

The same set of granulometric moments were used as features in SVMs. This is a multi-class classification problem, as there are 10 evolution times ($t = 1, 2, \ldots, 10$). Here the *one-to-one classification* approach is used. The theoretical aspects of SVM are described in Section 3.12, and their practical use using the R software is described in Section 5.5 and in Appendix II(a).

We examined different kernels in the SVM with a broad range of values for the cost and the kernel parameter. Table 6.7 shows error rates for a single training set using different kernels with different parameter settings. A wide range of

values of γ was tested but only few are displayed as the rest were similar. A polynomial kernel with $\eta = 1$ is the best kernel for these moments, as it gives 100% accuracy for many choices of the cost and γ. The value $\eta = 1$ was the best choice among the values 1 to 5. For example, any cost of 20 or above with any value of γ between 0.1 to 0.9 produced 100% correct classification. For the radial basis kernel, any cost between 1 and 100 and any γ in the range 0.1 to 0.9 produced 97% or better correct classification for the training set. The linear kernel is equally effective as it produced 97% or better correct classification using any cost between 1 and 100. Here we also tried a sigmoid kernel, but it was not an appropriate choice as the highest correct classification was only 50%, for cost 20 and $\gamma = 0.1$. Therefore, we used the polynomial kernel with $\eta = 1$ and a cost of 100 and $\gamma = 0.9$ in SVM, which provided 100% correct classification rate for both training and test sets.

Linear discriminant analysis

Linear discriminant analysis (LDA) is discussed in Section 3.9 and how it can be applied in the R computing package is described in Appendix II(b). As kurtosis from the square SE is constant up to 3 decimal places (i.e. -2.999), for all time points, kurtosis was not used in LDA. Using the proportions of the data in each class as (equal) prior probabilities, 70% of the moments were used to train the model and the rest was used for testing as before.

Neural network

A single hidden layer FF-NNET was used to classify the evolution time using granulometric moments. Section 3.11 contains a detailed description of the neural network and Appendix II(c) describes how to implement this in R.

The data were normalised before using FF-NNET and the parameter values were tuned empirically to produce higher classification accuracy. Firstly, a grid search approach was applied to find the optimum value of decay and rang and it was found that the best values of decay and rang were 10^{-4} and $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, respectively for this set of PS moments. Using these values we again sought for the number of hidden units to be used in the FF-NNET. Although any hidden units between 1 and 10 with the optimum value of decay and rang produced 0% training set error rate, 10 units were used in the final computation to make the classifier more generalisable. The maximum number of iterations was set to 5000, to ensure that the optimisation problem converged.

Table 6.7: Training set error rates for different combinations of cost and parameter γ for a radial basis kernel, a polynomial kernel with $\eta = 1$, a sigmoid kernel with $\eta = 1$, and a linear kernel using 8 foreground moments from the hat-transformed corrosion images. The bold figure represents the kernel and its parameter used in the final computation.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.5	0	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.9	0	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.03	0.03	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	**0**
	Sigmoid kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.6	0.53	0.50	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53
0.5	0.90	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87
0.9	0.90	0.93	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90
	Linear kernel										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
	0	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03

We computed average type 0 error and MAE for all classifiers using the first four PS moments from each SE as before. To see the effect of using fewer features for classifying these images, we also used the first two PS moments (mean and sd) from each of the square and disk SE in all classifiers, using 70% of the moments for training and the rest for training. Again we averaged the results over 10 runs. Type 0 error and MAE are identical again as none of the images were misclassified more than one unit away from its actual time.

Type 0 errors for each time and overall errors are shown in Table 6.8 for all classifiers, using all 8 PS moments and also 4 moments. SVM predicted time for all images correctly, while LDA was second best with 92% correct classification. REG was as good as the FF-NNET with 90% correct classification.

Using the reduced feature set, i.e. the PS mean and sd from each SE, the new

Table 6.8: Average test set Type 0 error rates for all classifiers using all 8 and also 4 foreground PS moments, for the corrosion images; results are averaged over 10 runs.

Time	8 moments				4 moments			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.000	0	0.000	0.000	0.000	0	0.000	0.000
2	0.000	0	0.133	0.067	0.000	0	0.133	0.000
3	0.000	0	0.000	0.000	0.000	0	0.133	0.000
4	0.000	0	0.000	0.133	0.000	0	0.200	0.167
5	0.250	0	0.133	0.067	0.250	0	0.600	0.133
6	0.083	0	0.200	0.200	0.083	0	0.133	0.200
7	0.167	0	0.000	0.067	0.167	0	0.133	0.167
8	0.000	0	0.000	0.067	0.000	0	0.133	0.167
9	0.333	0	0.267	0.267	0.333	0	0.533	0.267
10	0.167	0	0.067	0.133	0.167	0	0.267	0.133
Overall	0.100	0	0.080	0.100	0.100	0	0.227	0.123

regression classifier produced the same 10% type 0 error and SVM still gave 100% correct classification. However, for LDA and FF-NNET the type 0 error rates increased to 22.7% and 12.3% respectively (Table 6.8).

In Gray et al. (2006), McKenzie (2004), Gray et al. (2005) and McKenzie et al. (2003) these corrosion images were also classified according to their evolution time. They related granulometric moments to evolution parameters of the synthetic images, and back-prediction was used to predict the evolution time of a new image, based on the artificial image model and the observed granulometric moments from the new image. The difficulty with this is that real images of interest do not necessarily evolve in the same way as the artificial ones, so leading to higher error rates. Their method provided type 0 error as high as 48%, whereas our method produced only 10% error for classifying the same corrosion images.

The key step in getting successful results from the granulometric moments for the corrosion images was to use the bottom-hat transform with a disk that increases in size with time. We show the results for comparison, of applying bottom-hat transformation using a disk of fixed radius over times 1 to 10, rather than using increasing radii. Different radii, $7, 9, 11$ and 13, were experimented with and the results for radius 9 are shown here. Opening granulometry was applied to these bottom-hat transformed images using the square and disk SE and the first four average PS moments used in each classifier. The type 0 errors are reported in Table 6.9. Although SVM produced 100% correct classification,

the results from the other classifiers were much poorer than when the increasing disk size was used.

Table 6.9: Average test set Type 0 error rates for all classifiers using a fixed sized disk of radius 9 in the bottom-hat transform of the corrosion images, using 8 PS moments; results are averaged over 10 runs.

Time	Type 0 error			
	REG	SVM	LDA	FF-NNET
1	1.000	0	0.667	0.467
2	1.000	0	0.333	0.300
3	0.500	0	1.000	0.767
4	0.750	0	0.667	0.867
5	0.583	0	0.000	0.833
6	0.833	0	1.000	0.800
7	0.667	0	0.333	0.900
8	0.833	0	0.000	0.667
9	1.000	0	0.333	0.567
10	1.000	0	0.333	0.667
Overall	0.817	0	0.467	0.683

We compared the performance of the regression approach using different sub-images. Its classification accuracy varies if the model is built using moments from different sub-images. The lowest type 0 error (8%) was obtained by considering only the first 5 sub-images for each time point, where the images were extracted from along the bottom of the original corrosion images. The model was built using the average moments of those sub-images and tested for the individual ones. For the last 5 sub-images, extracted from the right side of the images, the error rate was 18%, but if the model was based on the average moments of all 10 sub-images, the error rate was 15%. The method worked better for the first 5 sub-images, as at each time point in this subset the sub-images are more homogeneous with respect to the size and number of spots within them. This is a disadvantage of the method.

6.4.3 Granulometry on the background images

As in the foreground images the spots of corrosion are darker than the background, in the background images the reverse is observed, so as a pre-processing technique the top-hat transformation is the best one to reduce the uneven illumination as well as highlight the brighter parts of the images. The results of

top-hat transformation of the background of the corrosion images are shown in Figure 6.14 (there is some shading in these images, although they are very dark).

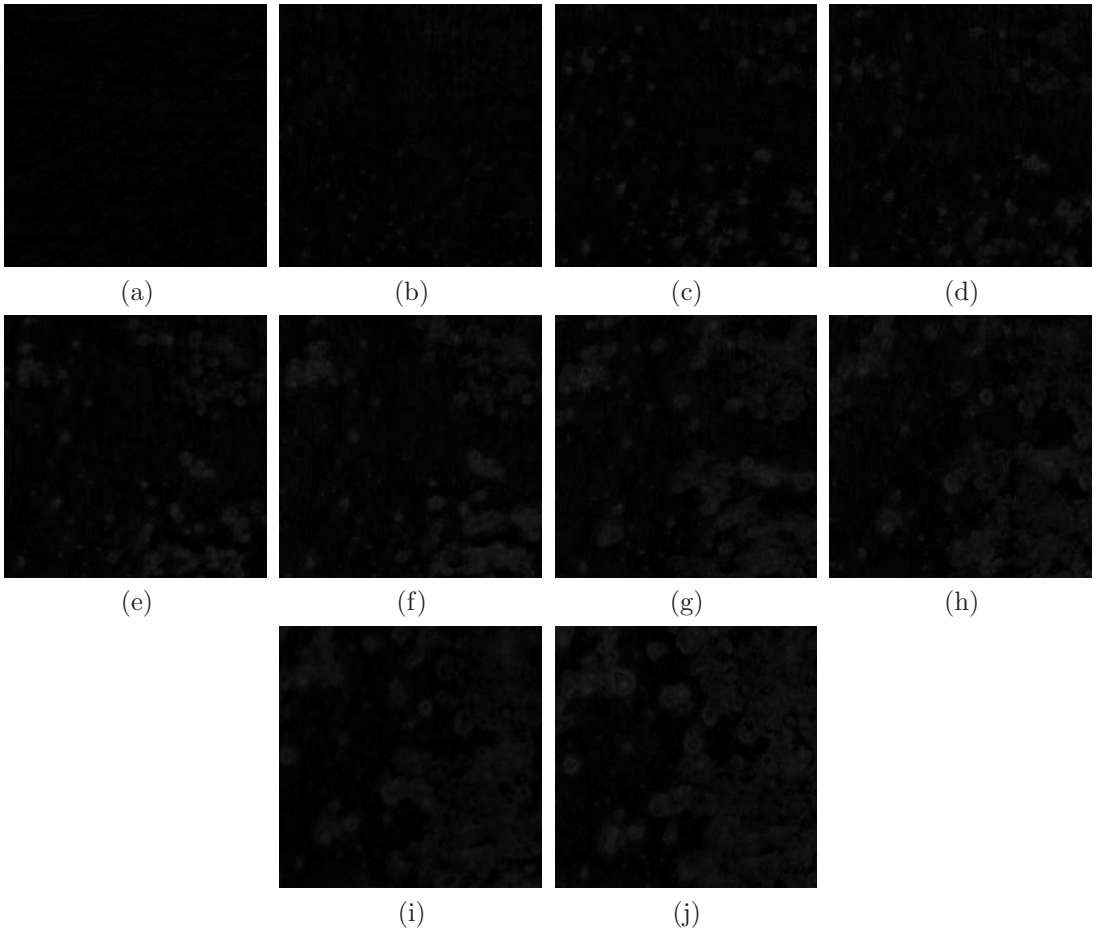


Figure 6.14: Top-hat transformed images of the background corrosion images.

Then granulometry was applied using the square and disk SEs and the first four granulometric moments were computed. Then average moments were calculated over the 10 sub-images at each time point. The top-hat transformed images look very similar to the corresponding bottom-hat transformed foreground images (see Figure 6.12), and their PS moments are nearly identical. Therefore using the PS moments from these background images is likely to produce very similar classification performance for each classifier, and so we did not proceed further with this set of moments.

6.5 Conclusion

Classification of corrosion images has rarely been addressed except in McKenzie et al. (2003), McKenzie (2004), Gray et al. (2005) and Gray et al. (2006). Some

other papers in the literature have used image analysis of corrosion images, but these all focus on distinguishing one type of corrosion from others or separating the corroded parts from the non-corroded parts of an image. None of them considered classifying corrosion images according to their evolution time. For example, Pidaparti et al. (2010) successfully classified the pits and cracks in nickel aluminium (NiAl) bronze metal images using image analysis. They considered four classes of corrosion, namely no corrosion, cracks, cracks and pits, and pits. Six sub-images of size 256^2 were extracted from scanning electron microscopy (SEM) of NiAl bronze sample images. Fractal dimensions (FD) were computed from these images and it was shown that images with no corrosion had the smallest FD, whereas pitted images had the highest FD. Cracks had the second lowest and the combination of cracks and pits had the second highest FD. Wavelet-packet decomposition was also used and Shannon entropies from each decomposed image were computed. No classifier was used to classify the pits and cracks, rather the Shannon entropy was plotted against FD, which clearly distinguished four types of corrosion, with only two cracked and pitted images misclassified as pitted. The FD and entropy allowed linear separation of the classes, so classification of pits/cracks is possible using these features.

Choi and Kim (2005) used digital image processing for analysis and classification of images of corrosion damage, namely, crevice corrosion, irregular corrosion, pitting, fretting, uniform corrosion and non-corroded images. The corroded images were characterised by colour, texture and shape features. They considered the hue, saturation and intensity planes of the colour images separately and computed five features, namely horizontal and vertical means and sds and contrast from a co-occurrence matrix. Multi-dimensional scaling was applied on the co-occurrence features to classify different corrosion images. The shape and size of pitting corrosion was analysed by digital image processing in Codaro et al. (2002). Different boundary statistics of the pits, i.e. area, elongation, roundness and perimeters, were obtained and plotted. Area provided most information about the pits and was more effective for identifying pitting corrosion.

Our regression-based classification approach using granulometric moments as texture features is a substantial improvement for classifying images of corrosion to a point in time compared to the approach in McKenzie et al. (2003), McKenzie (2004), Gray et al. (2005) and Gray et al. (2006), where the lowest classification error for the corrosion images was as high as 48% using the maximum likelihood classifier. Our method produced only 10% error for classifying the same images, moreover SVM produced 100% correct classification.

Increasing the radius of the disk SE in the hat transform of the images is of crucial importance, as granulometric features computed from hat-transformed images obtained using the same size disk SE over all time points or classes produced very high classification error for all classifiers.

In the next chapter we apply the same methodology on images of tea granules of increasing size to classify them according to their granule size.

Chapter 7

Analysing Images of Tea Granules

In this chapter we now use the same methodology on a different set of real images, which are textured images of Indian black tea granules. The accurate sorting of tea leaves is an important stage in the manufacturing process in the tea industry, and applications of image texture analysis have proven advantageous in Borah et al. (2007) and Li et al. (2008a). Image processing has numerous uses for online inspection and monitoring in the food industries in general (Chen et al. (2008b), Bhattacharyya et al. (2007), Borah and Bhuyan (2003), Wu et al. (2008), Wu et al. (2007b), and Zenoozian and Devahasti (2009)).

Sorting has traditionally been carried out by sieving with a series of sieves of differently sized mesh, however more recently computer vision approaches have been investigated for a more automated approach (Borah et al. (2007)) with a view to online monitoring.

Borah et al. (2007) consider images of cut-tear-curl (CTC) black tea granules of 'even' appearance, comprising eight grades of tea from different tea gardens in Assam, India, and sorted by granule size in terms of approximate diameter in mm. They use automated texture classification using Daubechies wavelet (Daubechies (1988)) features in MLP and LVQ neural networks, implemented in Matlab, and achieve correct classification rates of 74.67% for MLP and 80% for LVQ for 150 test images. This study is part of a larger body of work addressing tea quality assessment by automated means. In this chapter we use granulometric features and various classifiers on the same images used by Borah et al. (2007), with improved results.

7.1 Background and Related Work

Image analysis and pattern recognition techniques have been used for assessment of tea quality, classifying different types of tea and also to classify teas to different geographical regions. Tea quality is mainly determined by flavour, aroma, colour and strength (Bhattacharyya et al. (2007)). Bhuyan and Borah (2001) describe use of an electronic nose to assess tea aroma and flavour, traditionally assessed by a human sensory panel. Fermentation is one of several stages in the manufacturing of black tea and the most important one in determining the quality of brewed tea liquor. The optimum fermentation time for tea quality is the time at which the tea colour changes from green to a deep coppery red (Borah and Bhuyan (2003)). Aroma also changes from a grassy smell to a floral smell (Bhattacharyya et al. (2007)). Under- or over-fermentation adversely affects flavour. This point is traditionally detected by human observation, which is subjective and likely to vary between observers. Alternatively, indirect colorimetric assessment may be carried out, involving chemical analysis of samples collected at intervals during fermentation and boiled to produce tea extract. Output from a colorimeter produces a profile representing optical absorbance against time. Once this profile peaks for the second time, optimal fermentation has been reached, corresponding to the change to the deep copper colour in the fermenting tea (Bhattacharyya et al. (2007)).

Borah and Bhuyan (2003) describe a non-destructive, machine vision system for colour matching using the Hue-Saturation-Intensity colour model for illumination invariance. This system used a histogram-based comparison of the hue plane of an input image of fermenting tea leaves with the hue plane of two reference images from a database of ten standard images judged by a sensory panel as being at the optimum point of fermentation. A simple threshold-based method using histogram intersection (Swain and Ballard (1991)) was used to classify the input image, with 90% success in a sample of ten test images. Borah et al. (2002) used a neural network for colour matching. A two-layer perceptron, using hue and saturation of colour images of tea leaves as the input features gave 94% successful classification for 50 test images of tea granules, judged by a sensory panel on the basis of strength of the made tea.

Bhattacharyya et al. (2007) and Dutta et al. (1994) also consider detection of optimum fermentation time, but based on electronic nose-based monitoring of volatile emissions during fermentation and detection of peaks in the sensor output, rather than by detection of change in aroma by human supervisors. Bhattacharyya et al. (2007) describe identification of eight sensors for use in

an electronic nose used for monitoring fermentation. Principal component analysis (PCA) was used on readings from each sensor at regular intervals, and the first principal component was found to indicate more clearly than any one sensor individually the nature of the peaks in recorded emissions. As in the colorimetric curve, the position of the second peak again indicates the point at which optimum fermentation has been reached and the fermentation process should be stopped. The times indicated by the PCA accurately matched the optimal fermentation times indicated by human expert assessment in all trials, and in most cases matched the colorimetric assessment as well.

Li et al. (2008b) used wavelet-based image features for classification of five Chinese green teas using multi-spectral images. These tea types are not obviously distinguished by tea granule size, so are not ordered textures, and the shapes in the image textures are quite different from Indian black tea granule images, as the green tea leaves are longer and thinner. The shade of the green colour of the tea leaves also varies between tea types. Li et al. (2008b) used variable selection to select 11 grey level co-occurrence features computed from wavelet decompositions of each of the three image colour planes separately. The chosen features are used in linear discriminant analysis with 100% accurate classification for all 250 test and training images.

In Wu et al. (2007) and Wu et al. (2008) four different categories of Chinese green tea types were discriminated based on multi-spectral images. They used the entropy values for 320 three channel (R, NIR (near infrared) and G) images, 80 from each category, as a feature set in a least squares support vector machine (LS-SVM) and achieved 97.5% classification accuracy. The accuracy was further improved to 100% by subtracting one channel image from the others, namely R-NIR, R-G, NIR-R, NIR-G, G-R and G-NIR, so there were 6 images for each of the 320 sample images. Again entropy values were computed, PCA was applied and the first 6 PCs together expressed 100% of the variation in the feature set. A LS-SVM based on the 6 PCs was 99.6% accurate in discriminating the four green tea types.

In Borah et al. (2007), starting with 160 images, 20 from each of 8 grades of black tea, four-level pyramidal decomposition using Daubechies wavelets was applied to each image, giving four sub-band images from four levels of resolution. Only the approximation sub-band image was used for feature extraction. Initially they computed four features, i.e. mean, variance, entropy and energy, from each of the four sub-band images from the four levels of resolution. Using only energy and entropy as features, PCA and the clustering technique 'self organising maps'

(SOMs) were unable to make definite clustering of the data. Therefore they employed Mahalanobis distance as a measure of dissimilarity among the images. Mahalanobis distance was calculated using the features from each image in each class with respect to the rest of the images in that class. Image pairs with the smallest Mahalanobis distance were considered to be the most similar images within a given class, and the statistical features of either of these was calculated. Then the Mahalanobis distance of a test image was calculated with respect to each of the eight selected images from the different classes, and these distances were used as features for texture classification. With this new feature set PCA and SOM were able to distinguish the grades better than the previous feature set. Two different neural networks, a multi-layer perceptron (MLP) and learning vector quantisation (LVQ), were trained using 700 images and tested on another 150 images, and achieved 74.67% and 80% classification accuracy respectively.

Chen et al. (2009b) differentiated 8 different Chinese teas based on the tea content in terms of 15 metals. LDA and a BP-ANN both differentiated them with 100% accuracy. The amount of catechin and caffeine, measured simultaneously by high performance liquid chromatography (HPLC), was used to identify the quality level of four different green teas in Chen et al. (2008a). The identification rates for SVM, BP-ANN and LDA were 90%, 75% and 80% respectively. In Chen et al. (2008b) a discrete cosine transform of multi-spectral images (red, near-infrared (NIR) and green bands) of six different classes of teas was used. Using the standard deviation (sd) of each of the original or filtered NIR images with a SVM produced 73.33% or 100% correct classification respectively. Five varieties of Chinese green teas were distinguished using statistical moments and spectral measurements as features in Chen et al. (2008c). LDA using the first 11 PCs produced 98.33% correct identification. Using spectral differences of green, black and Oolong teas in the NIR region as input, SVM produced 100% classification accuracy for black tea, 95% for green tea and 90% for Oolong tea respectively in Chen et al. (2007) and Zhao et al. (2006). Chen et al. (2006) carried out a qualitative and quantitative study of 4 different types of teas using NIR spectroscopy with soft independent modelling of class analogy (SIMCA). Separate SIMCA models were built for each variety of tea based on PCs, which provided 80% correct identification for Biluochun tea and 100% for the other tea categories.

Four different grades of Chinese green teas were identified by means of HPLC in Li and He (2008). PCA was applied to wavelet-based features and 8 PCs were used in an ANN with 77.3% accuracy. Five different types of tea namely, white, green, black, Oolong and Pu-erh, were classified according to their total mineral

content in terms of 14 chemicals by McKenzie et al. (2010). LDA produced 100% correct classification for Oolong tea, 93% for Pu-erh and white teas and 86% and 64% for black and green teas. A probabilistic NNET gave 100% correct classification for all varieties except Pu-erh (93%). Li et al. (2011) applied a 2-D DWT at two different resolutions on multi-spectral images of 8 types of Chinese tea and then computed 5 GLCM features, namely energy, entropy, contrast, correlation and homogeneity, from each sub-band image, so 35 features were computed for each grey scale image and each sample consists of 3 grey scale images corresponding to wavelengths 540nm, 670nm and 800nm. Hence, 105 features were computed from each sample multi-spectral image. A LS-SVM was able to reach up to 96.82% correct classification.

Measurements of 8 different chemical descriptors were computed from 48 samples of green, black and Oolong teas in Herrador and González (2001). PCA was applied to explore the discrimination abilities of these chemicals and then LDA and BP-ANN were applied to the original metal contents with 90% and 95% correct classification. LDA was used on different chemical compositions to differentiate two classes of green and black tea with 100% success in Valera et al. (1996). PCA and partial least squares analyses of near infrared spectroscopic (NIRS) data were able to discriminate different categories of partially fermented tea samples from Taiwan with 94% to 99% accuracy in Liu et al. (2010). Ivanciuc (2003) used 3 metal contents to classify black and green teas in a SVM. Different kernels and kernel parameters were experimented with and the radial basis kernel was found to be the best, giving 84% correct classification.

Some authors distinguished different varieties of teas according to geographic areas of origin. For example, Chen et al. (2009a) proposed classifying roast green teas according to geographic region by a Fourier Transform near-infrared (FT-NIR) technique coupled with LDA, k-nearest-neighbour (KNN) classification, back-propagation artificial neural networks (BP-ANN) and SVMs. Correct classification rates for LDA, KNN, BP-ANN and SVM were 92%, 96.3%, 96.3% and 100%. Moreda-Piñeiro et al. (2003) classified 85 samples of teas from different parts of the world (36 from Asia, 18 from Africa, 24 commercial blends and 7 of unknown region) according to their region of origin using trace metal contents. PCA and cluster analysis were employed as exploratory techniques. LDA identified all African tea samples correctly but 1 sample was misclassified from the Asian group giving 94.4% correct classification. Correct classification rates using SIMCA were 100% and 91.7% for Asian and African teas respectively. Fernández-Cáceres et al. (2001) differentiated tea varieties from different

geographical regions based on levels of 12 metals. The discrimination abilities of LDA and a BP-ANN were 93.5% and 95.6% respectively.

7.2 Description of the Tea Images

Initially we were supplied with 21 tea leaf images in JPG image file format. Some of them contain granules of the same size. The images were sorted visually according to their grain size and examples of the 8 classes distinguished in Borah et al. (2007) are shown in Figure 7.1, from class 1, for the smallest granule size, to class 8, for the largest granule size. Several images were available from all except one class (8), which only contained one image.

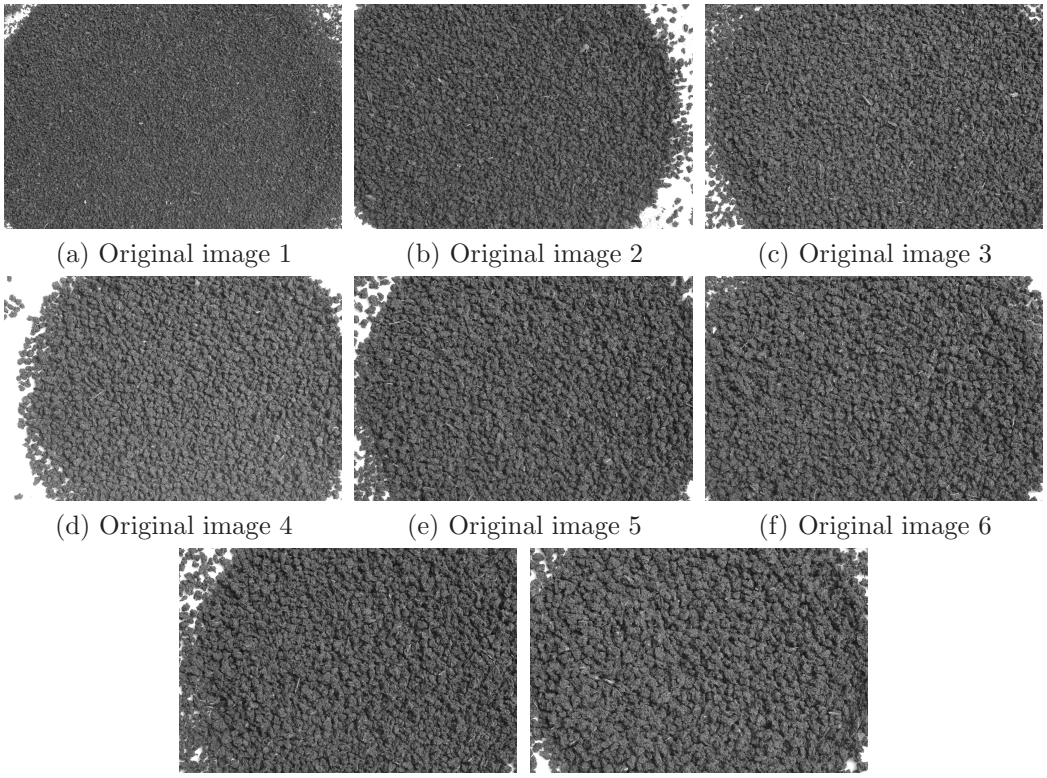
Each image is a colour image of size 2000×3008. The eight different classes mentioned in Borah et al. (2007) are Dust, OF (Orange Fannings), PD (Pekoe Dust), PF (Pekoe Fannings), BP (Broken Pekoe), BOPSM (Broken Orange Pekoe Small), BOP (Broken Orange Pekoe) and BOPL (Broken Orange Pekoe Large). The approximate diameter (in mm) of Dust is not specified and the rest have approximate granule diameter sizes of 0.25, 0.355, 0.5, 1.0, 1.3, 1.7 and 2.0 mm respectively.

Here we investigate the usefulness of our methodology for classifying these images. To obtain training and test sets, images of size 256^2 were extracted from each of the original images in Figure 7.1 representing each class, and were converted to grey scale. A total of 50 non-overlapping sub-images were extracted from one image from each class, avoiding the edges which are not fully covered by the tea leaves, giving a total of 400 sample images. One such grey scale image from each class is displayed in Figure 7.2.

It is clear from Figures 7.1 and 7.2 that tea granule size increases over the classes, so we can consider the tea images as *ordered* textures, and the same methodology can be used as for the evolving texture images considered in earlier chapters. We define ordered textures as ones which can be ranked according to some scale of fineness or coarseness, or by size of texture primitives (shapes) in the images. In this case the tea granules increase in size between the classes.

7.3 Thresholded Tea Images

Initially, we obtained binary versions of the sub-images using *Otsu's thresholding algorithm* (see Section 1.2). The tea granules became white and the background black (Figure 7.3).

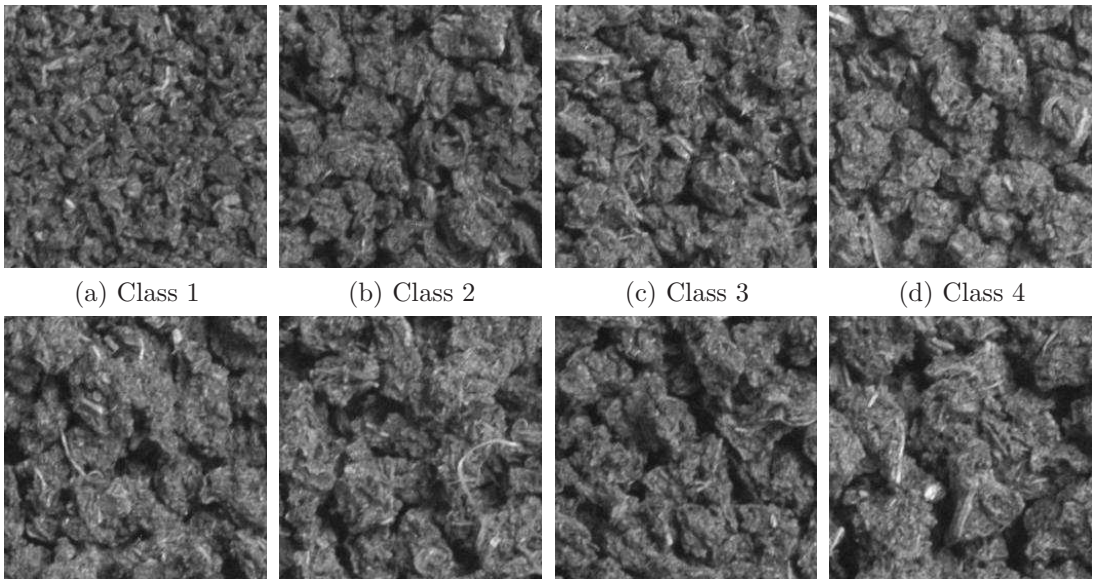


(a) Original image 1 (b) Original image 2 (c) Original image 3

(d) Original image 4 (e) Original image 5 (f) Original image 6

(g) Original image 7 (h) Original image 8

Figure 7.1: Original colour tea images with different granule sizes, labelled here as class 1 to 8.



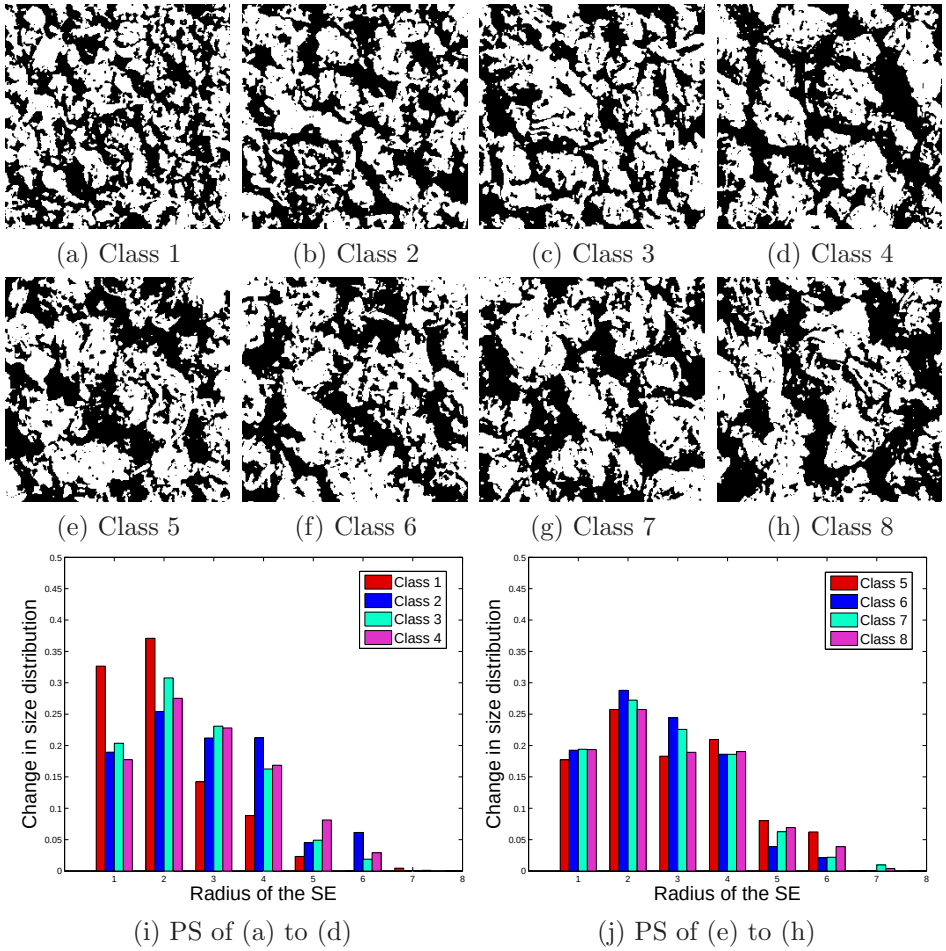
(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

(e) Class 5 (f) Class 6 (g) Class 7 (h) Class 8

Figure 7.2: Grey scale sample 256^2 tea images, one from each of the eight classes.

7.3.1 Granulometries on the thresholded images

Granulometries were then applied to the foreground of each of the 50 binary images from each class, using four different SEs, namely a square, a disk, horizontal



(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

(e) Class 5 (f) Class 6 (g) Class 7 (h) Class 8

(i) PS of (a) to (d) (j) PS of (e) to (h)

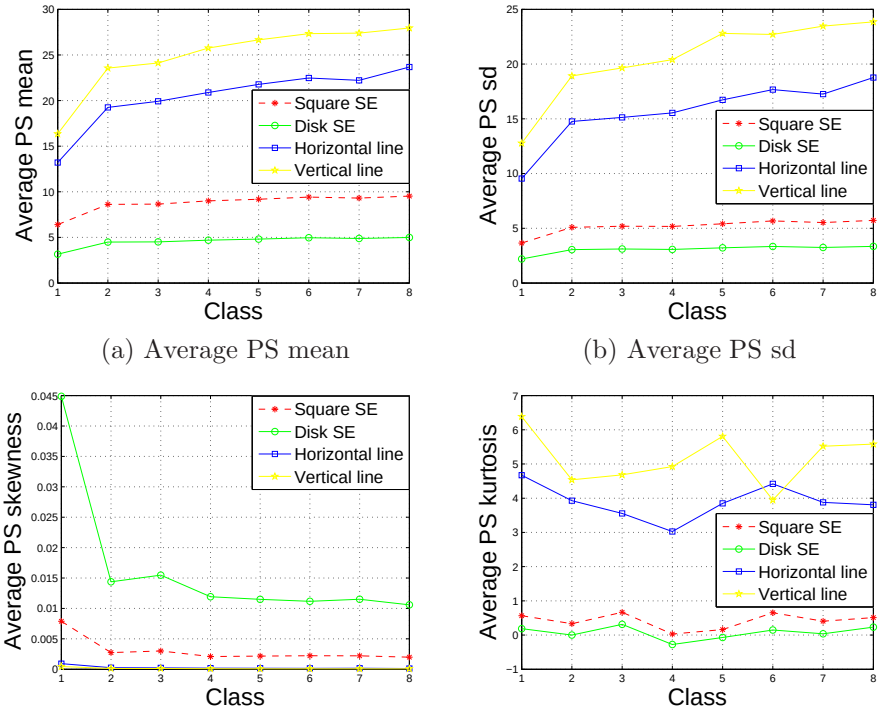
Figure 7.3: One binary tea image from each class, and their PS using a disk SE.

line and vertical line, and PS moments were computed as before. One binary image from each class, along with its PS, using disk SE is shown in Figure 7.3.

In the granulometric opening the image volume drops successively at each opening, so any details smaller than the applied SE are completely removed from the image and create a spike in the PS. A significant drop in volume between two consecutive openings suggests that the image contains many objects/details smaller than the applied SE at that stage and the highest spike on the PS corresponds to that size of SE, but the PSs here do not show that trend. This may be because there are some black holes within the tea granules. Closing the images with a smaller SE, or applying median filtering, would lessen the problem but would cause another problem, i.e. of joining the adjacent granules, which would affect the original shape and size of the granules.

Foreground PS moments from binary images

As four different SEs are used for all 50 sub-images from each class and the first four PS moments are considered, the moments data form a 400×16 matrix. Average moments were computed over 50 sample images. So the average moment data consists of 8 rows, one for each class, and 16 columns, four average moments for each of the four SEs, 12 of which are used for prediction. The first four average foreground PS moments using four different SEs are plotted against tea class in Figure 7.4. It can be seen that the average PS mean and sd using all SEs increase with class, although the rate of increase is different for different SEs and moments, but PS skewness is very low in magnitude for all SEs and a decreasing trend is observed. Kurtosis does not show any trend with class, therefore we used the first three PS moments in model building.



(a) Average PS mean

(b) Average PS sd

(c) Average PS skewness

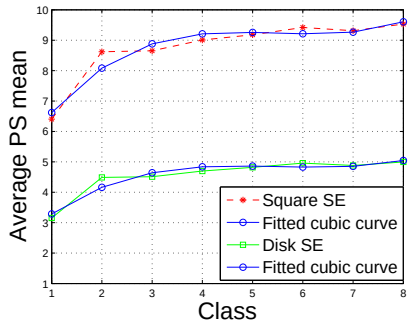
(d) Average PS kurtosis

Figure 7.4: Plot of the average foreground PS moments of the binary images using 4 SEs against class.

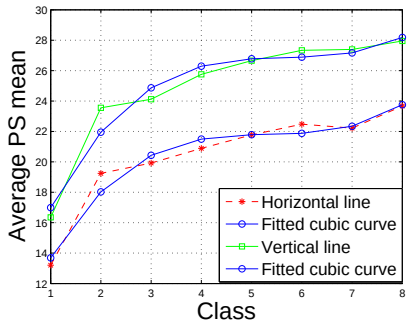
The regression approach was used to relate the foreground and background granulometric moments separately to tea class, and the fitted model was then used to predict the class for each image. We consider foreground features first.

In Figure 7.4 the first three foreground PS moments possess a strong relationship with class, but the relationships are curvilinear. The first three average

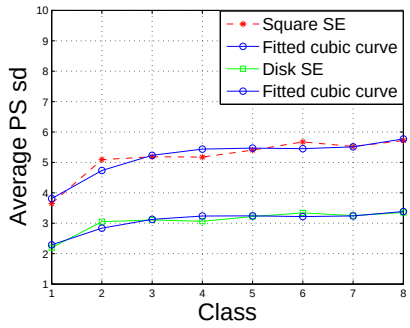
moments for each of the four SEs were modelled using a polynomial regression (equation (5.14)). Each of the 12 moments and corresponding fitted cubic is plotted in Figure 7.5. Although some of the figures suggest that a higher order polynomial than a cubic would be more appropriate (especially skewness using a disk) solving a 4^{th} or higher degree polynomial would be difficult and may be too complicated for good prediction.



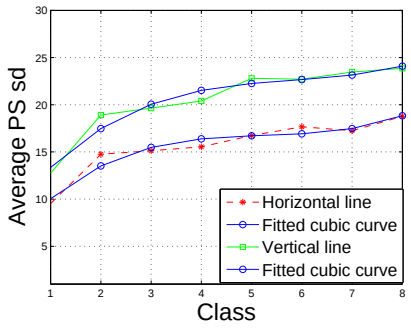
(a) PS mean using square and disk SE



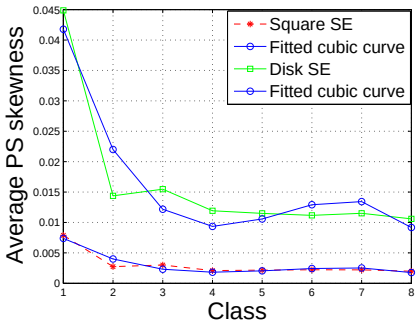
(b) PS mean using line SEs



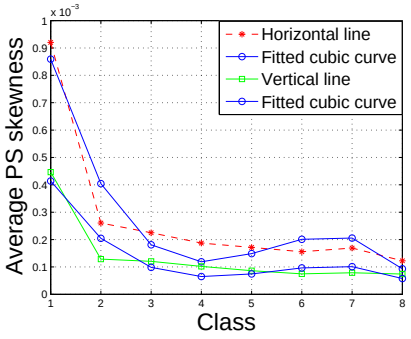
(c) PS sd using square and disk SE



(d) PS sd using line SEs



(e) PS skewness using square and disk SE

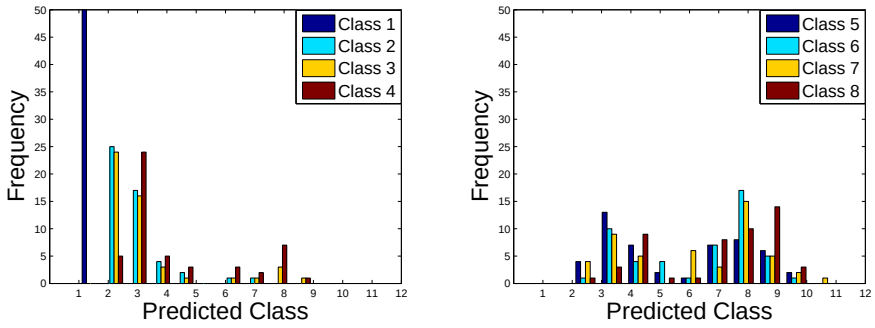


(f) PS skewness using line SEs

Figure 7.5: Plot of the average foreground PS moments for the binary images against class using different SEs, with fitted cubic curves (blue lines).

The fitted cubic model was used at this stage to predict class using the single set of moments for each of the 50 sub-images from each class. The histograms of predicted class are shown for all eight classes in Figure 7.6. The predicted classes

are quite spread out and are not concentrated on the correct class.



(a) Actual class = 1 to 4 (b) Actual class = 5 to 8

Figure 7.6: Frequency histograms of predicted class using the first three foreground PS moments of the binary tea images from square, disk, horizontal and vertical line SEs, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

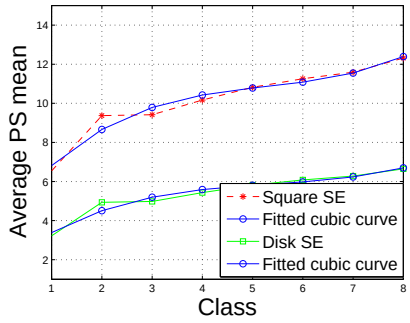
Background PS moments from binary images

Since granulometry on the foreground of an image only provides the size distribution of the objects within the image but not spatial arrangement, applying granulometry on the image background may also be useful. Again we used the four SEs above and computed the first four PS moments from each of the 50 images.

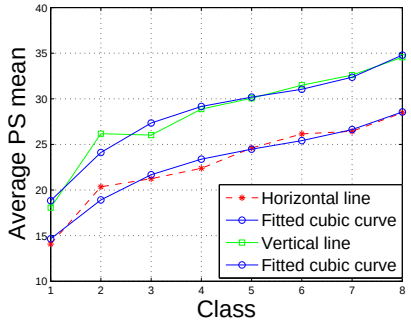
Figure 7.7 shows the first three background PS moments using the four different SEs, with fitted cubic regressions. Again the PS mean and sd using all SEs are increasing functions of class, although the rates of increase differ, and again the PS skewness is very low in magnitude for all SEs and a decreasing trend is observed. Again kurtosis (not shown) does not show much trend with class, therefore again we used only the first three PS moments in model building. The combined fitted model was used for prediction of class using the moments from each of the 50 sub-images from each class. The predicted classes are shown in Figure 7.8, but are still widely spread.

Principal component analysis

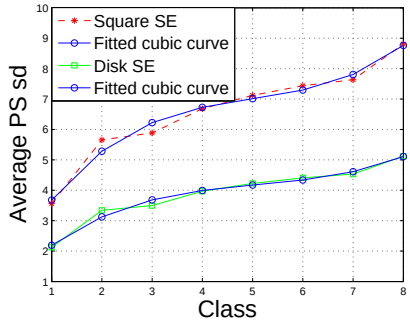
Both the first three foreground and first three background average PS moments using all four SEs are now considered, and PCA with and without normalisation applied to the full data matrix containing these 24 moments. In the un-normalised case, the first PC expresses 98.4% of the variation in the data and the first 2 PCs



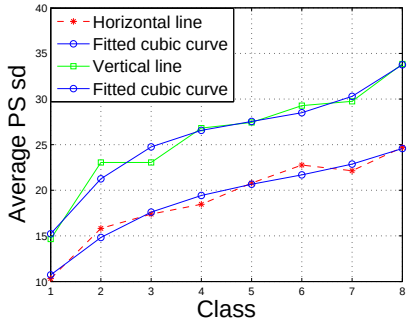
(a) PS mean using square and disk SE



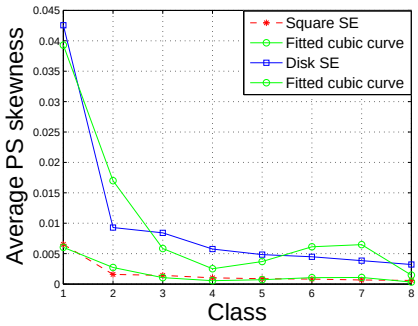
(b) PS mean using line SEs



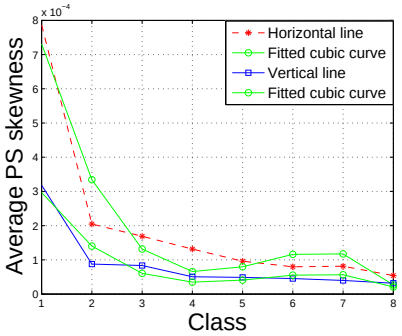
(c) PS sd using square and disk SE



(d) PS sd using line SEs



(e) PS skewness using square and disk SE

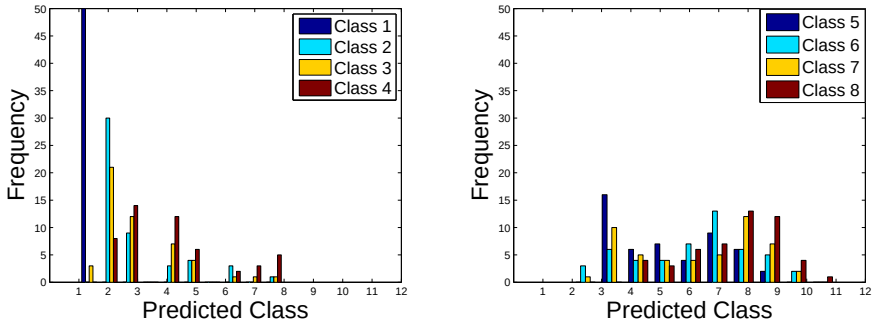


(f) PS skewness using line SEs

Figure 7.7: Plot of the average background PS moments for the binary images against class using different SEs, with fitted cubic curves (blue lines).

explain 99.3% of the variation, while without normalisation the first PC explains 95.5% of the variation and the first 2 PCs explain 99.5% of the variation in the data.

The PCs coefficients for the normalised case are shown in Table 7.1. In PC1, the strongest coefficients are for the line SEs. The background PS sd using a vertical line SE has the highest coefficient, and the second highest coefficient corresponds to the background PS mean using a vertical line. The coefficients are near zero using both foreground and background skewness for the line SEs. In general, both foreground and background PS skewness have very low coefficients

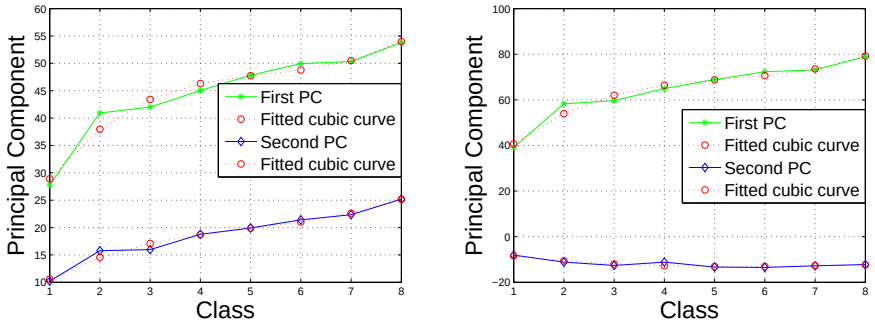


(a) Actual class = 1 to 4 (b) Actual class = 5 to 8

Figure 7.8: Frequency histograms of predicted class using the first three background PS moments of the binary tea images from square, disk, horizontal and vertical line SEs, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

and the line SEs have higher coefficients for the background PS mean and sd than the foreground ones. Therefore the PS means and sds using the line SEs appear to be the most informative.

Rather than using 12 foreground and 12 background PS moments, the first two PCs with and without normalisation were modelled as a function of class using cubic regression and a reasonably good fit was obtained. The PCs and fitted cubic curves are shown in Figure 7.9. With normalisation both PCs increase with class, but without normalisation only the first PC shows a relationship with class. Therefore the PCs obtained using normalisation were used in model building for prediction of class for each of the 50 sub-images in each class. The predicted classes for the 50 sample images in each class are plotted in Figure 7.10. Still the results are not ideal.

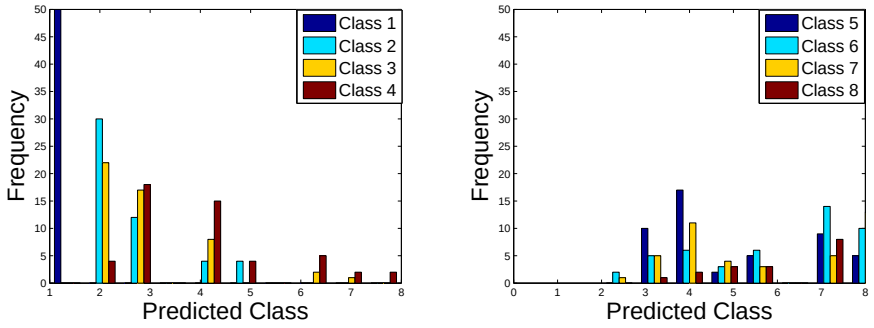


(a) PCs with normalisation (b) PCs without normalisation

Figure 7.9: Plots of the first two PCs of the 24 moments from the binary images against class, with fitted cubic curves.

Table 7.1: Principal component coefficients using 12 foreground and 12 background moments from the binary tea images. The bold figures refer to the strongest coefficients.

PS Moments	Principal component coefficients (with normalisation)					
	SE	PC1	PC2	PC3	PC4	PC5
Foreground mean	square	0.0781	-0.2071	0.0869	0.1148	-0.0250
	disk	0.0460	-0.1307	0.0469	0.0721	-0.0218
	line at 0°	0.2548	-0.3495	0.1212	0.4015	0.0209
	line at 90°	0.2985	**-0.5553**	0.1278	-0.0605	0.4948
Foreground sd	square	0.0513	-0.1319	0.0025	0.1521	-0.1729
	disk	0.0283	-0.0874	0.0021	0.0906	-0.1146
	line at 0°	0.2230	-0.1936	-0.0125	0.4547	**-0.5448**
	line at 90°	0.2889	-0.3465	-0.3215	**-0.4919**	-0.0319
Foreground skewness	square	-0.0001	0.0006	-0.0005	-0.0002	0.0000
	disk	-0.0008	0.0039	-0.0023	-0.0017	0.0014
	line at 0°	-0.0000	0.0001	-0.0000	-0.0001	0.0000
	line at 90°	-0.0000	0.0000	-0.0000	-0.0000	0.0000
Background mean	square	0.1443	0.0265	0.0100	-0.1161	-0.1459
	disk	0.0863	-0.0014	0.0013	-0.0789	-0.0661
	line at 0°	0.3631	0.1793	-0.3642	-0.0463	-0.3321
	line at 90°	0.4147	0.0900	0.2620	-0.4629	-0.2606
Background sd	square	0.1261	0.1466	0.0494	0.0143	0.1141
	disk	0.0732	0.0681	0.0183	-0.0172	0.0971
	line at 0°	0.3650	0.2858	**-0.5991**	0.2882	0.3922
	line at 90°	**0.4633**	0.4221	0.5410	0.1305	0.1802
Background skewness	square	-0.0001	0.0006	-0.0003	-0.0002	0.0003
	disk	-0.0010	0.0043	-0.0022	-0.0017	0.0018
	line at 0°	-0.0000	0.0001	-0.0000	-0.0000	0.0000
	line at 90°	-0.0000	0.0000	-0.0000	-0.0000	0.0000

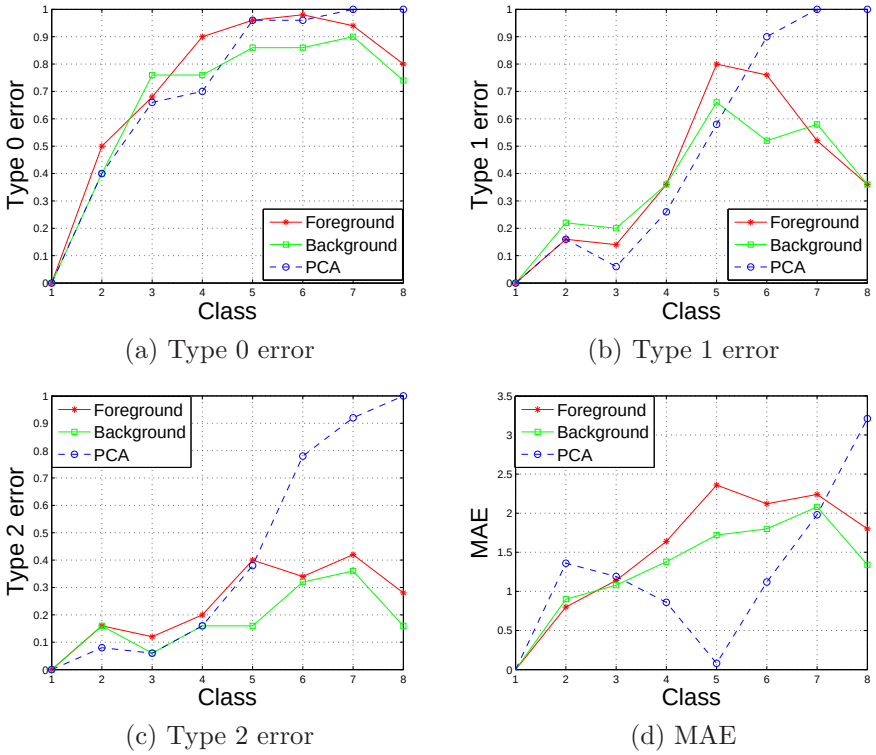


(a) Actual class = 1 to 4 (b) Actual class = 5 to 8

Figure 7.10: Frequency histograms of predicted class using the first two PCs derived from 24 moments (12 foreground and 12 background) of the binary tea images from square, disk, horizontal and vertical line SEs, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

Prediction errors

The accuracy of the predicted class was computed as in Section 5.3.4. Type 0, Type 1, Type 2 error rates and MAE were computed for each of the three regression models (foreground moments, background moments and PC-based) and shown in Figure 7.11 and Table 7.2. Overall type 0 error rate for the foreground model is 72%, for the background model it is 66% and for the PC-based model it is 71%. The background moments model produced better classification than the foreground moments or PC-based model, although the error rates are still very high. In terms of MAE (Figure 7.11(d)), the PC-based model produced slightly lower MAE than the background moments model. For all models, later classes are more difficult to classify.



(a) Type 0 error

(b) Type 1 error

(c) Type 2 error

(d) MAE

Figure 7.11: Overall prediction error measures of the three binary image models tested on each of the 50 sub-images from each class.

Since the background moments provide comparatively better classification results we used the background moments in the other classifiers, i.e. SVM, LDA and FF-NNET, and compared their performance with that of the regression approach (REG).

Table 7.2: Overall error rates for all 3 regression models tested on each of the 50 sub-images from each class, for the binary tea images.

Class	Foreground moments				Background moments			
	MAE	Type 0	Type 1	Type 2	MAE	Type 0	Type 1	Type 2
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.80	0.50	0.16	0.16	0.90	0.40	0.22	0.16
3	1.14	0.68	0.14	0.12	1.08	0.76	0.20	0.06
4	1.64	0.90	0.36	0.20	1.38	0.76	0.36	0.16
5	2.36	0.96	0.80	0.40	1.72	0.86	0.66	0.16
6	2.12	0.98	0.76	0.34	1.80	0.86	0.52	0.32
7	2.24	0.94	0.52	0.42	2.08	0.90	0.58	0.36
8	1.80	0.80	0.36	0.28	1.34	0.74	0.36	0.16
Overall	1.51	0.72	0.39	0.23	1.29	0.66	0.36	0.17
Class	Principal components (2 PCs)							
	MAE	Type 0	Type 1	Type 2				
1	0.00	0.00	0.00	0.00				
2	1.36	0.40	0.16	0.08				
3	1.19	0.66	0.06	0.06				
4	0.86	0.70	0.26	0.16				
5	0.08	0.96	0.58	0.38				
6	1.12	0.96	0.90	0.78				
7	1.98	1.00	1.00	0.92				
8	3.21	1.00	1.00	1.00				
Overall	1.23	0.71	0.50	0.42				

7.3.2 Other classifiers

We considered different kernels and a grid search approach to choose the optimum kernel, parameter values and cost for the SVM. Error rates were computed for a single training set, where the training set consists of 70% of the data. For SVM the error rates for different kernels are shown in Table 7.3. Any value of the cost (we tested costs between 1 and 100) with the linear kernel produces 100% correct classification. The radial basis kernel with $\gamma = 0.1$ and a cost of 10 or above produced only 1% error. A polynomial kernel produced 11% error rate for any cost of 10 or above and γ greater than or equal to 0.5. The default value of $\eta = 0$ produced higher error but any value of η between 1 to 5 produced the same accuracy, hence we considered $\eta = 1$ for the polynomial kernel. Therefore we used the simpler linear kernel with a cost of 100 for this dataset. Similarly, with normalised features, ten units in the hidden layer of a FF-NNET with decay as 10^{-4} and rang as $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, produced a lower

error rate for the training set (Table 7.4).

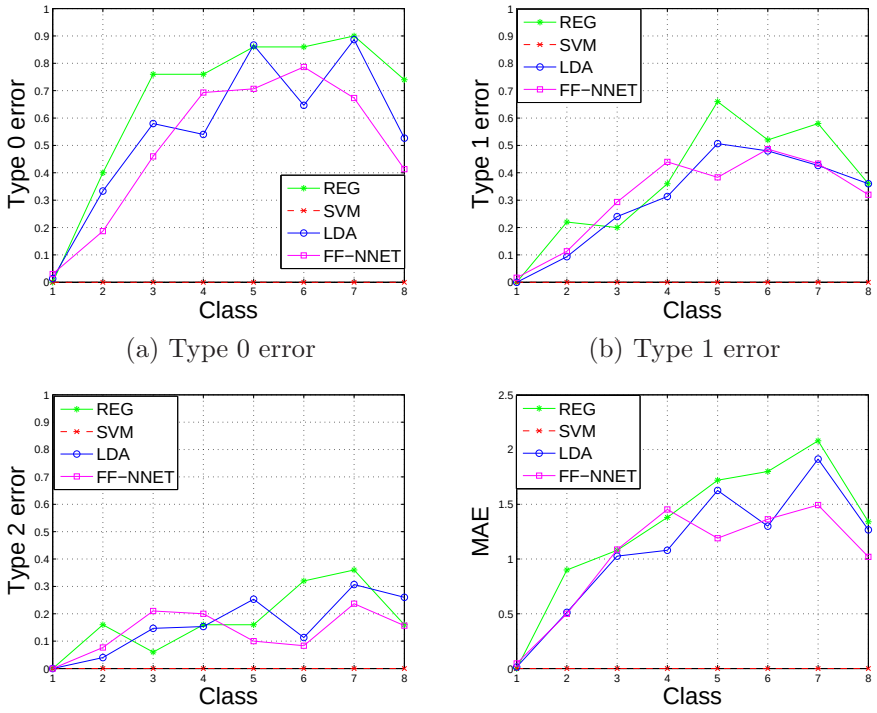
Table 7.3: Training set error rates for different combinations of cost and γ for the radial basis kernel and the polynomial kernel with $\eta = 1$, and a linear kernel using 12 background moments from the binary tea images. A 0 below means exactly 0.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.3	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.5	0.23	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12
0.9	0.24	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.59	0.38	0.33	0.26	0.25	0.23	0.22	0.20	0.19	0.18	0.17
0.5	0.17	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
0.9	0.12	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
	Linear kernel										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
	0	0	0	0	0	0	0	0	0	0	**0**

Table 7.4: Training set error rates from a grid search approach for finding the optimum number of hidden neurons for FF-NNET using 12 background moments from the binary tea images.

	Number of units									
	1	2	3	4	5	6	7	8	9	10
Error rate	0.52	0.36	0.34	0.25	0.24	0.26	0.21	0.13	0.11	**0**

The error rates for these classifiers are shown in Table 7.5 and Figure 7.12. SVM produced 100% correct classification for all classes. For REG, LDA, and FF-NNET the error rates are 66% (Table 7.2), 55% and 49% respectively. MAEs are also shown in the same table and again SVM is best.



(a) Type 0 error

(b) Type 1 error

(c) Type 2 error

(d) MAE

Figure 7.12: Error rates for all classifiers using 12 background PS moments from the binary tea images using all the 50 sub-images from each class.

7.4 Granulometry on Grey Scale Tea Images

Foreground grey scale images

Granulometry is now applied using the foreground of the grey scale image directly rather than thresholding first. We first subtracted the minimum intensity from the grey scale image so that successive openings using increasing size SEs should remove the whole image volume. This is true for a square or a disk SE but applying a line SE of increasing length does not remove the whole image volume. Once the line has brought the image grey scale down to the minimum grey level of the pixels lying along the line, opening again has no further effect. To overcome this problem, instead of subtracting the overall minimum image grey level from each pixel's intensity, we subtracted from each row of the original grey scale image its respective minimum intensity before applying the horizontal line SE. Similarly before applying a vertical line SE we subtracted from each column of the grey scale image its minimum intensity. This could be done for lines at 45° and 135° similarly.

We computed the first four PS moments from the square, disk, horizontal line and vertical line SEs. The average moments are plotted against class in

Table 7.5: Overall error rates for SVM, LDA and FF-NNET using 12 background PS moments of binary tea images for all 50 sub-images from each class.

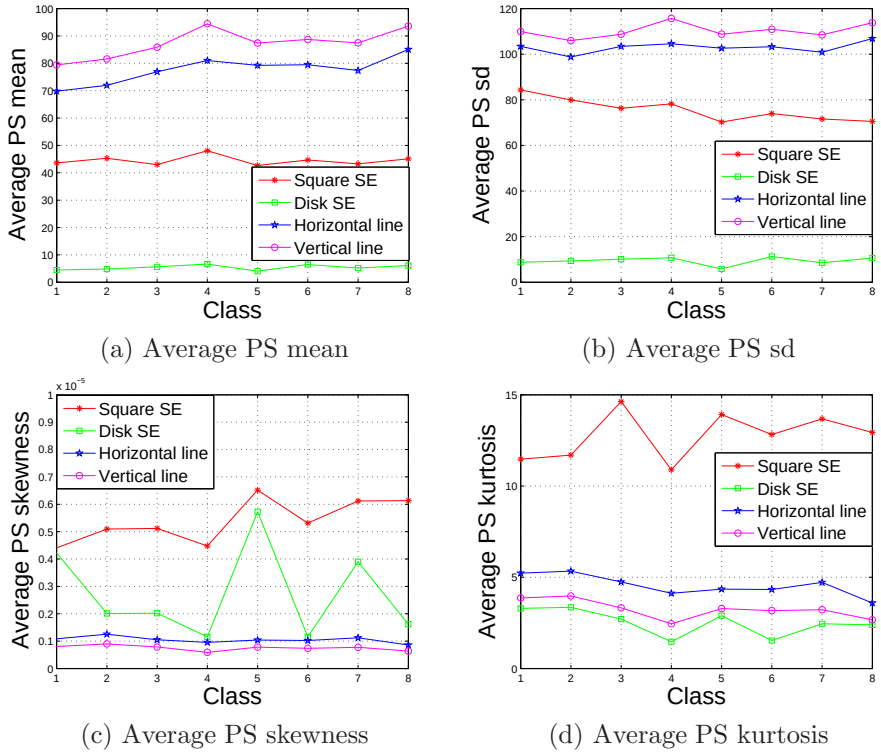
Class	SVM				LDA			
	MAE	Type 0	Type 1	Type 2	MAE	Type 0	Type 1	Type 2
1	0	0	0	0	0.01	0.01	0.00	0.00
2	0	0	0	0	0.51	0.33	0.09	0.04
3	0	0	0	0	1.03	0.58	0.24	0.15
4	0	0	0	0	1.08	0.54	0.31	0.15
5	0	0	0	0	1.63	0.87	0.51	0.25
6	0	0	0	0	1.30	0.65	0.48	0.11
7	0	0	0	0	1.91	0.89	0.43	0.31
8	0	0	0	0	1.27	0.53	0.36	0.26
Overall	0	0	0	0	1.09	0.55	0.30	0.16
Class	FF-NNET							
	MAE	Type 0	Type 1	Type 2				
1	0.05	0.03	0.02	0.00				
2	0.50	0.19	0.11	0.08				
3	1.09	0.46	0.29	0.21				
4	1.45	0.69	0.44	0.20				
5	1.19	0.71	0.38	0.10				
6	1.36	0.79	0.49	0.08				
7	1.49	0.67	0.43	0.24				
8	1.02	0.41	0.32	0.16				
Overall	1.36	0.49	0.31	0.13				

Figure 7.13. None of the moments show a strong relationship with class, especially skewness from a disk SE and kurtosis from a square SE. So we no longer consider these two moments and use the remaining 14 moments for prediction.

The predicted classes from cubic polynomial regression are shown in Figure 7.14. Prediction is now even worse than using the binarised tea images (Figure 7.6). The predicted classes are more widely spread and are not even centred at the actual class. A confusion matrix of the predicted classes is shown in Table 7.6.

Background grey scale images

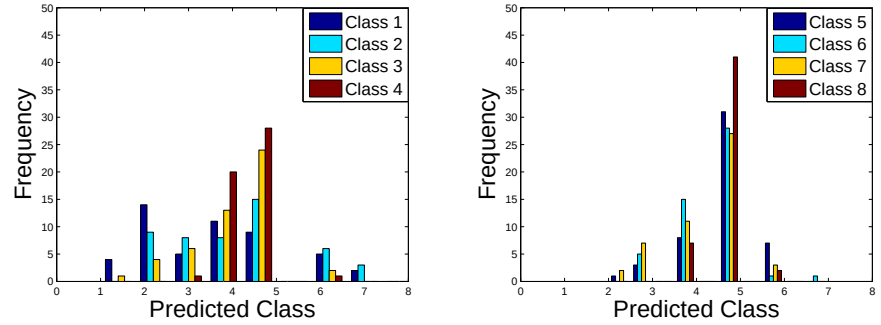
Example background tea images, one from each class, are shown in Figure 7.15. We applied granulometry using only a square and a disk SE (in this case the granules are more likely to be disk-shaped and closer to square-shaped then a line SE) on the background of the grey scale tea images. Average PS moments



(a) Average PS mean (b) Average PS sd

(c) Average PS skewness (d) Average PS kurtosis

Figure 7.13: Plots of the grey scale tea image average PS foreground moments against class using different SEs.



(a) Actual class = 1 to 4 (b) Actual class = 5 to 8

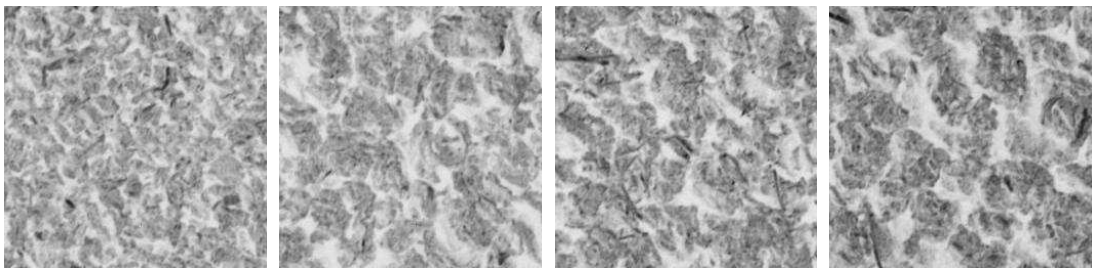
Figure 7.14: Frequency histograms of predicted class using 14 foreground PS moments from all 4 SEs (excluding skewness from the disk and kurtosis from the square SE) for the grey scale tea images, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

over all 50 sub-images from each class are shown in Figure 7.16. The PS mean and sd from both SEs in general decrease with class, but a disk SE shows a much smoother relationship with class than a square SE. Skewness from a square SE is near zero, while it decreases with class for a disk SE. Kurtosis from a square SE is almost constant and from a disk SE shows an irregular trend. Therefore we

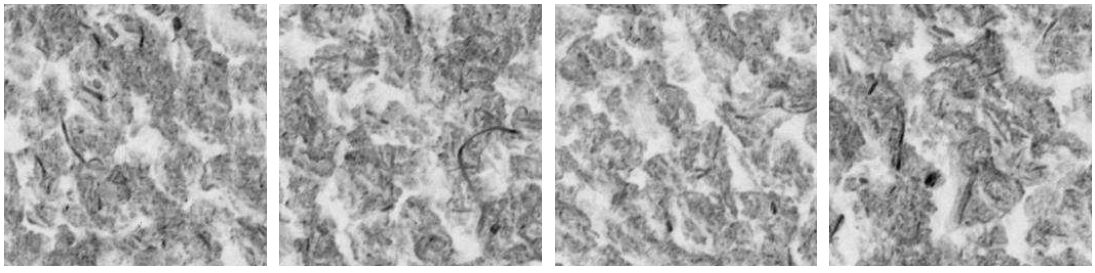
Table 7.6: Confusion matrix of the predicted class using the regression approach for the grey scale tea images, for all 50 sub-images in each class. The bold figures show the total number of correctly classified images.

	Predicted Class																
	Foreground PS moments									Background PS moments							
Class	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	7	8
1	**4**	14	5	11	9	5	2	0	0	**25**	13	4	3	4	1	0	0
2	0	**9**	8	8	15	6	3	0	1	0	**18**	13	8	6	5	0	0
3	1	4	**6**	13	24	2	0	0	0	0	5	**19**	6	18	2	0	0
4	0	0	1	**20**	28	1	0	0	0	0	2	7	**11**	26	4	0	0
5	0	1	3	8	**31**	7	0	0	0	0	0	8	6	**31**	5	0	0
6	0	0	5	15	28	**1**	1	0	0	0	2	4	4	31	**9**	0	0
7	0	2	7	11	27	3	**0**	0	0	0	0	3	5	40	2	**0**	0
8	0	0	0	7	41	2	0	**0**	0	0	0	2	1	41	6	0	**0**

used only the first two PS moments from both SEs and skewness from the disk SE in prediction. The predicted classes are shown in Figure 7.17 and Table 7.6. The predicted class ranges between 1 to 6 whereas the actual class labels are 1 to 8. Overall error rate is very high for all the classes. Therefore we did not use other classifiers with this set of moments.

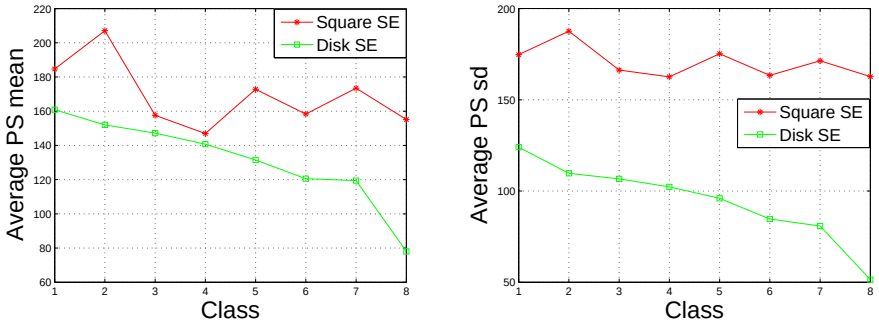


(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4



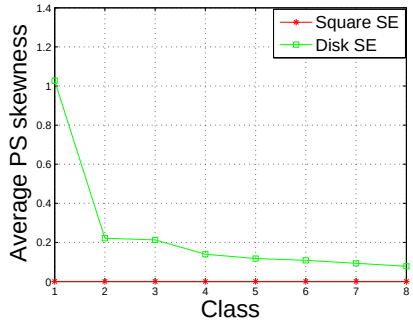
(e) Class 5 (f) Class 6 (g) Class 7 (h) Class 8

Figure 7.15: Background of grey scale tea images, one from each class 1 to 8.

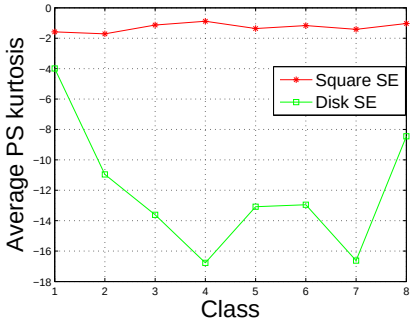


(a) Average PS mean

(b) Average PS sd

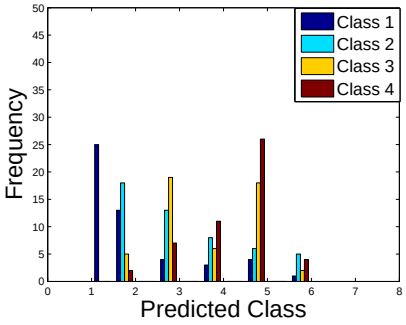


(c) Average PS skewness

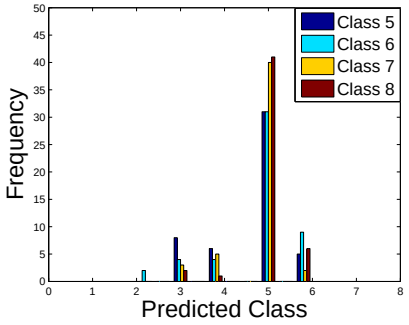


(d) Average PS kurtosis

Figure 7.16: Plots of the grey scale average PS background moments against class using a square and a disk SE.



(a) Actual class = 1 to 4



(b) Actual class = 5 to 8

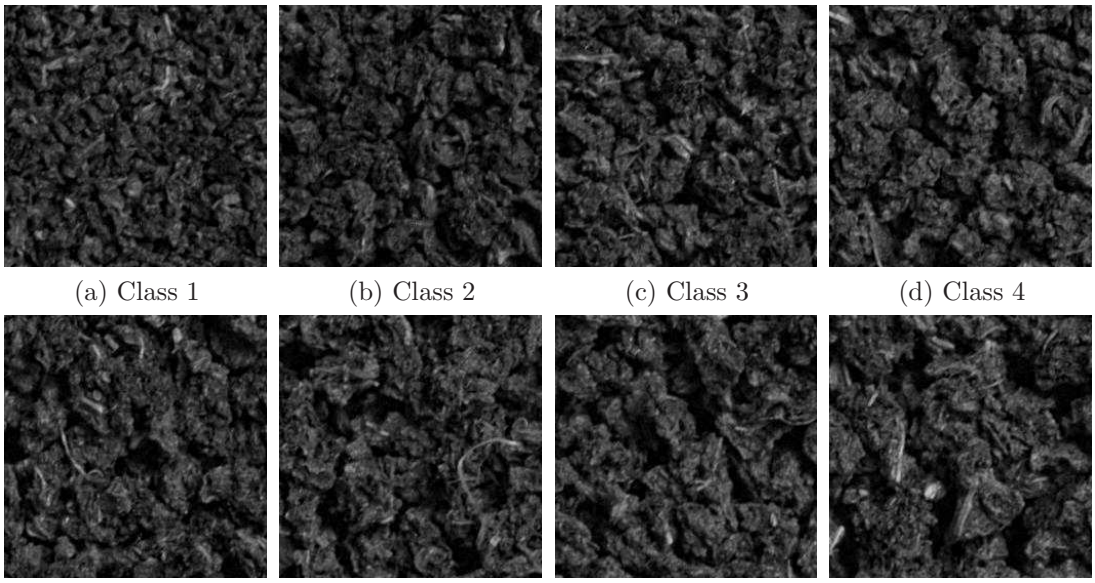
Figure 7.17: Frequency histograms of predicted class using 5 background PS moments from a disk and a square SE (first 2 moments from both and skewness from disk) for the grey scale tea images, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

7.5 Granulometry on Top-hat Transformed Images

We now investigate pre-processing of the images for better results. There is substantial intensity variation within the images (clearly visible in Figure 7.1),

as some parts of the images are brighter than others. Hence, the granulometric moments for the sub-images from each class vary widely. Using such a wide range of moments yields a wide range of the predicted class. So the uneven background grey level makes the prediction more challenging.

This uneven background variation can be reduced by using the top-hat transformation (Section 2.6) to highlight the tea granules (bright parts) and suppress the darker parts (background). As granule size increases with class, a disk SE of increasing size is used in the top-hat transformation. A wide range of radii were tested and it was found that disks of radius 13, 15, 17, 19, 21, 23 and 25 were the most appropriate ones for classes 1-8 respectively, to preserve best the tea granule sizes and shapes. A similar strategy as described in Section 6.4 would be needed in practice in the case of images with unknown class labels. One top-hat image from each class is shown in Figure 7.18.



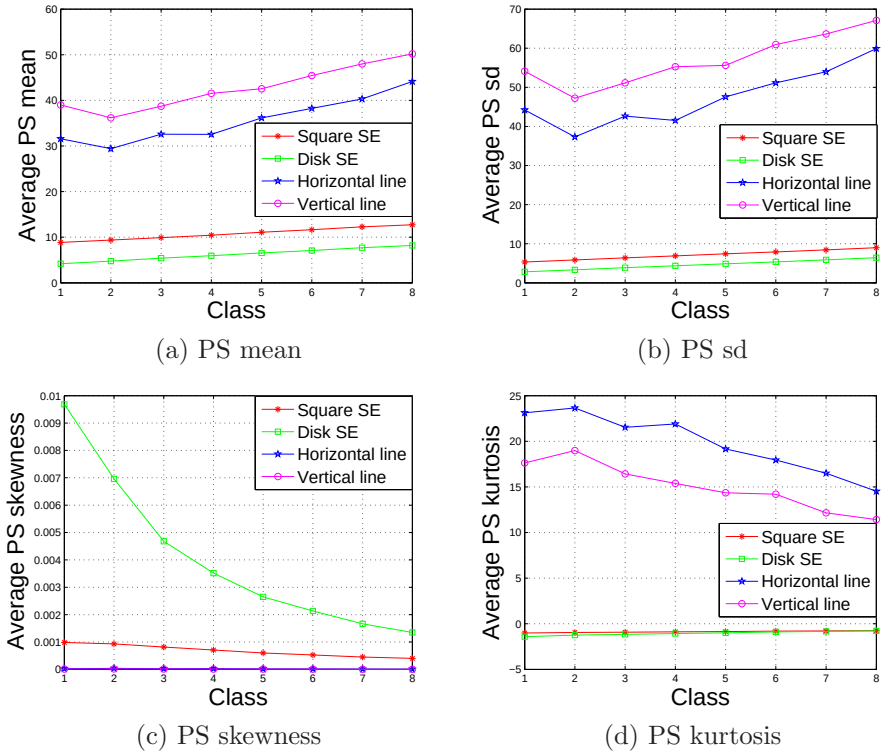
(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

(e) Class 5 (f) Class 6 (g) Class 7 (h) Class 8

Figure 7.18: Top-hat transformed grey scale images, one from each of classes 1 to 8.

Granulometries were applied using the square, disk, horizontal line and vertical line SEs and the first four PS moments from each SE were computed. Average PS moments were calculated using all sub-images from each class to give an 16×8 matrix and are plotted against class in Figure 7.19. The PS mean and sd using all SEs clearly increase with class, while a decreasing trend is observed for skewness using square and disk SEs, though it decreases faster for the disk SE than for a square SE and for a line SE skewness is near-zero. Kurtosis for the square and disk SE increase very slightly over the classes, whereas it decreases for any line

SE. As PS moments computed from a square and a disk SE provide a smoother relationship with class than the moments from the line SEs, they are more likely to provide better prediction. Therefore at first we used only the PS moments from the square and disk SE for prediction.



(a) PS mean (b) PS sd

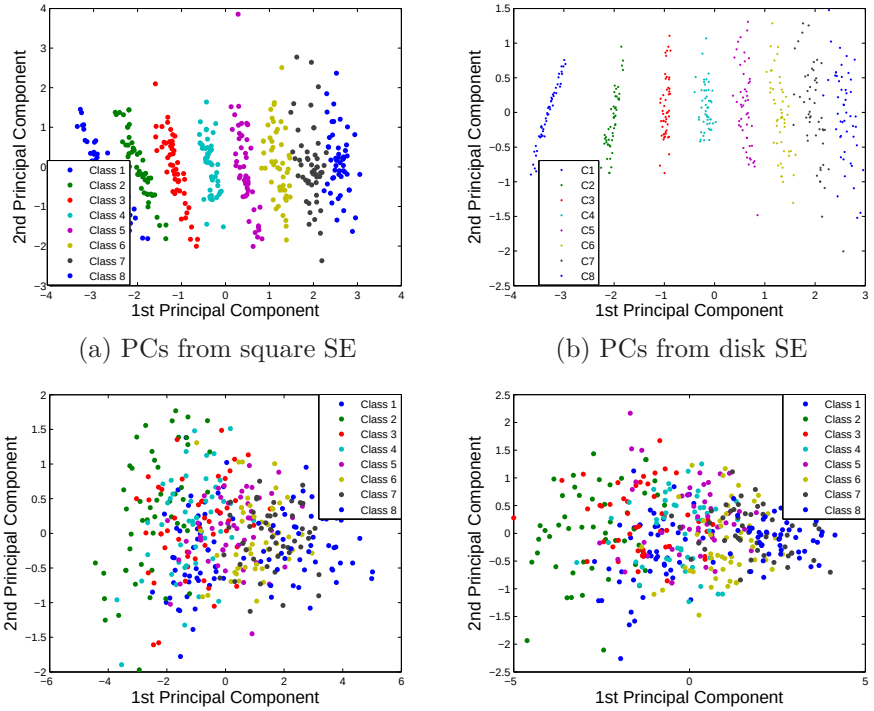
(c) PS skewness (d) PS kurtosis

Figure 7.19: Plots of the average top-hat image foreground PS moments against class, for square, disk, horizontal line and vertical line SEs, using all 50 sub-images from each class.

7.5.1 Exploratory analysis for top-hat images

The granulometric moments computed in Section 7.5 are used as texture features for predicting the class of the tea images. The moments data available consists of the first four PS moments using the square, disk, horizontal line and vertical line SEs for 50 non-overlapping sample images from each of the 8 classes, giving a 400×16 data matrix from all the images. The CVs of the PS moments were computed for this dataset (see Appendix IV). The CVs are very low at different class labels for all PS moments computed from both square and disk SE and would expect to give better classification results than the synthetic images used in Chapter 5. First we apply PCA for exploratory analysis and then compare results of the REG, SVM, LDA and FF-NNET as supervised classifiers.

Principal component analysis: Here we examine the discrimination ability of the PCs (with normalisation) of different sets of moments. Figure 7.20 shows scatter plots of the first two PCs from different sets of PS moments. The PCs from a square or a disk SE distinguished all 8 classes well. However the PCs from either of the line SEs moments cannot distinguish the different classes at all. Therefore we can say that a square or a disk SE extracts more information regarding the shapes and size of the tea granules than the other SEs, confirming the impression from Figure 7.19.



(a) PCs from square SE

(b) PCs from disk SE

(c) PCs from horizontal line SE

(d) PCs from vertical line SE

Figure 7.20: Scatter plots of 1st two PCs of the top-hat image foreground PS moments, from square, disk, horizontal line and vertical line SEs, using all 50 sub-images from each class (C1-C8).

Class separability measures: PCA provides only pictorial information regarding the discrimination ability of the data. Hence we computed scatter matrix-based class separability measures to get a numerical measure of the discriminating ability of the different feature sets. Webb (2002) defines the within-class scatter matrix as

$$\boldsymbol{S_W} = \sum_{i=1}^{K} \left[\sum_{j=1}^{N_i} (\boldsymbol{x_{i,j}} - \boldsymbol{m_i})(\boldsymbol{x_{i,j}} - \boldsymbol{m_i})^T \right]$$

where K is the number of classes, $N_i, i = 1, 2, \ldots, K$, is the number of cases in the i^{th} class, $\boldsymbol{x_{i,j}}$ is the j^{th} feature vector in the i^{th} class, and $\boldsymbol{m_i}$ denotes the mean vector of the cases in class i.

The between-class scatter matrix is:

$$\boldsymbol{S_B} = \sum_{i=1}^{K} \left[p(\omega_i)(\boldsymbol{m_i} - \boldsymbol{m})(\boldsymbol{m_i} - \boldsymbol{m})^T \right],$$

where $p(\omega_i) = N_i/N$ is the proportion of cases in class i, N is the total number of images and $\boldsymbol{m}$ is the overall mean vector, i.e. $\boldsymbol{m} = \sum_i \frac{N_i}{N} \boldsymbol{m}$ ($\boldsymbol{m}$ is the sample mean vector of class i, i.e. $\boldsymbol{m_i} = \sum_j \boldsymbol{x_{i,j}}/N_i$). The total scatter matrix $\boldsymbol{S_T}$ is the sum of the within-class and between-class scatter matrices, $\boldsymbol{S_T} = \boldsymbol{S_W} + \boldsymbol{S_B}$.

We used three different separability measures (Webb (2002)), namely, $J_1 = Tr\{\boldsymbol{S_W}^{-1}\boldsymbol{S_B}\}$, the ratio of the determinants $J_2 = \frac{|\boldsymbol{S_T}|}{|\boldsymbol{S_W}|}$ and the ratio of the traces $J_3 = \frac{Tr\{\boldsymbol{S_B}\}}{Tr\{\boldsymbol{S_W}\}}$, where $Tr\{\boldsymbol{A}\}$ is the sum of the diagonal elements of matrix A, $\boldsymbol{S_W}$ is a measure of the average variance of the features over all classes, and $Tr\{\boldsymbol{S_B}\}$ measures average distance of the mean of each class from the overall mean over all classes. A large $\boldsymbol{S_B}$ indicates good separation of the classes and a small $Tr\{\boldsymbol{S_W}\}$ indicates that the data points are well clustered around their mean within each class. A smaller value of $Tr\{\boldsymbol{S_W}\}$ or $|\boldsymbol{S_W}|$ compared to $Tr\{\boldsymbol{S_B}\}$ or $|\boldsymbol{S_B}|$ generates larger value of the measures J_1, J_2 and J_3. Larger values of these measures indicate stronger clustering of the feature set vectors for each case around the mean vector in that class, hence greater class separability.

Table 7.7: Scatter matrix-based separability measures for different PS moments sets $J_1 = Tr\{\boldsymbol{S_W}^{-1}\boldsymbol{S_B}\}$, $J_2 = \frac{|\boldsymbol{S_T}|}{|\boldsymbol{S_W}|}$ and $J_3 = \frac{Tr\{\boldsymbol{S_B}\}}{Tr\{\boldsymbol{S_W}\}}$. The bold figures represent the best values of the statistics.

SE	Tr(S_W)	Tr(S_B)	J_1	J_2	J_3
Square	24.278	0.411	18.515	19.515	0.017
Disk	**9.659**	0.451	**19.044**	**20.044**	**0.047**
Horizontal line (HL)	339.845	4.570	0.737	1.737	0.013
Vertical line (VL)	519.153	4.398	1.393	2.393	0.009
Square+Disk	17.789	0.431	7.936	8.936	0.024
Square+Disk+ HL	236.357	1.251	0.436	1.436	0.005
Square+Disk+ VL	14.46	0.191	4.336	5.336	0.013
Square+Disk+ HL+VL	367.045	1.829	0.586	1.586	0.005

Table 7.7 shows the values of these separability measures for different sets of features. The smallest value of $Tr\{\boldsymbol{S_W}\}$ corresponds to the disk SE and the

second smallest corresponds to the combination of the square, disk and vertical line PS moments. Therefore the PS moments from the disk SE are expected to be the best feature set for separation of the classes, and the moments from the square, disk and vertical line SEs the second best set. Although the best (largest) values of $Tr\{\boldsymbol{S_B}\}$ are for the horizontal line and vertical line SEs, these also have very high $Tr\{\boldsymbol{S_W}\}$. The largest values of J_1, J_2 and J_3 correspond to the disk SE, and the second largest values for J_1 and J_2 are from the square SE (for J_3 the second best is for square and disk combined), which implies that the PS moments from a disk SE have slightly greater separation ability than those from a square SE. The line SE moments have much less class separation ability, even when combined with those of a square and disk SE. Therefore the separability measures and Figure 7.20 provide the same kind of information regarding the discrimination ability of different sets of moments.

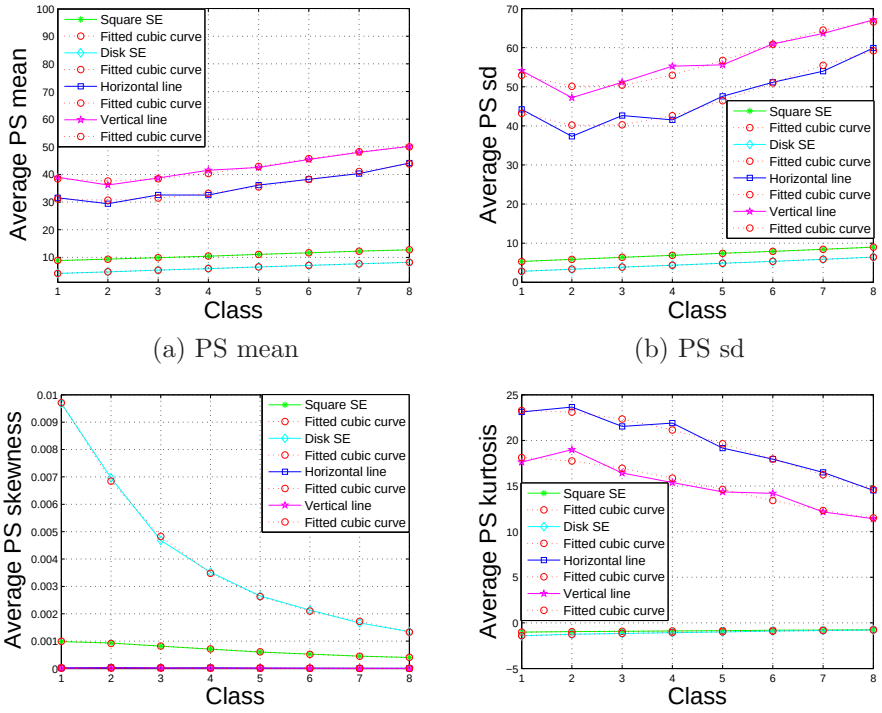
Since the PCA and the different separability measures reveal better separation ability of the disk and square PS moments, we now use the different classifiers with these moments sets.

7.5.2 Regression approach

Figure 7.21 shows the first four PS average moments using a square, disk, horizontal line and vertical line SEs, and the fitted cubic curves. All moments increase or decrease with class and better fit is obtained for the moments corresponding to the square and disk SE than for the line SEs. Since the first four PS moments corresponding to the square and disk SEs look more promising, only these were used to predict class for each individual sub-image.

Cubic polynomial regression models were fitted for all moments using all 50 sub-images from each class. In the root finding algorithm, in some cases none of the roots were real and the predicted class was taken as the first class. In the case of multiple real roots the smallest one was used, as previously. In the case of predictions outside of the range 1 to 8, the predictions were clipped to 1 to 8 as appropriate. This made only a small difference to the results.

The predictions are shown in Figure 7.22. The technique correctly predicted class for all 50 samples in the first 2 classes. For class 3, 46 images (92%) are predicted correctly, and 4 are predicted as class 4, and for class 4 49 images (98%) are predicted correctly and only one is predicted as class 5. For class 5, 41 images (82%) are predicted correctly, 7 of them are predicted as class 6 and 2 as 4. For class 6, 43 images (86%) are predicted correctly, 4 of them are predicted as class 5 and 3 of them are predicted as class 7. For class 7, 35 images (70%) are predicted

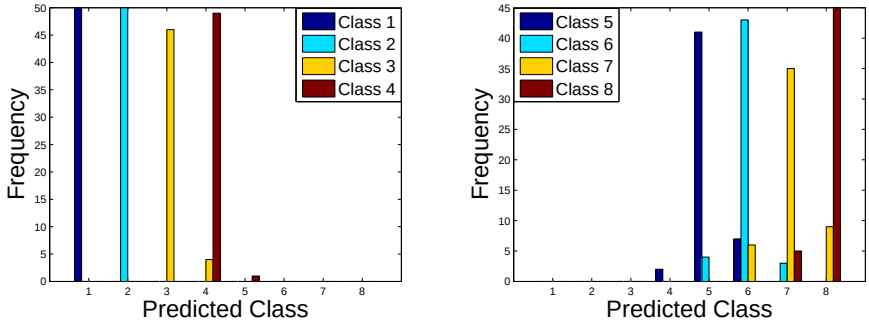


(a) PS mean (b) PS sd

(c) PS skewness (d) PS kurtosis

Figure 7.21: Plots of the first four average PS moments against class, using square, disk, horizontal line and vertical line SEs, along with the fitted cubic curves.

correctly, 6 of them are predicted as class 6 and 9 of them are predicted as class 8. For the last class, only 5 images are predicted as class 7, to give an overall 90% correct classification rate for the regression approach. The misclassifications are all only one class away from the correct class.



(a) Actual class = 1 to 4 (b) Actual class = 5 to 8

Figure 7.22: Frequency histograms of predicted class using 4 foreground PS moments from each of a square and disk for the top-hat transformed images, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

The above results were obtained to compare to results in the previous sections, but a more systematic approach is now followed to evaluate the accuracy of the

new regression classifier. Rather than using the full dataset to produce average moments to build the model and predicting the class separately for all of the sub-images, 70% of the data, i.e. 35 randomly sampled sub-images from each class, were used to build the cubic regression model and then we tested it on the remaining sub-images from each class. The process was repeated 10 times and the average predicted times, average error rates and average MAE were computed. The confusion matrix of the average predicted times (averaged over 10 runs) is shown in Table 7.8. A total of 12 images out of 120 are misclassified giving an average overall error rate of 10% and they all predicted only one class away from the actual class, so type 0 error and MAE are the same. However computing the error rate at each run and averaging these error rates yielded an average 10.9% error rate (Table 7.11).

Table 7.8: Confusion matrix for the regression approach, from the 1st 4 foreground top-hat image PS moments from each of the square and disk SEs, training on 35 sub-images and testing on the rest from each class; results are averaged over 10 runs. The bold figures are the number of correctly classified test set images.

Actual class	Predicted class using regression approach							
	1	2	3	4	5	6	7	8
1	**15**	0	0	0	0	0	0	0
2	0	**15**	0	0	0	0	0	0
3	0	0	**15**	0	0	0	0	0
4	0	0	0	**15**	0	0	0	0
5	0	0	0	2	**13**	0	0	0
6	0	0	0	0	7	**8**	0	0
7	0	0	0	0	0	0	**15**	0
8	0	0	0	0	0	0	3	**12**

7.5.3 Other classifiers

We again applied SVM, LDA and FF-NNET using the four PS moments from a square and a disk SE for all 50 sub-images from each class. To be comparable with the regression approach, we randomly chose 70% of the moments data to train the classifiers and tested on the rest, repeated the process 10 times and averaged the results over 10 runs to give final results.

We used the grid search approach with a single training set, as before, to identify the best kernel for the SVM and its parameters, and the optimum number of hidden neurons for a single hidden layer FF-NNET. Training set error rates

for different kernels in SVM are shown in Table 7.9. A radial basis kernel with $\gamma = 0.1$ produced 1% error rate for any cost of 10 or above. A polynomial kernel with $\eta = 1$ produced the lowest error rate of 1% for any cost of 20 or above with $\gamma = 0.1$. The default value of $\eta = 0$ for a polynomial kernel produced a larger error rate whereas any value between 1 and 5 produced the same error, so we chose $\eta = 1$. A linear kernel with any cost between 1 and 100 produced 100% correct classification. Therefore we used the linear kernel for SVM with a cost of 100.

Table 7.9: Training set error rates for different combinations of cost and parameter γ for a radial basis kernel, a polynomial kernel with $\eta = 1$ and a linear kernel using 8 moments from the top-hat transformed tea images using square and disk SEs.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.03	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.5	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
0.9	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.07	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.5	0.01	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
0.9	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
	Linear kernel										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
	0	0	0	0	0	0	0	0	0	0	**0**

For FF-NNET the data were normalised first and a grid search approach was applied to find the best value of decay and rang with a fixed number of hidden units and it was found that decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$ produced the lowest training set error. Then the optimum number hidden neurons were sought for this feature set. Any number of units between 1 to 10 with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$ produced 100% accuracy for a single training set using the moments from the square and disk SEs (fifth row of Table 7.10). We used 10 units in the hidden layer of FF-NNET. The optimisation problem was allowed to iterate until it converged.

SVM produced 100% correct classification. The overall error rate was 10.9%

Table 7.10: Single training set error rates from a grid search approach for finding the optimum number of hidden neurons for FF-NNET with rang of $1/\max(|\boldsymbol{x}|)$ and decay of 10^{-4} for different sets of moments. A 0 means exactly zero. The bold figures represent the choice used in FF-NNET.

Error rate	Number of units									
SE	1	2	3	4	5	6	7	8	9	10
Square	0	0	0	0	0	0	0	0	0	**0**
Disk	0	0	0	0	0 2	0	0	0	0	**0**
Line at 0° (HL)	0.56	0.46	0.43	0.39	0.36	0.36	0.22	0.18	0.21	**0.16**
Line at 90° (VL)	0.53	0.43	0.38	0.32	0.29	0.30	0.23	**0.16**	0.18	0.18
Square+Disk	0	0	0	0	0	0	0	0	0	**0**
Square+Disk+ HL	0	0	0	0	0	0	0	0	0	**0**
Square+Disk+ VL	0	0	0	0	0	0	0	0	0	**0**
Square+Disk+ HL+VL	0	0	0	0	0	0	0	0	0	**0**

for REG and LDA and FF-NNET produced the same error rate of 0.3%, using the first four granulometric moments from the square and disk SEs as features (Table 7.11).

Table 7.11: Class-wise and overall test set classification error rates for different classifiers using the full feature space (8 moments, the first 4 PS moments from the square and the disk SEs); results are averaged over 10 runs.

Class	8 moments			
	REG	SVM	LDA	FF-NNET
1	0.000	0	0.000	0.000
2	0.000	0	0.000	0.000
3	0.087	0	0.000	0.000
4	0.020	0	0.000	0.000
5	0.187	0	0.000	0.000
6	0.173	0	0.000	0.000
7	0.287	0	0.027	0.007
8	0.120	0	0.000	0.017
Overall	0.109	0	0.003	0.003

7.5.4 Prediction using different sets of moments

Firstly, we used the first four PS moments from the four different SEs separately as features and used all the classifiers to classify the tea images according to

granule size. For all classifiers 70% randomly chosen sub-images from each class were used as the training set and the remaining 30% sub-images used for testing. The process was repeated 10 times and the final results were averaged over 10 runs, as above.

It was found that for all moment sets, using a single training set SVM with a linear kernel produced 100% correct classification using any cost between 1 and 100 (since the error rates are exactly 0, they are not shown here). Hence for SVM a linear kernel with a cost of 100 was used for all sets of moments. Table 7.10 shows the training error rates corresponding to the different number of hidden units in the FF-NNET for different moment sets, using decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$. Although in general the effect of the number of units is not large, we chose the number of units corresponding to the lowest error rate i.e. 10 units for all different sets of PS moments except for the vertical line SE, for which 8 units was appropriate (choosing the largest number of units that gives that error rate).

As seen from Table 7.12, SVM attained 100% correct classification for all sets of moments. For the other classifiers error rates are slightly higher for a square SE than a disk SE, but are much worse when the SE is either a horizontal or vertical line. We also investigated using the moments from a square and a disk, a square, a disk and a horizontal line and a square, a disk, a horizontal line and a vertical line. Using a square or a disk SE was much more successful than either of the line SEs, and most classifiers produced a lower error rate for a disk SE. Overall a disk SE, or a square and a disk SE do best.

It is clear that SE size and shape greatly affects the PS moments and hence the classification results. So, determining the best size and shape of a SE is of crucial importance. The overall selection of a SE depends upon the geometric shapes we attempt to extract from the image data. For example, tea granules are more likely to be a disk shape rather than square and are very distinct from lines at any direction.

The class-wise type 0 error rate or MAE for all classifiers using PS moments from the disk SE only is shown in Table 7.13. Type 0 error rates and MAEs are identical as the predicted classes were only one unit away from the actual class. Again SVM produced 100% correct classification. The overall error rates for REG, LDA and FF-NNET are 8.1%, 0.9% and 1.5% respectively, using the first four PS moments from the disk SE as features.

We also examined the effect of using only the PS mean and PS sd from both SEs, as removing redundant features can improve classification results. The

Table 7.12: Overall test set classification error rates for all classifiers using the first 4 PS moments from each different set of SEs; results are averaged over 10 runs.

SE	REG	SVM	LDA	FF-NNET
Square	0.188	0	0.018	0.018
Disk	0.081	0	0.009	0.015
Horizontal line (HL)	0.807	0	0.509	0.577
Vertical line (VL)	0.793	0	0.491	0.587
Square+Disk	0.109	0	0.003	0.003
Square+Disk+ HL	0.762	0	0.003	0.010
Square+Disk+ VL	0.718	0	0.003	0.009
Square+Disk+ HL+VL	0.793	0	0.008	0.013

same parameter settings were used as above. SVM was unaffected by using fewer features, but REG, LDA and FF-NNET do slightly worse. For all classifiers, classification of the earlier classes is easier than the later ones (Table 7.13).

Table 7.13: Class-wise and overall test set classification error for the classifiers using the full feature set (the first 4 PS moments from a disk SE) and a reduced feature set (the first 2 PS moments from a disk SE).

Class	4 moments				2 moments			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.000	0	0.000	0.000	0.000	0	0.000	0.000
2	0.000	0	0.000	0.003	0.000	0	0.000	0.000
3	0.020	0	0.000	0.000	0.020	0	0.000	0.007
4	0.000	0	0.000	0.000	0.020	0	0.013	0.023
5	0.100	0	0.000	0.000	0.120	0	0.100	0.117
6	0.160	0	0.000	0.020	0.193	0	0.173	0.157
7	0.220	0	0.030	0.057	0.300	0	0.133	0.187
8	0.147	0	0.040	0.040	0.167	0	0.080	0.077
Overall	0.081	0	0.009	0.015	0.103	0	0.063	0.071

The results in Table 7.13 were obtained by training the classifiers using 70% of the moments, randomly selected, and testing on the rest, repeating this 10 times and averaging the results. To consider the effect of size of the training set on each classifier we also used 40%, 50% and 60% of the available data to train each classifier and test on the rest, averaging the results over 10 runs. SVM is very robust to training set size (Table 7.14). REG and FF-NNET produced a slightly lower error rate for a larger training set, whereas LDA produced slightly lower

error rates for relatively smaller training sets. This will be due to randomness, as it would be expected that the error rate would be larger for smaller training sets, however the effect is very small.

Table 7.14: Average test set classification error rates for all classifiers using different training set sizes with the first 4 PS moments from a disk SE; results are averaged over 10 runs.

Training set size	4 moments			
	REG	SVM	LDA	FF-NNET
40%	0.093	0.000	0.001	0.020
50%	0.093	0.000	0.002	0.016
60%	0.091	0.000	0.003	0.015
70%	0.081	0.000	0.009	0.015

PCA on the transformed images: We computed the first four PS moments from the four SEs, and there are 50 sub-images in each of the 8 classes, so the moment data is of dimension 400×16. PCA with normalisation was applied to the whole dataset to identify the most informative moments. The cumulative proportions of the variation explained by the first 4 PCs are 72.93%, 80.86%, 87.73% and 92.61% respectively. The first 4 PCs are plotted against class in Figure 7.23. The first 3 PCs clearly increase with class, whereas the fourth PC decreases after class 4. The coefficients of the first 4 PCs are shown in Table 7.15. In PC1 the sd from a square SE has the highest coefficient, while the second highest corresponds to the sd from a disk SE. Again the mean from the line SE at 90° has the strongest coefficients both in PC2 and PC4 while kurtosis from line SE at 0° has the strongest coefficient in PC3. In general, square and disk SE moments have slightly higher absolute coefficients, so are more informative than the line SEs.

We used the first 4 PCs in the regression approach to predict class labels. The predicted classes are shown in Figure 7.24. The prediction is not better than prediction using the original moments. Type 0 error rate for the regression approach using 4 PCs is 83.3%, whereas using all 4 PS moments from the 4 SEs it is 79.3% (Table 7.12). Hence in this regard the use of PCs is not beneficial.

Table 7.15: Principal component coefficients using the 16 foreground moments from the top-hat transformed images.

PS Moments	Principal component coefficients (with normalisation)				
	SE	PC1	PC2	PC3	PC4
Mean	square	0.2825	0.0052	-0.1490	0.1908
	disk	0.2838	-0.0666	-0.1441	0.1377
	line at 0°	-0.2541	-0.1845	0.2220	-0.3805
	line at 90°	0.1754	**-0.5856**	-0.0418	**-0.4828**
Sd	square	**0.2841**	-0.0573	-0.1559	0.1331
	disk	0.2840	-0.1046	-0.1486	0.0843
	line at 0°	-0.2546	0.1067	0.3202	-0.2821
	line at 90°	0.2318	-0.4564	-0.1517	-0.1993
Skewness	square	0.2680	0.2612	-0.0675	-0.1905
	disk	0.2475	0.3006	0.0462	-0.3021
	line at 0°	-0.2417	-0.3449	-0.0277	0.3192
	line at 90°	-0.2009	-0.3125	0.1934	0.3939
Kurtosis	square	0.2685	-0.0163	0.3182	0.1354
	disk	0.2404	-0.0235	0.4654	0.0579
	line at 0°	-0.2346	0.0457	**-0.5196**	-0.0674
	line at 90°	-0.2182	0.0808	-0.3184	-0.1088

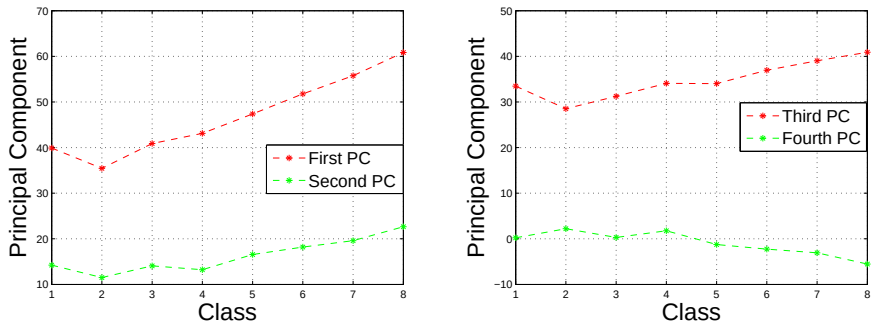
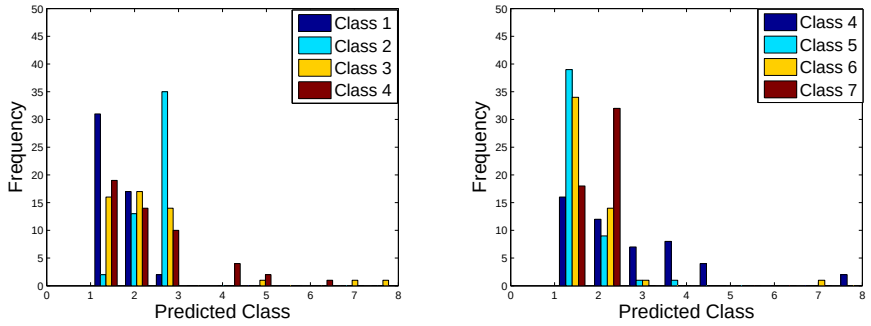


Figure 7.23: The first four PCs from the top-hat transformed images using all 4 moments from all 4 SEs, against class.

7.5.5 Top-hat transformation using the same disk SE

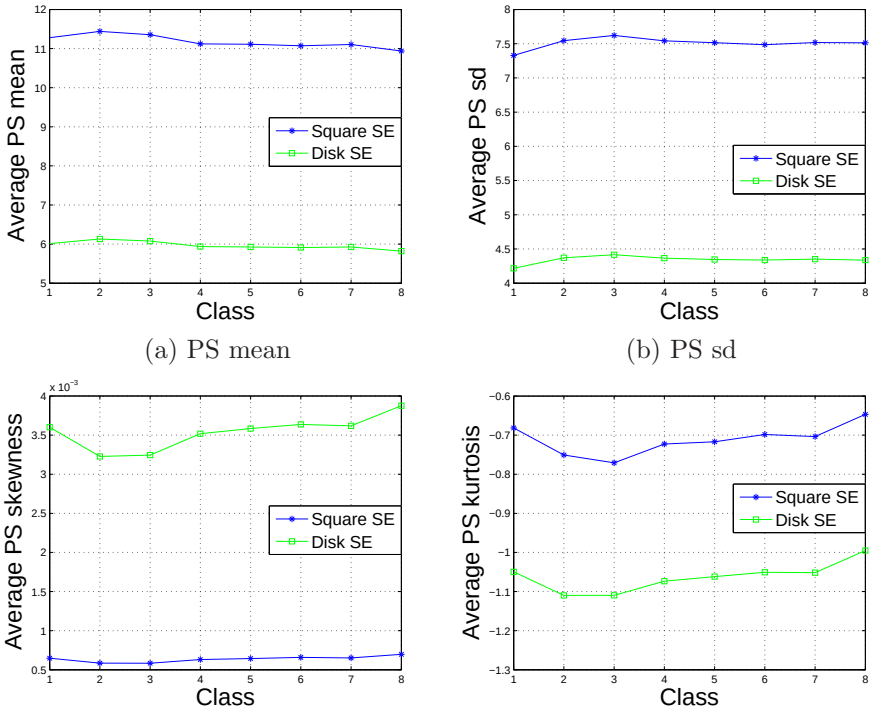
The key step in getting successful results from the granulometric moments for the tea images was to use the top-hat transform with a disk that increases in size for classes with larger tea granules. Here we show the results, for comparison, of applying the top-hat transformation using a disk of fixed radius over class 1 to 8, rather than using increasing radius. Different radii of $15, 17, 19, 21$ were experimented with and the results of radius 17 are shown here as an example.



(a) Actual class = 1 to 4 (b) Actual class = 5 to 8

Figure 7.24: Frequency histograms of predicted class using 4 PCs from 16 PS moments of the top-hat transformed images from 4 SEs, using cubic regression, for all 50 sub-images in each of the classes 1-4 (a) and 5-8 (b).

Granulometry was applied on those top-hat transformed images using the square and disk SE and the first four PS moments were computed as before. Average moments are shown in Figure 7.25. Average PS means and sds using both SEs are almost constant across class, whereas skewness and kurtosis increase slightly over time.



(a) PS mean (b) PS sd

(c) PS skewness (d) PS kurtosis

Figure 7.25: Plots of the PS moments against class, using square and disk SEs with a fixed size disk used in the top-hat transformation.

From this new moments dataset, again 70% of the moments were used for

training and the rest for testing, averaging the results for 10 runs. Type 0 error rates for all classifiers are shown in Table 7.16. The results are much worse than using a disk of increasing radius in the top-hat transform (Table 7.13).

The overall error rate for the regression approach is 83.9%. SVM using a linear kernel with any cost between 1 and 100 produced 100% correct classification. LDA produced a 63.4% error rate and FF-NNET using 10 units in the hidden layer and decay of 10^{-4} and rang of 0.7 (which produced the lowest training set error) yielded a 71% error rate, whereas using an increasing disk radius in the top-hat transformation these classifiers were able to achieve near 100% classification accuracy (Table 7.13).

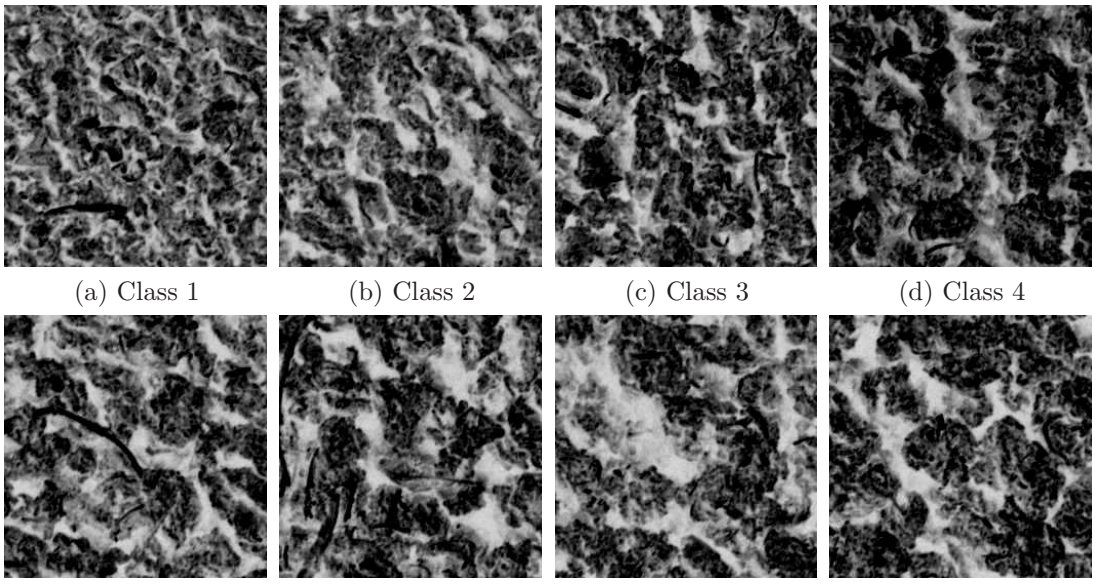
Table 7.16: Average test set Type 0 error rates for different classifiers using fixed sized disk in the top-hat transform, using four PS moments from a square and a disk SE; results are averaged over 10 runs.

Class	Type 0 error			
	REG	SVM	LDA	FF-NNET
1	0.600	0	0.140	0.293
2	0.780	0	0.420	0.563
3	0.900	0	0.573	0.700
4	1.000	0	0.813	0.860
5	1.000	0	0.907	0.853
6	0.987	0	0.853	0.887
7	0.980	0	0.933	0.833
8	0.467	0	0.433	0.690
Overall	0.839	0	0.634	0.710

7.5.6 Granulometry on the background images

For comparison, we applied bottom-hat transformation on the background of the tea images with the same SEs as were used in the top-hat transformation of the foreground of the tea images. For the corrosion images using the bottom-hat transform of the foreground and the top-hat transform of the background produced exactly the same images, hence the same PS moments (Section 6.4.3), but the situation is different for the tea images. The images in Figures 7.18 and 7.26 look different. So we applied granulometry on the bottom-hat transformed images using a square and a disk SE and calculated the first four PS moments. Average moments from both SEs are plotted against class, in Figure 7.27. None of the moments shows a clear trend with class, therefore it is unlikely to give

better classification results, especially with the regression approach. Hence we did not proceed with this set of moments.



(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

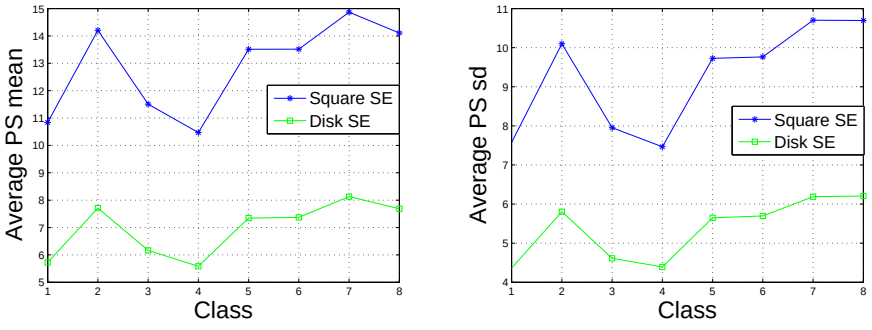
(e) Class 5 (f) Class 6 (g) Class 7 (h) Class 8

Figure 7.26: Results of bottom-hat transform on the background of the grey scale images using an increasing disk size, one image from each classes 1 to 8.

7.6 Use of Colour Images

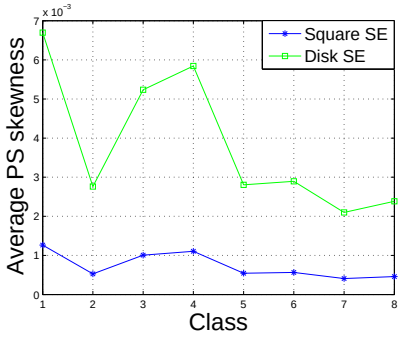
These tea images were supplied as colour images but we have converted them to grey scale versions. We did investigate whether much information was lost by not using the colour tea images. One colour sub-image of size 256^2 from each class is shown in Figure 7.28. Histograms of the red, green and blue planes of each of the eight classes are shown in Figure 7.29. In each colour plane the histogram of any one class looks different from those for the others classes but similar to the histogram for that class in the other colour planes. The grey scale image is an average of the intensities in the red, green and blue planes and, as the three colour planes are not very distinct from each other, the grey scale image should be as informative as any of the colour planes. Therefore we concluded that the grey scale images retain most of the information in the RGB images and use of the colour images would not provide substantially improved classification results.

We also considered the real HSV (hue, saturation and value/intensity) colour map of the images. Hue represents an angle from 0° to 360°, using a colour wheel starting from red at 0°, going through green at 120° and blue at 240° and then rolling back to red at 360°. The hue angles are given in radians. Saturation is a

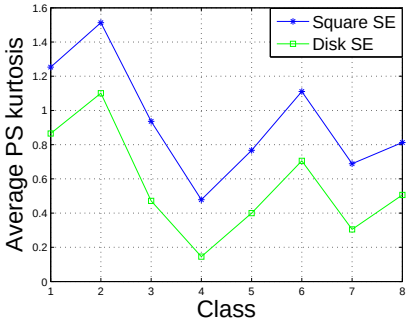


(a) PS mean

(b) PS sd

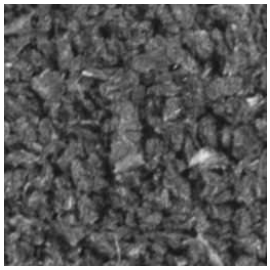


(c) PS skewness

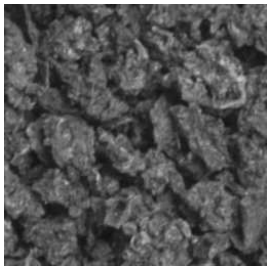


(d) PS kurtosis

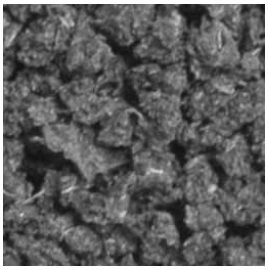
Figure 7.27: Plots of the PS moments against class, using square and disk SEs with increasing disk radius used in the bottom-hat transform of the images.



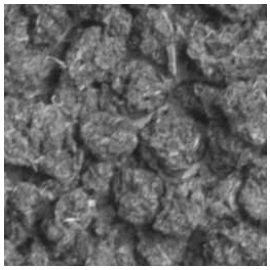
(a) Class 1



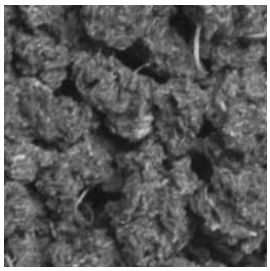
(b) Class 2



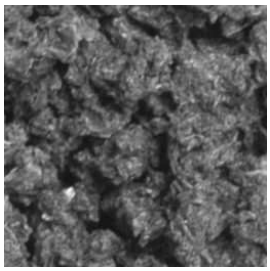
(c) Class 3



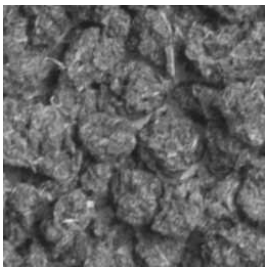
(d) Class 4



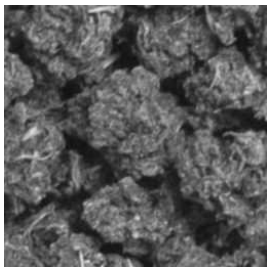
(e) Class 5



(f) Class 6



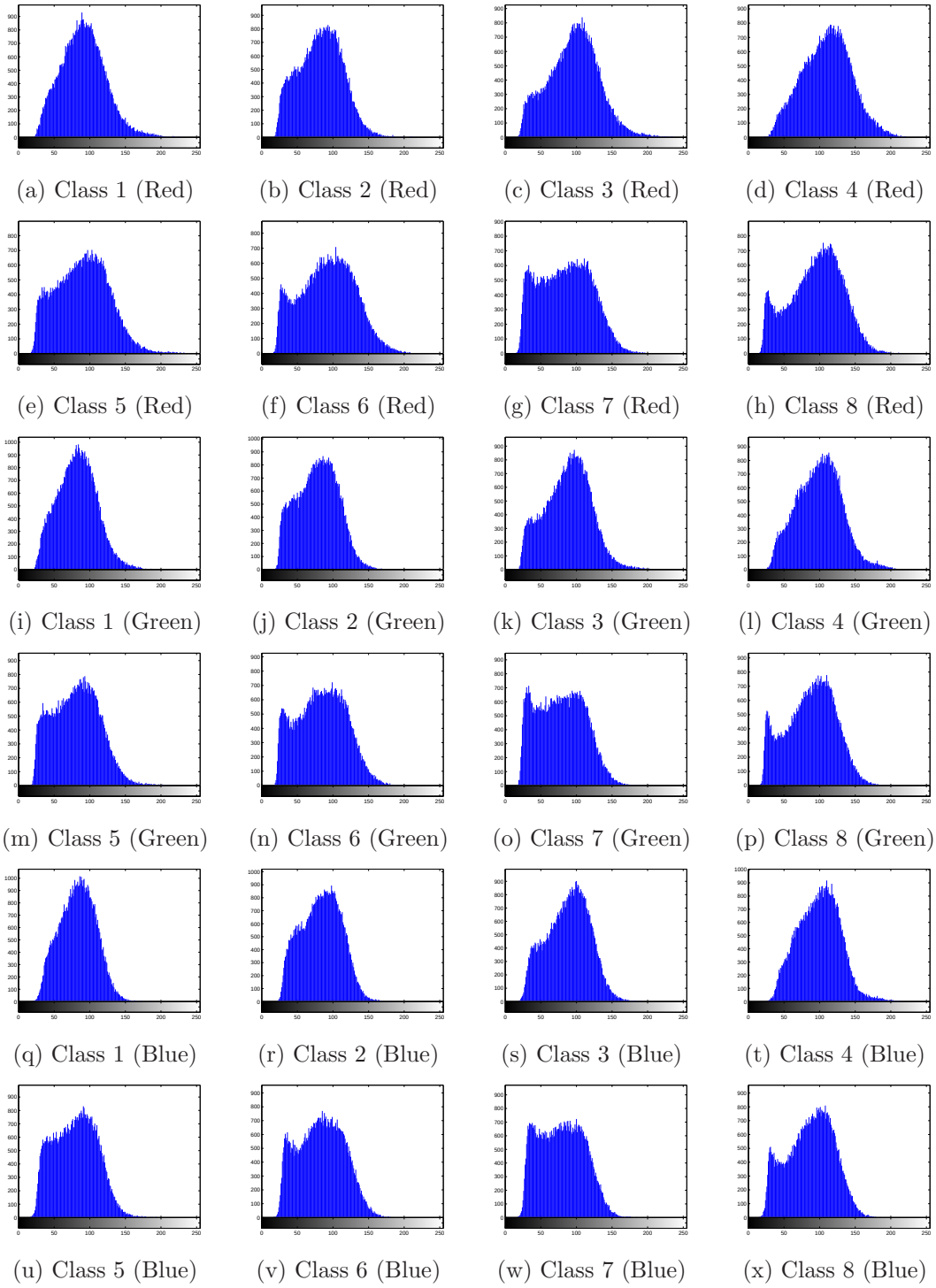
(g) Class 7



(h) Class 8

Figure 7.28: Sample 256^2 colour tea images, one from each of the eight classes.

measure of how pure the color is and it increases with the distance from the centre of the HSV colour space. For example, a pure red, dark blue and deep green are all highly saturated, while pink, light blue and light green are less saturated,



(a) Class 1 (Red) (b) Class 2 (Red) (c) Class 3 (Red) (d) Class 4 (Red)

(e) Class 5 (Red) (f) Class 6 (Red) (g) Class 7 (Red) (h) Class 8 (Red)

(i) Class 1 (Green) (j) Class 2 (Green) (k) Class 3 (Green) (l) Class 4 (Green)

(m) Class 5 (Green) (n) Class 6 (Green) (o) Class 7 (Green) (p) Class 8 (Green)

(q) Class 1 (Blue) (r) Class 2 (Blue) (s) Class 3 (Blue) (t) Class 4 (Blue)

(u) Class 5 (Blue) (v) Class 6 (Blue) (w) Class 7 (Blue) (x) Class 8 (Blue)

Figure 7.29: Histograms of the red, green and blue planes of the RGB colour tea images. One bar is used for each of intensities 0 to 255.

and grey has a saturation of zero. Intensity is the maximum of the normalised elements of the red, green and blue planes. If the images are of double precision the intensity of the hue, saturation, and intensity images ranges from 0 to 255,

but here the images are of 8-bits, so the intensities range from 0 to 1 (Levkowitz and Herman (1993)).

The colour images were converted to HSV images using MATLAB function 'rgb2hsv'. Although the pixel intensities in each of the red, green and blue planes range from 0 to 255, the intensities in the hue, saturation and intensity images range from 0 to 1. The histograms of the hue, saturation and intensity images were created using MATLAB function 'imhist', which creates a histogram using N specified bins of width $A/(N-1)$. For a double or single precision image A is 1 and for an 8-bit image A is 255 (Matlab help file). The hue, saturation and intensity images are shown in Figure 7.30. The classes are hard to distinguish from the hue images, apart from class 2. The saturation images are even harder to distinguish and the intensity images are similar to the grey scale images.

Figure 7.31 shows the histograms of the hue, saturation and intensity images, using $N = 256$ bins. Different numbers of bins, i.e. $N = 32$ and $N = 16$, were experimented with, and it was found that the bin size has no effect on the shape of the histogram. The shape of the histograms of the hue images only varies slightly over the classes (class 2 is a little different). For the saturation images the histograms are very similar to each other. As the intensity images are very similar to the grey scale images, the histograms vary over the classes and are similar to the corresponding histograms in the red, green and blue planes. Use of the hue and intensity (value) planes of HSV may bring some benefit over use of grey scale, but it was felt that this would be small and so these were not used here.

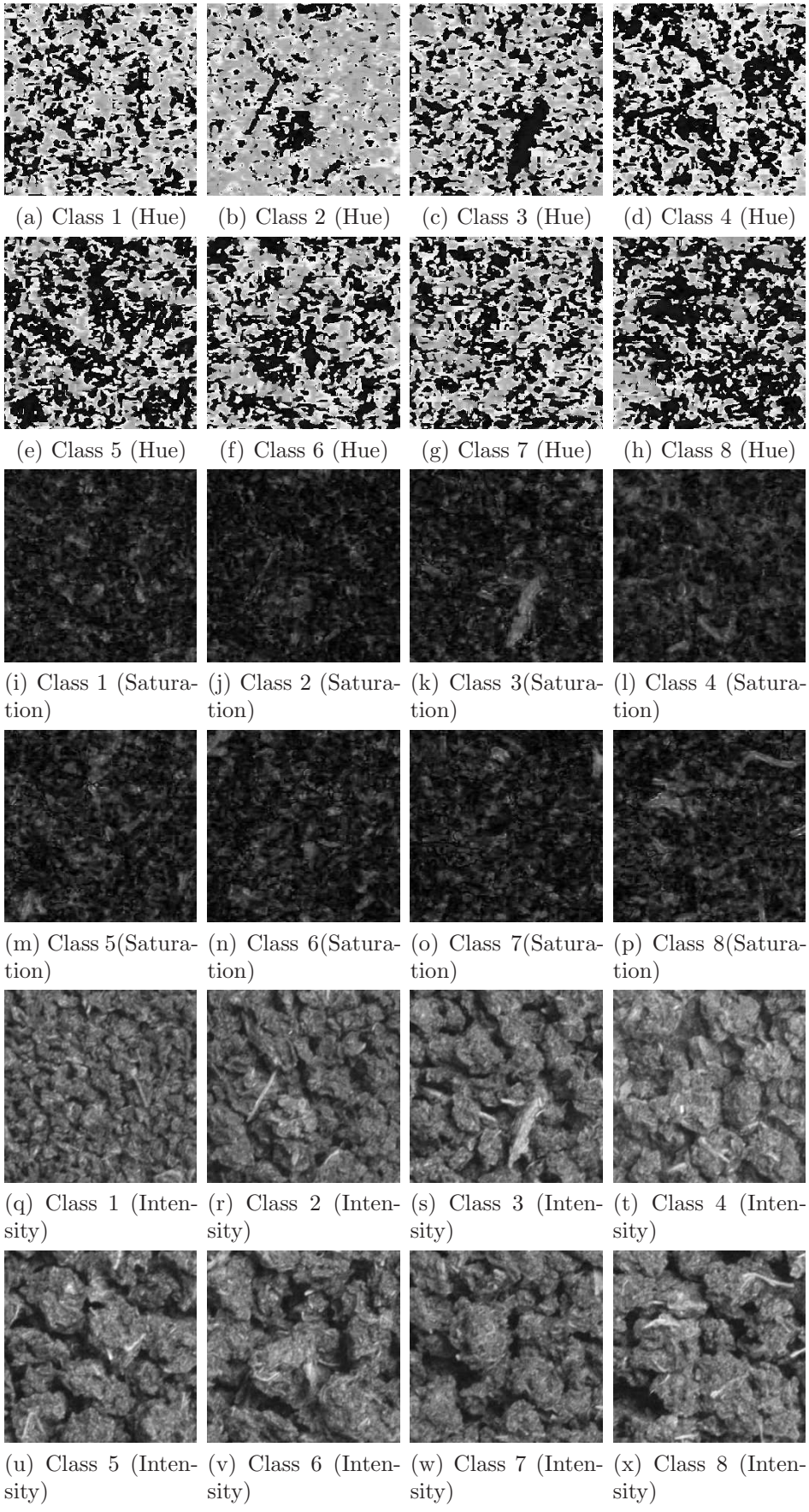
7.7 Conclusion

Borah et al. (2007) used image texture analysis to classify different grades of CTC tea according to their granule size. Wavelet-based features were used with two neural networks, i.e. the multilayer perceptron (MLP) and learning vector quantisation (LVQ) which gave 74.67% and 80% classification accuracy, i.e. the error rates of MLP and LVQ were 25.33% and 20% respectively.

This work is a substantial improvement on classifying these tea images over the results of Borah et al. (2007). Our highest classification error rate (8.1%) was obtained for the regression approach, 0% for SVM and 0.9% and 1.5% for LDA and FF-NNET, using the top-hat transformed images. Increasing the radius of the disk SE in the top-hat transform of the images from different classes is of crucial importance. Using the granulometric features from the top-hat im-

ages obtained using the same size disk SE over all 8 classes produced very high classification error for the regression approach especially.

The wavelet-based results of Borah et al. (2007) are not exactly comparable to ours, as we have extracted our own sub-images for algorithm development and testing. Nonetheless we conclude that extracting shape-based information from the tea granule images directly by use of morphological techniques provides very useful features for texture classification in any of a range of classifiers. We compare these directly with co-occurrence based features and wavelet-based features in Chapter 8.



(a) Class 1 (Hue) (b) Class 2 (Hue) (c) Class 3 (Hue) (d) Class 4 (Hue)

(e) Class 5 (Hue) (f) Class 6 (Hue) (g) Class 7 (Hue) (h) Class 8 (Hue)

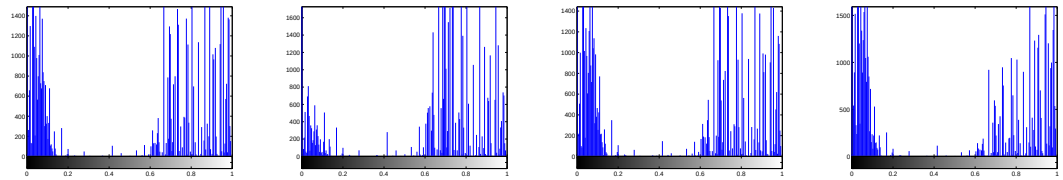
(i) Class 1 (Saturation) (j) Class 2 (Saturation) (k) Class 3(Saturation) (l) Class 4 (Saturation)

(m) Class 5(Saturation) (n) Class 6(Saturation) (o) Class 7(Saturation) (p) Class 8(Saturation)

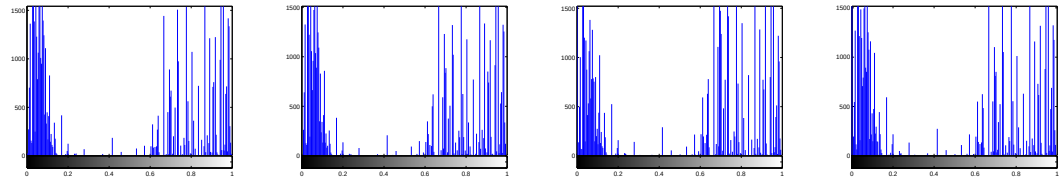
(q) Class 1 (Intensity) (r) Class 2 (Intensity) (s) Class 3 (Intensity) (t) Class 4 (Intensity)

(u) Class 5 (Intensity) (v) Class 6 (Intensity) (w) Class 7 (Intensity) (x) Class 8 (Intensity)

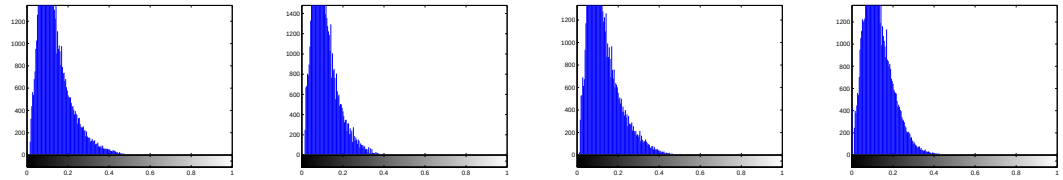
Figure 7.30: Hue, saturation and intensity (value) images from the HSV colour map of the colour tea images.



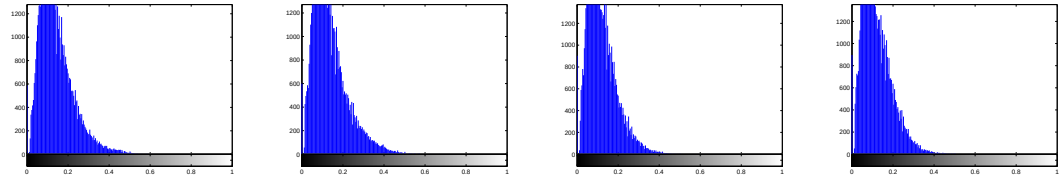
(a) Class 1 (Hue) (b) Class 2 (Hue) (c) Class 3 (Hue) (d) Class 4 (Hue)



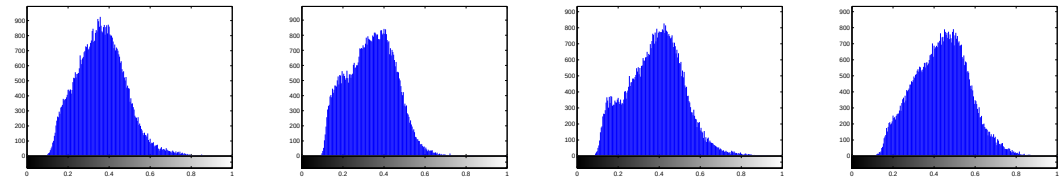
(e) Class 5 (Hue) (f) Class 6 (Hue) (g) Class 7 (Hue) (h) Class 8 (Hue)



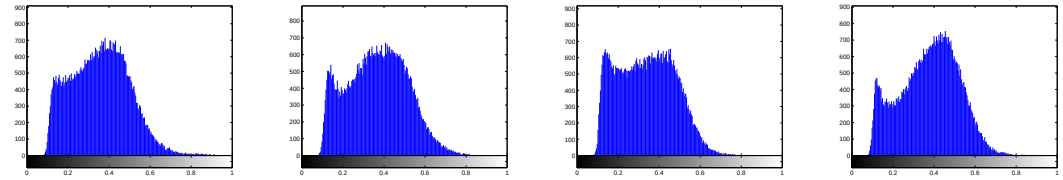
(i) Class 1 (Saturation) (j) Class 2 (Saturation) (k) Class 3 (Saturation) (l) Class 4 (Saturation)



(m) Class 5 (Saturation) (n) Class 6 (Saturation) (o) Class 7 (Saturation) (p) Class 8 (Saturation)



(q) Class 1 (Intensity) (r) Class 2 (Intensity) (s) Class 3 (Intensity) (t) Class 4 (Intensity)



(u) Class 5 (Intensity) (v) Class 6 (Intensity) (w) Class 7 (Intensity) (x) Class 8 (Intensity)

Figure 7.31: Histograms of the hue, saturation and intensity (value) images using 256 bins from the HSV colour map of the colour tea images. One bar is used for each bin.

Chapter 8

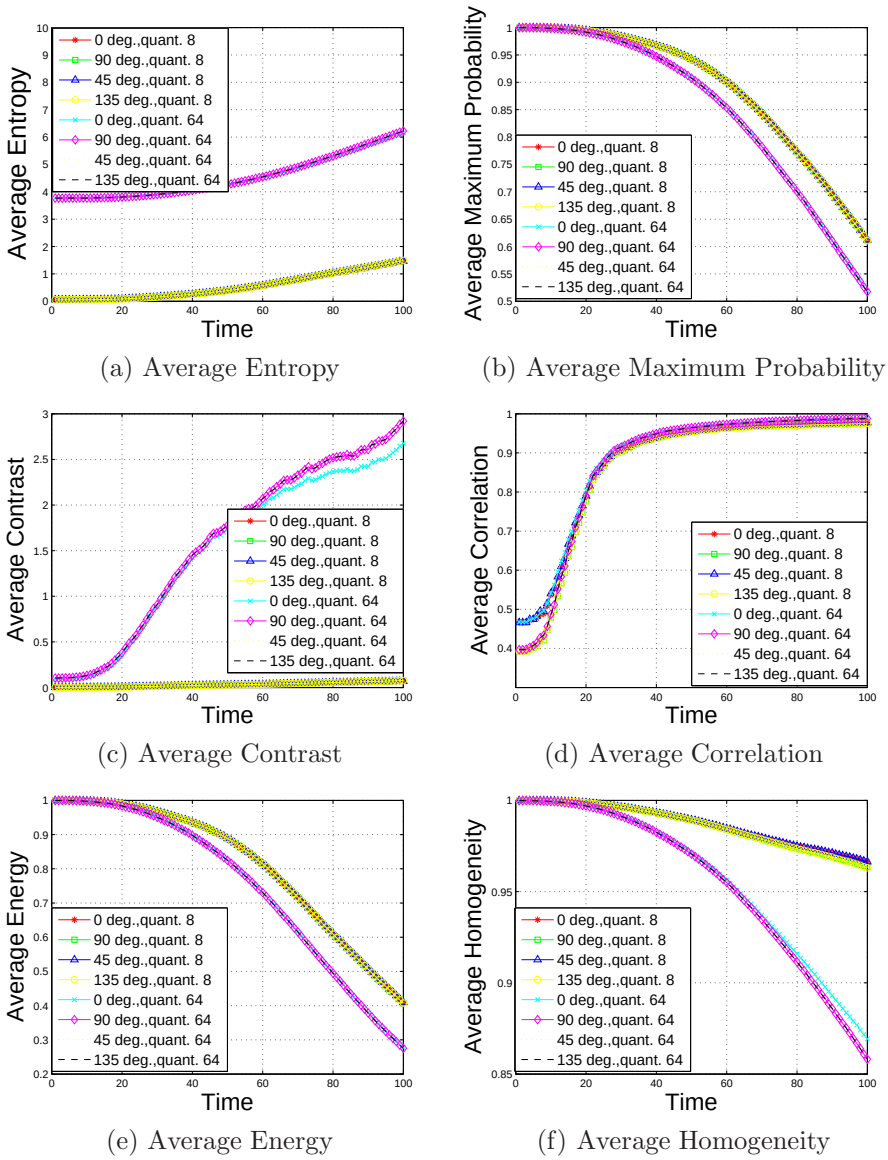
GLCM and Wavelet-based Classification

In this chapter we derive the grey level co-occurrence matrix (GLCM) and wavelet-based texture features from the synthetic images of pyramids and ellipses as well as the real images of corrosion and tea leaves. We classify these images according to their evolution time or class label by coupling these features with the different classifiers and compare the classification performance of granulometric moments with GLCM and wavelet-based features.

8.1 GLCM Features for Synthetic Images

A GLCM contains information about the positions of pixels having similar grey level values. To compute a GLCM, we first quantise the image intensities into a smaller number of levels. In this work we have quantised the usual 256^2 grey levels to 8 and 64 and we use a displacement of $d = 1$. The GLCM values $C(i, j)$ are obtained by counting all pairs of pixels, separated by distance d and at a given angle to each other, which have grey levels i and j. Then the normalised GLCM is obtained by dividing each entry by the sum of all entries, to give relative frequencies. Among many possible features, we computed six GLCM texture features, namely entropy, maximum probability, contrast, correlation, energy and homogeneity. See Section 3.5 for details.

Firstly we computed GLCM features from 100 stacks of 256^2 pyramid images using quantisation levels 8 and 64 and four different orientations, i.e. $0°, 90°, 45°$ and $135°$ but using the same inter-pixel distance $d = 1$ for both cases. The average GLCM features were obtained by averaging the values obtained over the 100 different simulations/stacks, and these are shown plotted against time



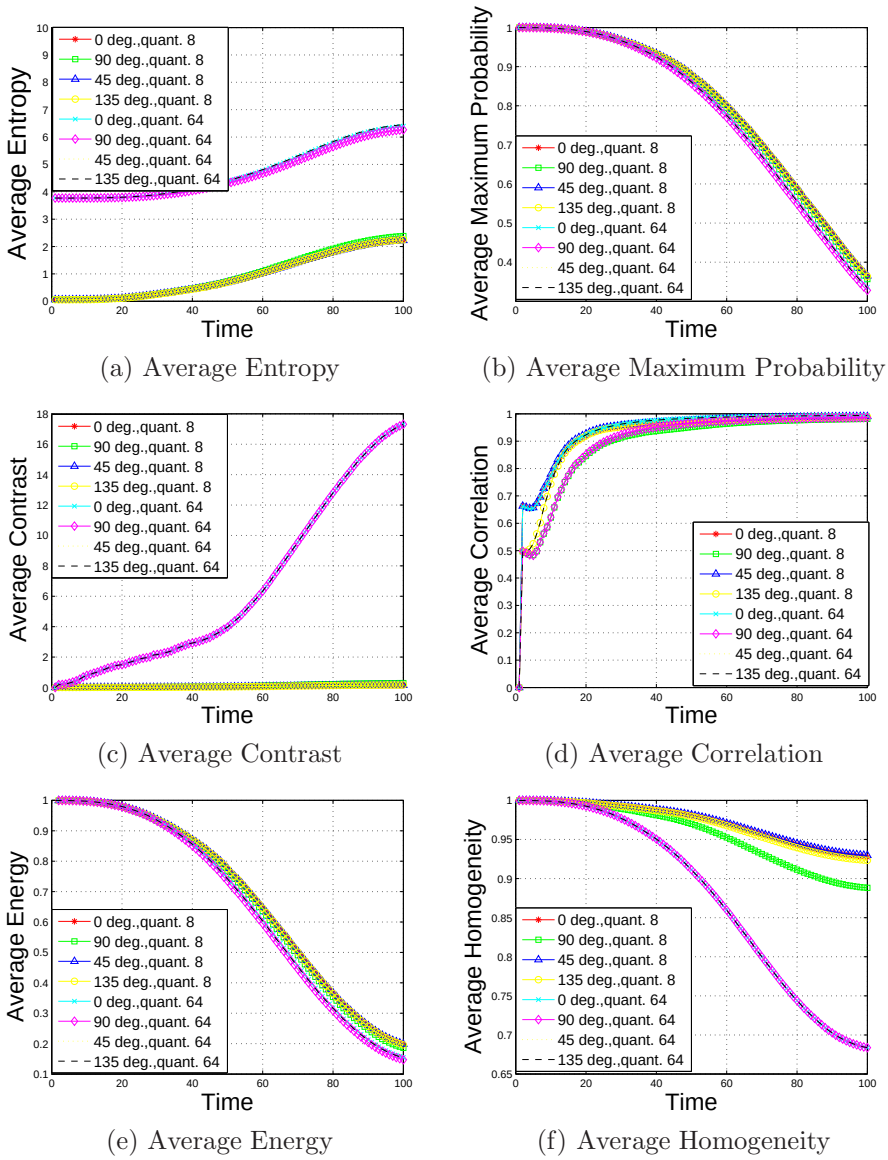
(a) Average Entropy (b) Average Maximum Probability

(c) Average Contrast (d) Average Correlation

(e) Average Energy (f) Average Homogeneity

Figure 8.1: Plots of the average GLCM features against evolution time using quantisation levels 8 and 64, averaged over 100 256^2 pyramid images at each time point (distance=1, and orientation = 0°, 90°, 45° and 135°).

in Figure 8.1. The features are invariant under orientation but change with quantisation level. Although the features computed at different orientations show a similar pattern, they vary in magnitude. Entropy is higher for quantisation level 64 than level 8, but both sets of entropy values have similar trends with time. Maximum probability, energy and homogeneity decrease over time for both quantisations, but less quantisation (level 64) produces relatively lower features. More quantisation (level 8) produces near-zero contrast, whereas it increases with time for level 64. Average correlations for both quantisations are alike except for times $t < 10$.



(a) Average Entropy (b) Average Maximum Probability

(c) Average Contrast (d) Average Correlation

(e) Average Energy (f) Average Homogeneity

Figure 8.2: Plots of the average GLCM features against evolution time using quantisation levels 8 and 64, averaged over 100 256^2 ellipse images at each time point (distance=1, and orientation = 0°, 90°, 45° and 135°).

Figure 8.2 shows the same GLCM feature plots for the ellipse images. Again all features except homogeneity are invariant under orientation but some change with quantisation level. Now maximum probability and energy for both quantisations are similar. The entropies for both quantisations have a similar relationship with time, although level 64 quantisation produces higher values. It can be seen that contrast for level 8 is very low, while it increases with time for level 64. Homogeneity for level 64 coincides for all orientations, but for level 8 angles 0°, 45° and 135° produce similar features, whereas 90° produces relatively lower homogeneity for times $t > 30$.

Since the GLCM features for both quantisation levels produce similar features and trends at different orientations, we used all 6 features at quantisation level 8 with 135° orientation. So the feature data consists of a 10000×6 matrix and the average feature data is of size 100×6 (averaged over 100 simulations). To compare the classification results using the GLCM features with those from the granulometric moments (as in Sections 5.4 and 5.5) we used all the classifiers used there. Again we randomly selected 70% of the features to train each classifier and tested its classification ability using the rest of the features. The process was repeated 10 times to obtain average error rates and MAEs.

8.1.1 Classification using GLCM features

All 6 GLCM features at 135° orientation were used to predict evolution time for the pyramid images. The regression classifier was built using average features, averaged over the training samples, and the prediction was done using features from an individual test image.

For SVMs a linear kernel with different values of cost was evaluated, and it was found that for the linear kernel any cost of 20 or above produced 100% correct classification (Table 8.1). A cost of 100 was used in the computation. Neither a radial basis kernel nor a polynomial kernel was able to attain 100% correct classification, so these are not shown here. For a single hidden layer FF-NNET the optimum values of decay and rang were 10^{-3} and 0.1 for the pyramid images and 10^{-4} and 0.1 for the ellipse images respectively, using normalised data. With this parameter setting, 7 hidden units produced the minimum training set error rate using a single training set (Table 8.2). Therefore 7 units were used in the final computation. LDA was also used to predict evolution times.

Table 8.1: Finding the optimum value of the cost using a linear kernel in SVM with 6 GLCM features from the pyramid and ellipse images; the table shows training set error rate.

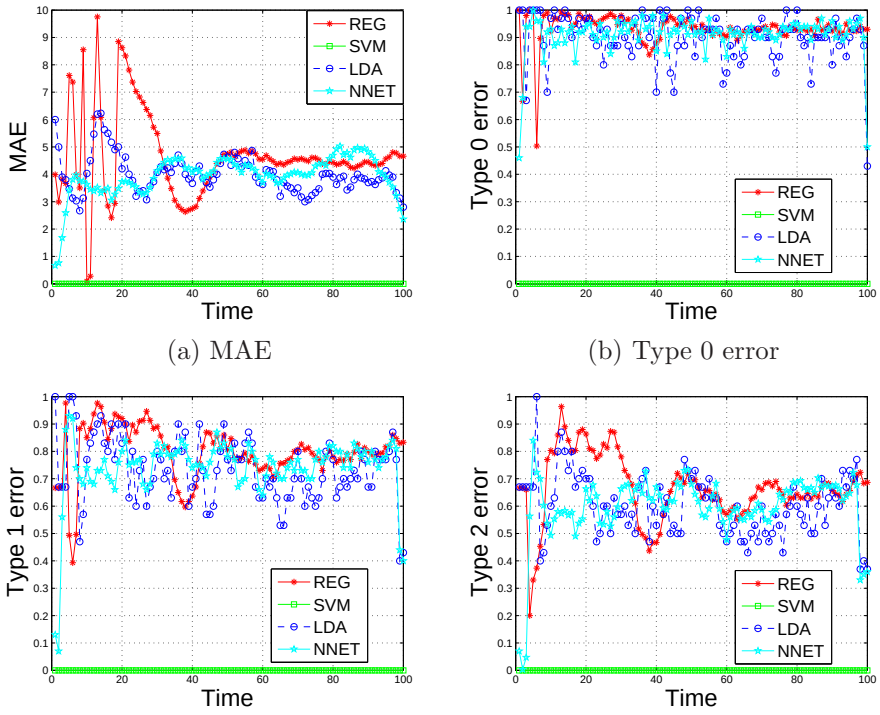
	Pyramid images										
Cost	1	10	20	30	40	50	60	70	80	90	100
Error rate	0.81	0.21	0	0	0	0	0	0	0	0	**0**
	Ellipse images										
Cost	1	10	20	30	40	50	60	70	80	90	100
Error rate	0.85	0.35	0	0	0	0	0	0	0	0	**0**

Error rates and MAEs for all classifiers are shown in Figure 8.3 and Table 8.3.

Table 8.2: Training set error rate from a grid search approach for finding the optimum number of hidden neurons in the FF-NNET using 6 GLCM features from the pyramid and ellipse images.

	Pyramid images						
No. of units	1	2	3	4	5	6	7
Error rate	0.92	0.90	0.90	0.90	0.90	0.90	**0.89**
	Ellipse images						
No. of units	1	2	3	4	5	6	7
Error rate	0.86	0.85	0.84	0.84	0.84	0.83	**0.82**

The regression classifier produced a 93.2% overall classification error rate which is comparable with LDA (90.1%) and FF-NNET (89.1%). MAEs for all classifiers (Figure 8.3(a)) are almost equal for all except SVM after time 30, and SVM produced an MAE of 0 through the entire evolution period as it classified perfectly. The other classifiers are all poor. Comparing Figures 8.3 and 5.18, the performance of the GLCM features is slightly better than that of the granulometric moments for these synthetic pyramid images.

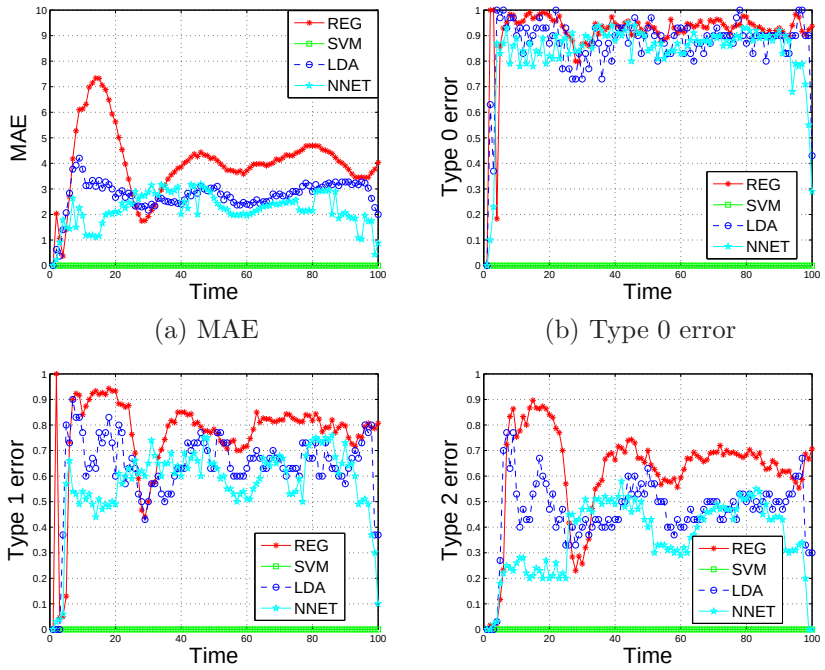


(a) MAE (b) Type 0 error

(c) Type 1 error (d) Type 2 error

Figure 8.3: Mean absolute error (a), type 0, type 1 and type 2 error (b)–(d), using 6 GLCM features in different classifiers, for the pyramid images.

Similarly, all 6 GLCM features from the ellipse images, at quantisation level 8, were used to classify the time of these images. For this dataset, again SVM with a linear kernel and any cost of 20 of more produced 100% classification accuracy (Table 8.1), but we used a cost of 100, and 7 units in the FF-NNET with decay of 10^{-3} and rang of 0.1, as this produced a slightly lower error rate (Table 8.2). MAEs and error rates are shown in Figure 8.4 and Table 8.3. Again SVM classifies perfectly. Average error rates for the regression classifier, LDA and FF-NNET are poor at 91.7%, 86.7% and 83.5% respectively. Again comparing Figures 8.4 and 5.19, we conclude that GLCM features provide slightly better classification results than the granulometric moments for the ellipse images.



(a) MAE (b) Type 0 error

(c) Type 1 error (d) Type 2 error

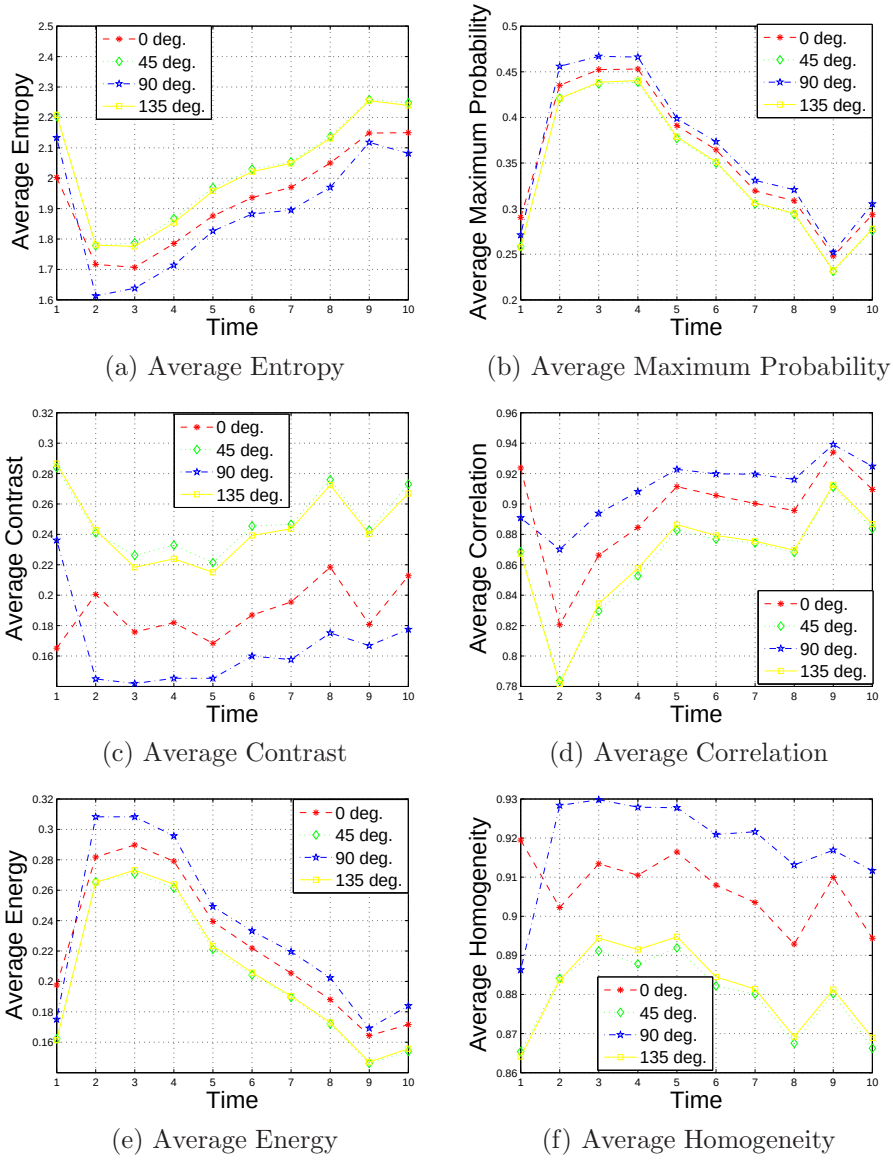
Figure 8.4: Mean absolute error (a), type 0, type 1 and type 2 error (b)–(d), using 6 GLCM features in different classifiers, for the ellipse images.

Table 8.3: Average test set error rates as proportions and MAEs using 6 GLCM features at quantisation level 8 for the synthetic images.

Error rate	Pyramid images				Ellipse images			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
Type 0	0.932	0	0.901	0.891	0.917	0	0.867	0.835
Type 1	0.796	0	0.734	0.702	0.764	0	0.636	0.574
Type 2	0.664	0	0.600	0.540	0.621	0	0.462	0.364
MAE	6.634	0	3.996	3.443	4.017	0	2.752	2.188

8.2 GLCM Features for Corrosion Images

We computed GLCMs from each of the 256^2 grey scale corrosion images directly, as applying GLCM to the bottom-hat transformed images used previously produced constant features over time for all sub-images. We used GLCMs based on a single inter-pixel distance, the four different orientations as above, and the two different quantisations (level 8 and 64) and computed the 6 features again, namely entropy, maximum probability, contrast, correlation, energy and homogeneity.



(a) Average Entropy

(b) Average Maximum Probability

(c) Average Contrast

(d) Average Correlation

(e) Average Energy

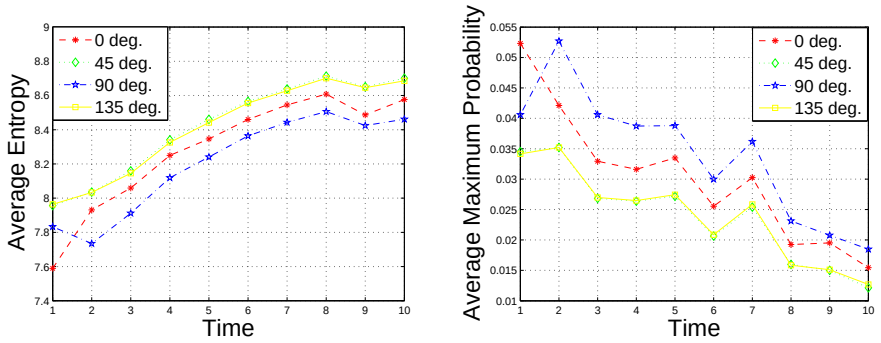
(f) Average Homogeneity

Figure 8.5: Average GLCM features for the grey level corrosion images for quantisation 8 using $d = 1$ and four orientations (0°, 90°, 45° and 135°).

The features from quantisation level 8 are shown in Figure 8.5. Contrast and homogeneity exhibit an irregular pattern with time. Entropy, maximum

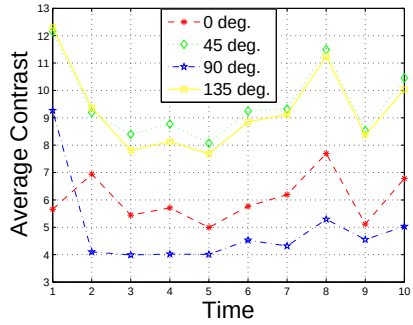
probability, correlation and energy show a more useful trend with time. Although each of these four features computed using different orientations show similar time trends, the features using 45° and 135° orientations are close to each other.

All 6 GLCM features using a unit inter-pixel distance as above, but with level 64 quantisation, are shown in Figure 8.6. Energy and entropy show a much smoother trend than for level 8. In this case, although correlation increases with time, the trend is not is smooth as the others and contrast shows random behaviour with time. Again features computed using 45° and 135° orientations are closer to each other.

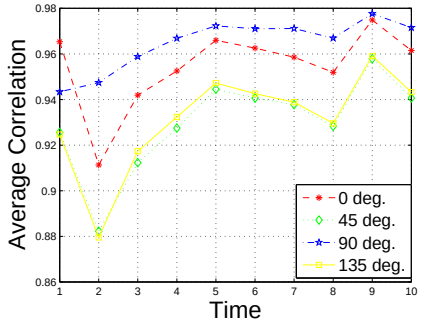


(a) Average Entropy

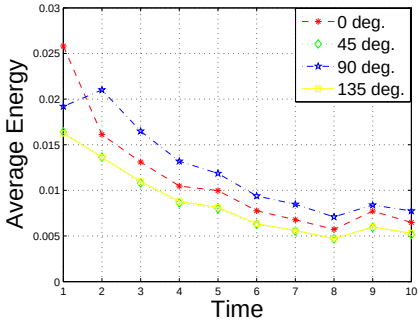
(b) Average Maximum Probability



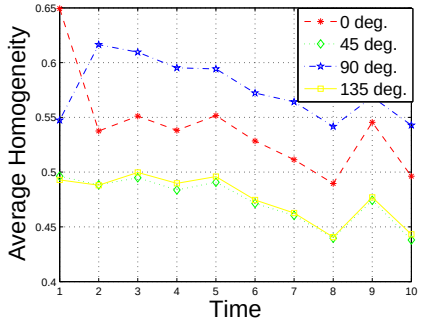
(c) Average Contrast



(d) Average Correlation



(e) Average Energy



(f) Average Homogeneity

Figure 8.6: Average GLCM features for the grey level corrosion images for quantisation 64 using $d = 1$ and four orientations (0°, 90°, 45° and 135°).

Since features computed at different orientations show similar time trends we used only features from 135° orientation for classification. For quantisation level 8, contrast and homogeneity, and for quantisation level 64 contrast and correlation do not show a clear trend with time. Hence we used the other 4 features in each case in all the classifiers. Using a single training set we again sought the best kernel and its parameter values for SVM, and the optimum values of decay, rang and the number of units in the hidden layer of the FF-NNET to minimise error rates.

Table 8.4: Training set error rates from a grid search for finding the optimum values of cost and parameter γ for the radial basis kernel and a polynomial kernel with $\eta = 1$ in SVM using 4 GLCM features at quantisation level 8, and for a linear kernel using 4 GLCM features from both quantisations, from the corrosion images.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.63	0.40	0.23	0.17	0.10	0.10	0.10	0.10	0.10	0.10	0.10
0.5	0.50	0.33	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
0.9	0.60	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.67	0.17	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.5	0.23	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
0.9	0.27	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
	Linear kernel										
	Cost										
	1	10	20	30	40	50	60	70	80	90	100
Level 8	0.33	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	**0.03**
Level 64	0.23	0	0	0	0	0	0	0	0	0	**0**

All possible values of γ between 0 to 1, in steps of size 0.1 were investigated and the error rates corresponding to $\gamma = 0.1,\ 0.5$ and 0.9 are shown in Table 8.4. At quantisation level 8, neither the radial basis kernel nor polynomial kernel was able to produce 100% correct classification as the lowest error rate was 10% with $\gamma = 0.1$ (or $\gamma = 0.2$, not shown here) and cost of 40 or above for the radial basis kernel, and the lowest error rate for a polynomial kernel was 3% for $\gamma = 0.1$ with any cost of 20 or more. For the polynomial kernel, the parameter value $\eta = 0$ (the default value) produced higher error than any η between 1 and 5, which produced

the same error, so $\eta = 1$ was used. Error rates for the radial basis kernel and a polynomial kernel were similar for features at quantisation level 64, hence are not shown here. A linear kernel with any cost of 10 or above produced only 3% error for the features at quantisation level 8 (Table 8.4). Although a polynomial kernel with $\gamma = 0.1$ or 0.2 and cost of 40 or above performed as well as the linear kernel with cost of 20 or more for the features computed at quantisation level 8, a linear kernel produced 100% correct classification for the features computed at quantisation level 64 with any cost of 10 or more. We used a linear kernel with a cost of 100 for both feature sets.

Table 8.5: Training set error rate from a grid search approach for finding the optimum number of hidden neurons for FF-NNET using 4 GLCM features from the corrosion images.

	Quantisation level 8									
No. of units	1	2	3	4	5	6	7	8	9	10
Error rate	0.41	0.34	0.27	0.16	0.01	0.03	0	0	0	**0**
	Quantisation level 64									
No. of units	1	2	3	4	5	6	7	8	9	10
Error rate	0.29	0.07	0.01	0	0	0	0	0	0	**0**

For the FF-NNET using the normalised features for quantisation level 8, the best values of decay and rang were 10^{-4} and $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, whereas for the features at quantisation level 64 they were 10^{-3} and $1/\max(|\boldsymbol{x}|)$ respectively. For both sets of features 10 units was best as it produced the lowest training set error rate (Table 8.5) and greater generalisability. Hence, 10 hidden units were used for both feature sets.

Again 70% of the features were used for training and the rest for testing, averaging results over 10 runs. In this case the images were sometimes misclassified by more than one unit, so in general type 0 error rates and MAEs are different. Error rates for all classifiers for both quantisations are shown in Table 8.6. For quantisation level 8, type 0 error rates for REG, LDA and FF-NNET are 92.0%, 70.7% and 79.3%, but only 2.7% for SVM. Quantisation level 64 has lower error rates than level 8 for all classifiers. The corresponding MAEs for all classifiers are also lower for level 64 than level 8. The error rates are noticeably higher for all classifiers than when using the granulometric moments (see Table 6.8).

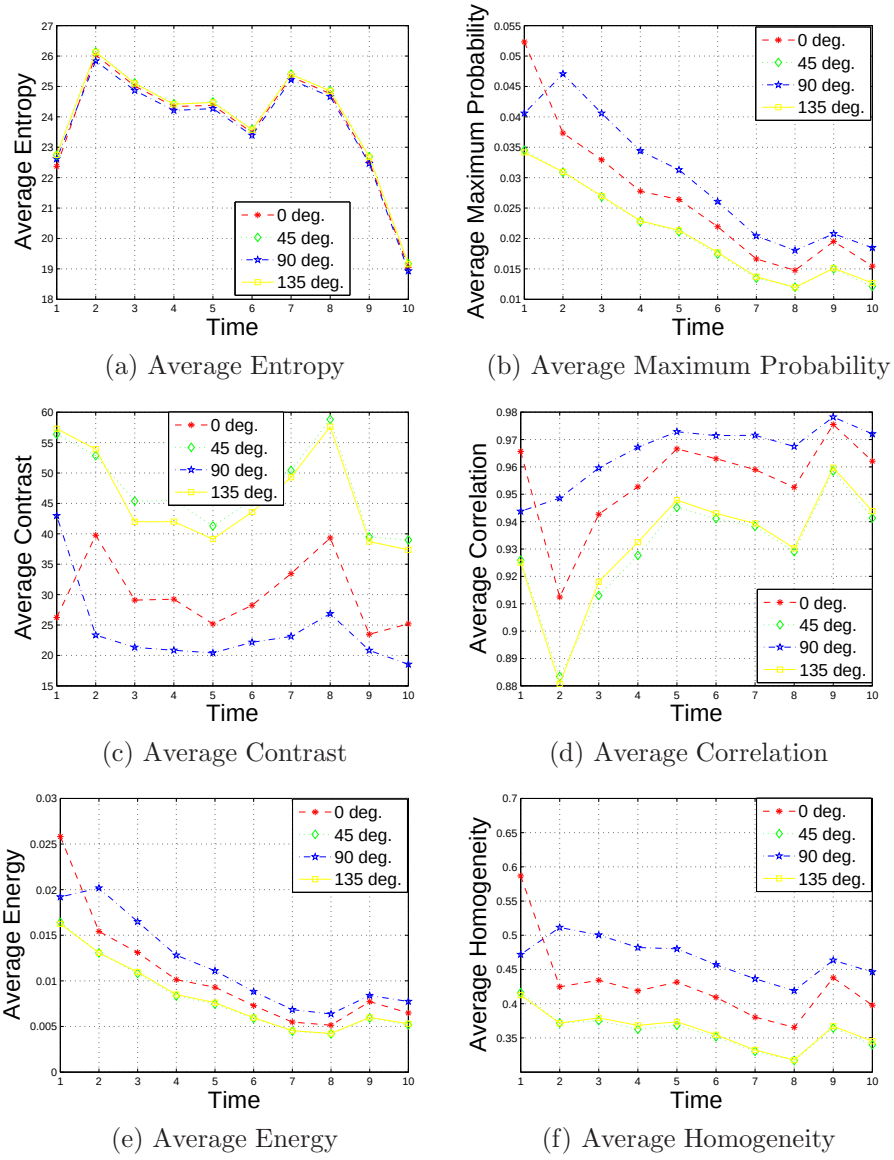
We also extracted 6 GLCM features without quantisation, shown in Figure 8.7. Entropy and contrast were excluded as they do not show such a clear trend with time, and the other features were used for classification. A linear kernel with

Table 8.6: Average test set Type 0 error rates and MAEs for all classifiers using 4 GLCM features with quantisation levels 8 and 64, for the corrosion images; results are averaged over 10 runs.

	Type 0 errors							
Time	Quantisation at level 8				Quantisation at level 64			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.300	0.100	0.867	0.933	0.417	0.000	0.000	0.267
2	0.900	0.133	0.367	0.433	0.333	0.000	0.467	0.567
3	1.000	0.067	0.867	0.867	0.667	0.000	0.733	0.800
4	1.000	0.000	0.800	0.733	1.000	0.000	0.767	0.867
5	1.000	0.000	0.900	0.867	1.000	0.000	0.900	0.733
6	1.000	0.000	1.000	0.733	1.000	0.000	0.867	0.767
7	1.000	0.000	0.733	0.800	1.000	0.000	0.500	0.533
8	1.000	0.000	0.767	0.933	1.000	0.067	0.367	0.567
9	1.000	0.000	0.300	0.833	1.000	0.100	0.300	0.500
10	1.000	0.000	0.467	0.800	1.000	0.000	0.333	0.600
Overall	0.920	0.027	0.707	0.793	0.842	0.017	0.523	0.620
	MAEs							
Time	Quantisation at level 8				Quantisation at level 64			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	1.767	0.100	5.167	5.533	2.667	0.000	0.000	0.500
2	8.000	0.133	0.900	1.167	3.733	0.000	1.000	1.133
3	6.300	0.067	1.400	1.533	3.367	0.000	0.767	1.133
4	5.800	0.000	1.700	1.433	4.367	0.000	1.700	1.833
5	4.767	0.000	2.067	1.967	5.133	0.000	2.833	1.733
6	4.367	0.000	1.833	1.833	4.967	0.000	1.867	1.467
7	5.533	0.000	2.267	2.533	5.033	0.000	1.000	0.933
8	6.900	0.000	2.733	3.100	6.500	0.067	0.667	1.500
9	8.100	0.000	1.567	3.233	5.567	0.100	0.800	0.767
10	9.000	0.000	1.600	2.567	7.433	0.000	0.633	1.733
Overall	6.053	0.027	2.12	2.490	4.877	0.017	1.127	1.273

a cost of 100 for SVM and 10 hidden units with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$ for FF-NNET were used, as they produced the lowest training error rates. Type 0 error rates and MAEs for all classifiers are recorded in Table 8.7. Overall the regression approach performs better than for quantisation levels 8 and 64 (80.0% error rate). SVM has a 1.3% error rate, which is again better than for both levels of quantisation. LDA and FF-NNET results are (at 59% and 62.7% respectively) better than for quantisation 8 but worse than for quantisation level 64. MAE from no quantisation is higher than for both quantisations for the regression approach but better for SVM. For LDA, no quantisation produces

better MAE than both quantisation levels, whereas for FF-NNET no quantisation yields better results than quantisation level 8 but worse than level 64.



(a) Average Entropy

(b) Average Maximum Probability

(c) Average Contrast

(d) Average Correlation

(e) Average Energy

(f) Average Homogeneity

Figure 8.7: Average GLCM features for the grey level corrosion images for no quantisation using $d = 1$ and four orientations (0°, 90°, 45° and 135°).

We also tried using all 15 GLCM features mentioned in Section 3.5 at quantisation levels 8 and 64 and also without quantisation. Again for SVM, a linear kernel with a cost of 100 and 10 hidden units with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$ for FF-NNET (with normalised features) were used. Type 0 error rates and MAEs are shown in Table 8.8. Use of 15 GLCM features severely affects the performance of SVM as it now has 24.7%, 31.4% and 41% error rates for quantisation level 8, level 64 and no quantisation respectively. LDA works slightly better with more GLCM features and some quantisation but worse with

Table 8.7: Average test set Type 0 error rates and MAEs for all classifiers using 4 GLCM features without quantisation, for the corrosion images; results are averaged over 10 runs.

Time	Type 0 error				MAE			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	1.000	0.067	0.033	0.267	12.000	0.067	0.033	0.567
2	1.000	0.033	0.433	0.467	14.333	0.033	0.700	1.200
3	0.667	0.000	0.933	0.900	1.667	0.000	1.400	1.533
4	0.333	0.000	0.867	0.700	1.667	0.000	2.067	1.833
5	0.667	0.000	0.800	0.833	1.667	0.000	1.500	2.000
6	0.667	0.000	0.500	0.767	2.333	0.000	0.767	1.533
7	1.000	0.000	0.767	0.767	3.333	0.000	0.900	1.133
8	1.000	0.033	0.467	0.500	5.000	0.033	0.733	1.067
9	1.000	0.000	0.567	0.433	2.333	0.000	1.467	0.867
10	0.667	0.000	0.533	0.633	4.667	0.000	1.067	2.367
Overall	0.800	0.013	0.590	0.627	8.167	0.013	1.063	1.410

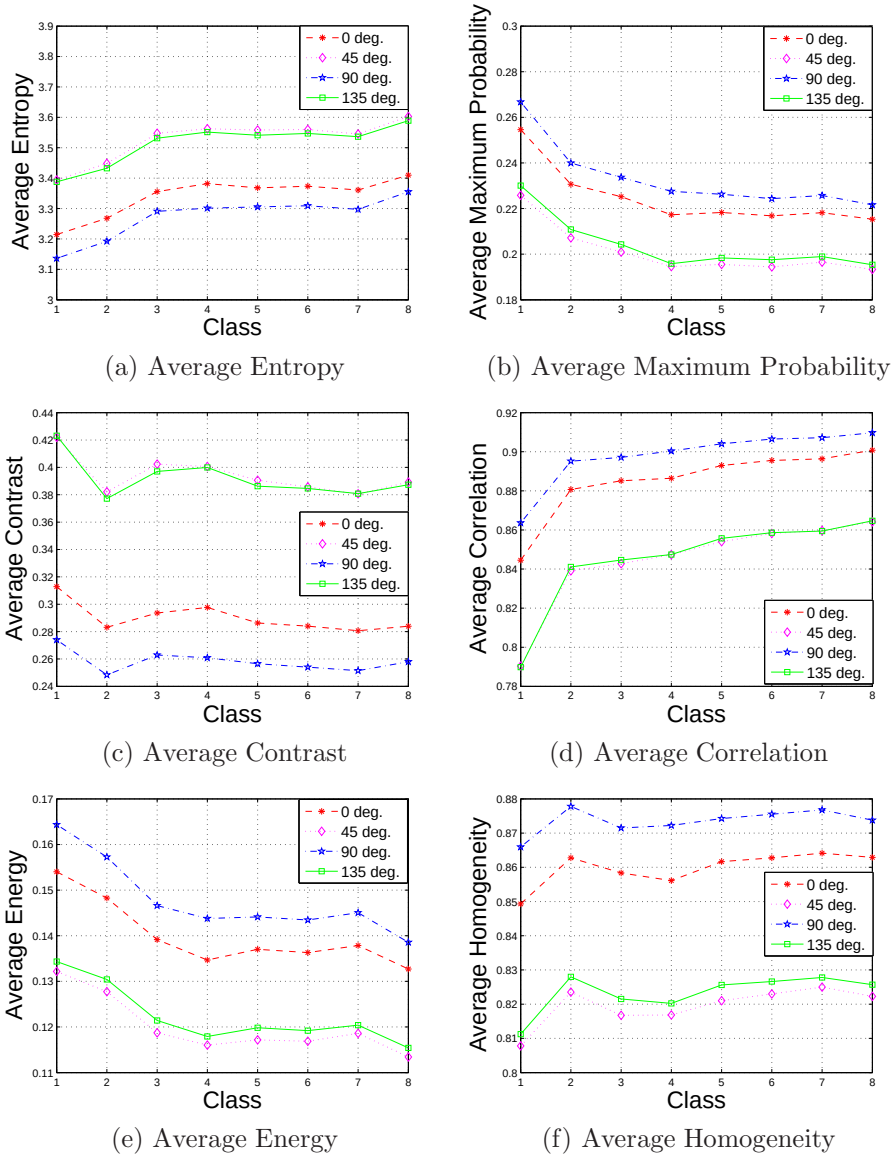
more. FF-NNET works better with more features at level 8 but not level 64 or with no quantisation. The regression approach produces worse results for both quantisation levels or no quantisation with more GLCM features.

Table 8.8: Average test set Type 0 error rates and MAEs for all classifiers using all 15 GLCM features at quantisation levels 8 and 64 and without quantisation for the corrosion images; results are averaged over 10 runs.

	Quantisation level 8							
Time	Type 0 errors				MAEs			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.333	0.167	0.333	0.267	1.000	0.333	1.300	1.100
2	1.000	0.233	0.300	0.733	8.333	0.233	1.000	2.967
3	1.000	0.133	0.567	0.967	2.333	0.133	0.733	2.933
4	1.000	0.233	0.733	0.867	3.333	0.367	1.067	2.367
5	1.000	0.367	0.933	0.733	6.000	0.500	1.700	1.767
6	1.000	0.133	0.967	0.833	7.667	0.133	1.867	2.000
7	1.000	0.267	0.800	0.867	7.333	0.400	1.300	2.000
8	1.000	0.400	0.600	0.767	6.000	0.700	1.733	2.833
9	1.000	0.267	0.433	0.633	10.000	0.333	1.267	2.700
10	1.000	0.267	0.300	0.367	11.333	0.300	0.767	1.500
Overall	0.933	0.247	0.597	0.703	6.333	0.343	1.273	2.217
	Quantisation level 64							
Time	Type 0 errors				MAEs			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.667	0.167	0.233	0.633	1.333	0.200	0.900	4.100
2	1.000	0.500	0.333	0.633	8.333	0.567	1.033	1.867
3	1.000	0.400	0.567	0.567	2.333	0.433	0.633	1.067
4	1.000	0.400	0.733	0.800	4.333	0.533	1.100	1.267
5	1.000	0.367	0.633	0.833	3.667	0.667	1.367	1.800
6	1.000	0.233	0.800	0.633	5.333	0.233	1.433	1.467
7	1.000	0.267	0.467	0.967	7.333	0.267	0.733	2.533
8	1.000	0.267	0.367	0.800	6.667	0.367	0.867	2.267
9	0.667	0.367	0.333	0.667	4.667	0.533	0.833	2.133
10	1.000	0.167	0.367	0.767	12.000	0.233	0.767	4.200
Overall	0.933	0.314	0.483	0.730	5.600	0.403	0.967	2.270
	Without quantisation							
Time	Type 0 errors				MAEs			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.333	0.067	0.233	0.500	0.333	0.067	1.633	3.167
2	0.667	0.400	0.500	0.500	1.000	1.000	2.267	1.767
3	0.333	0.533	0.800	0.500	0.333	0.567	1.800	0.900
4	1.000	0.500	0.900	0.867	6.333	1.333	2.400	1.967
5	1.000	0.467	0.833	0.800	3.667	1.067	2.100	1.933
6	1.000	0.433	0.900	0.867	4.000	1.300	2.533	2.367
7	1.000	0.667	0.800	0.800	4.667	1.533	1.533	2.067
8	1.000	0.333	0.733	0.700	5.667	0.733	2.167	2.600
9	1.000	0.433	0.300	0.733	7.333	2.200	1.333	3.533
10	1.000	0.267	0.367	0.533	8.000	0.767	0.633	3.233
Overall	0.833	0.410	0.637	0.680	4.133	1.057	1.840	2.355

8.3 GLCM Features for the Tea Images

We obtained GLCMs from each of the original untransformed 256^2 grey scale tea images and computed the 6 GLCM features as above. As there are 50 sub-images from each class $1, \ldots, 8$, the feature data is of size 400×6. Again we used a unit inter-pixel distance $d = 1$, four different orientations and two levels of quantisations (8 and 64).



(a) Average Entropy (b) Average Maximum Probability

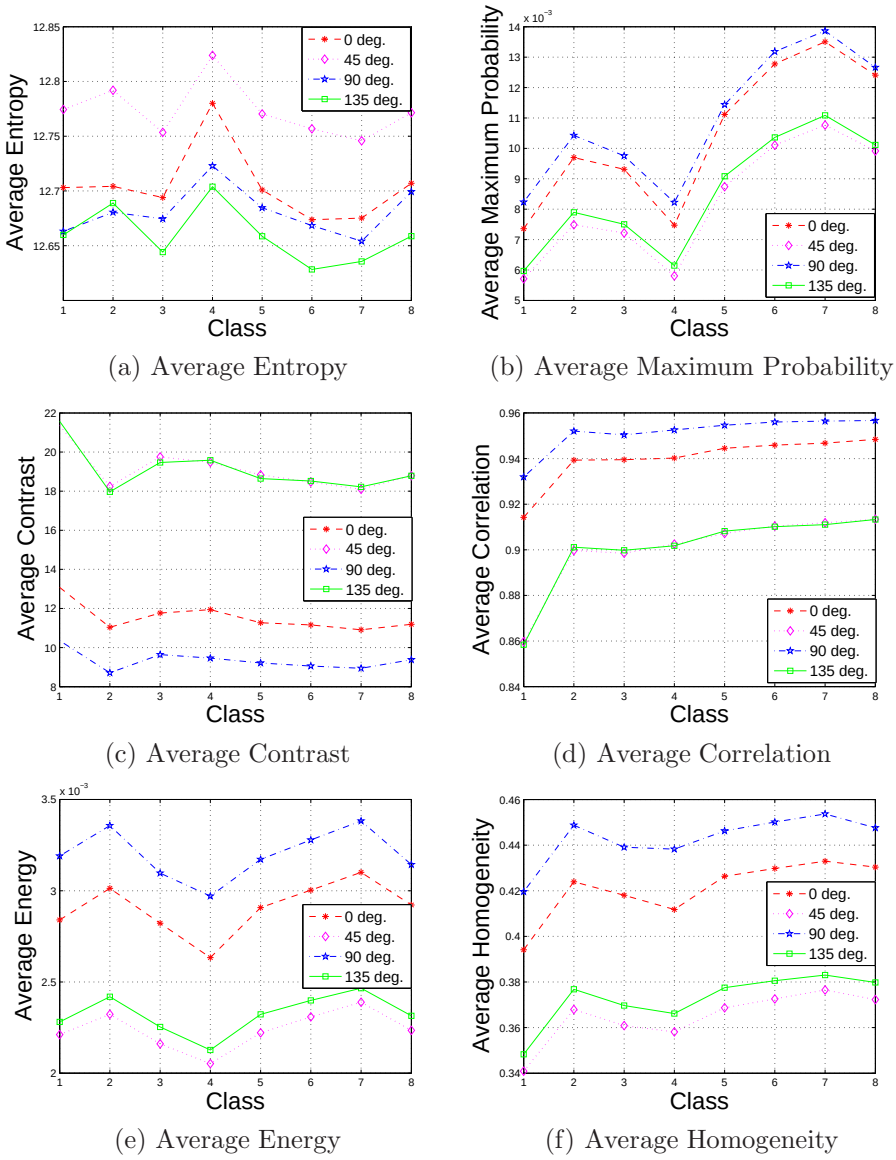
(c) Average Contrast (d) Average Correlation

(e) Average Energy (f) Average Homogeneity

Figure 8.8: Average GLCM features against class for the grey scale tea images for quantisation 8 using $d = 1$ and four orientations (0°, 90°, 45° and 135°).

Figure 8.8 shows the relationship of average GLCM features for quantisation level 8 with class. Average entropy and correlation increase with class, and average maximum probability and energy decrease over the classes, but there is no

clear trend for average contrast or homogeneity with class. The features were also computed for level 64 quantisation, and are shown in Figure 8.9. None of them have any clear trend with class but we considered maximum probability, contrast, correlation and homogeneity for classification purposes. In general, quantisation level 8 produces smoother trends than level 64. For both quantisations, the features show similar trends with different orientations.



(a) Average Entropy (b) Average Maximum Probability

(c) Average Contrast (d) Average Correlation

(e) Average Energy (f) Average Homogeneity

Figure 8.9: Average GLCM features against class for the grey scale tea images for quantisation 64 using $d = 1$ and four orientations (0°, 90°, 45° and 135°).

8.3.1 PCA on GLCM features for the tea images

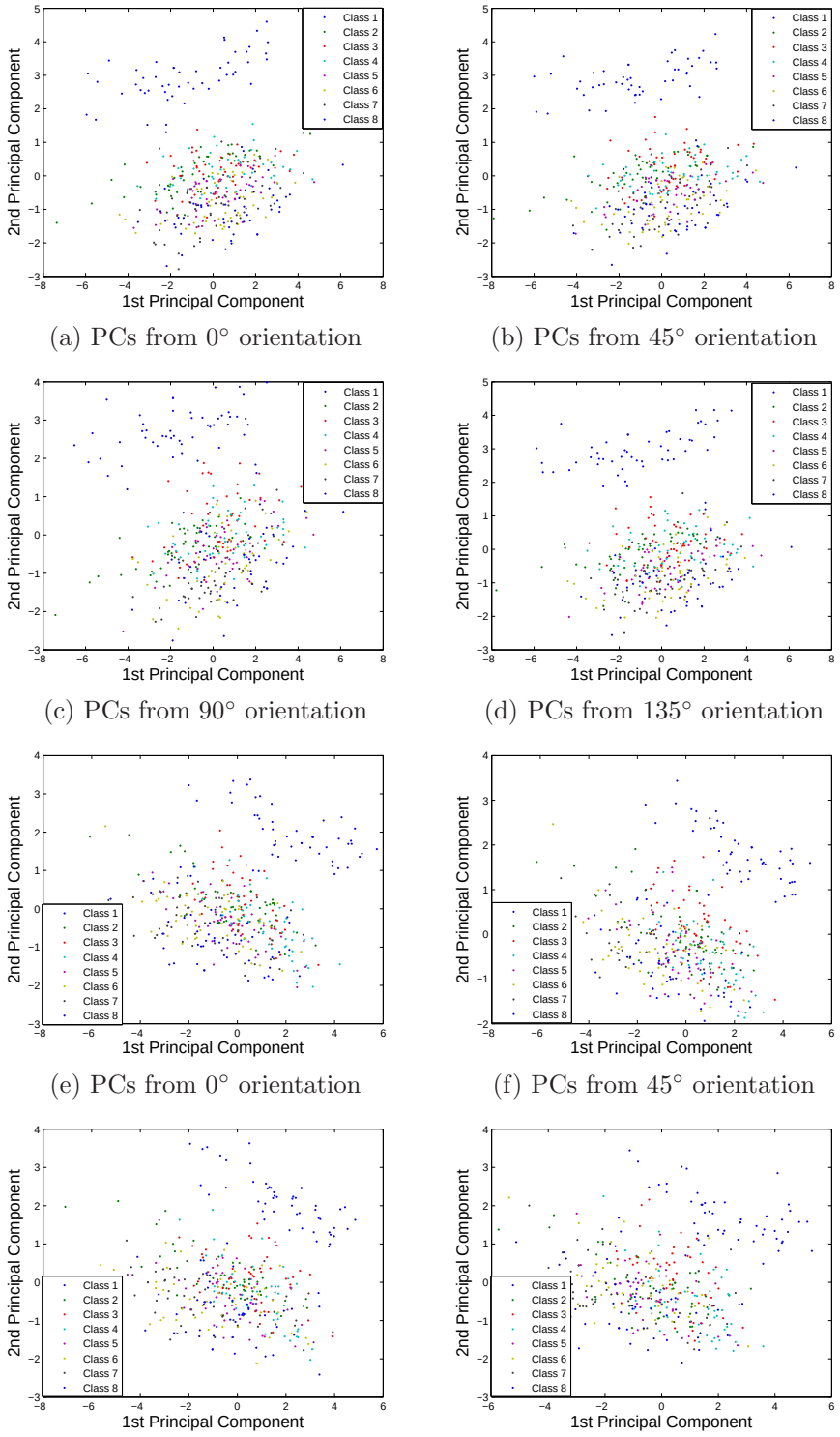
PCA was applied to the best four GLCM features for both quantisations as described above, and all four orientations. Figure 8.10 shows the scatterplots of the first two PCs derived from the GLCM features computed for all orientations at quantisation levels 8 and 64. Features computed at level 8 for different orientations have similar discrimination ability, as they are only able to distinguish class 1 clearly and the other classes overlap. The results for quantisation level 64 indicate the same. The PCs from PS moments using either a disk or a square SE clearly distinguished all 8 classes, shown in Figure 7.20, unlike the PCs from the GLCM features at either quantisation level. Therefore, GLCM features are unlikely to give good performance in any classifier.

8.3.2 Classification using supervised classifiers

We used the regression approach, SVMs, LDA and FF-NNET, as before to classify the tea images. Again we trained each classifier using 70% of the features and tested for the rest and computed the average error rates and MAEs over 10 runs. As in Chapter 7 for the Indian tea images, we clipped predictions to lie in the range of classes 1 to 8.

We used 4 GLCM features, entropy, maximum probability, correlation and energy computed at quantisation 8 and 135° orientation, as contrast and homogeneity are more constant over the classes. For quantisation level 64 maximum probability, contrast, correlation and homogeneity at 135° orientation were used for classification.

SVM with a radial basis kernel produced only 1% error for any cost of 10 or above with $\gamma = 0.1$, for both quantisation levels (Table 8.9). We also computed the training set error rate using a polynomial kernel using $\eta = 1 - 5$. For quantisation 8, 100% correct classification can be obtained for any cost of 10 or more with $\gamma = 0.1$, while for quantisation 64 the lowest error rate of 2% can be attained using the same parameter settings. A linear kernel produced 100% correct classification for the training set with any cost between 1 to 100, using the 4 GLCM features at 135° rotation and for both quantisations, so they are not shown in the table. However, we used a linear kernel with cost of 100. LDA was considered without cross-validation. For FF-NNET the best values of decay and rang were found to be 10^{-4} and 0.5 for normalised features at both quantisation levels. Ten hidden units were used for the FF-NNET at both quantisations, as they produced lower training set error rates (Table 8.10).



(a) PCs from 0° orientation

(b) PCs from 45° orientation

(c) PCs from 90° orientation

(d) PCs from 135° orientation

(e) PCs from 0° orientation

(f) PCs from 45° orientation

(g) PCs from 90° orientation

(h) PCs from 135° orientation

Figure 8.10: Scatter plots of the first two PCs using 4 GLCM features for all four orientations at quantisation levels 8 ((a)–(d)) and 64 ((e)–(h)), using the grey scale tea images.

Table 8.9: Training set error rates for different values of cost and γ for radial basis kernel, a polynomial kernel with $\eta = 1$ and a linear kernel in SVM using 4 GLCM features from the tea images with quantisations level 8 and 64.

	Radial basis kernel										
	Cost at quantisation level 8										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.16	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.5	0.07	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.9	0.08	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06
	Radial basis kernel										
	Cost at quantisation level 64										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.16	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.5	0.10	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
0.9	0.13	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09
	Polynomial kernel										
	Cost at quantisation level 8										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.07	0	0	0	0	0	0	0	0	0	0
0.5	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.9	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
	Polynomial kernel										
	Cost at quantisation level 64										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.15	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
0.5	0.08	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
0.9	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10

Type 0 error and MAE in this case are different as the predicted classes are more than one unit away from the actual class. See Table 8.11. Type 0 errors and MAEs for all classifiers are generally higher for quantisation level 64 than level 8, as expected from the plots. SVM attained 100% correct classification for both quantisations. For quantisation level 8, LDA and FF-NNET have 55.9% and 63.2% error rates respectively, and the regression approach has the highest error rate (78.3%), while with quantisation level 64, LDA, FF-NNET and the regression approach gave 57.8%, 65.5% and 85.1% error rates respectively. This pattern of performance is similar to what was found for the corrosion images. However for the tea images the quantisation level 64 produced higher classification error. This is different from the conclusion of Soh et al. (2004) that level 8 quantisation is undesirable and level 64 quantisation is efficient and sufficient, however this

Table 8.10: Training set error rates from a grid search approach for finding optimum number of hidden neurons for FF-NNET using 4 GLCM features for the quantisation level 8 and 64, with rang $= 1/\max(|\boldsymbol{x}|)$, for the tea images.

	Number of units									
Error rate	1	2	3	4	5	6	7	8	9	10
Quantisation 8	0.53	0.45	0.49	0.38	0.34	0.29	0.31	0.22	0.16	**0.01**
Quantisation 64	0.56	0.54	0.49	0.39	0.40	0.27	0.31	0.22	0.23	**0.07**

Table 8.11: Average test set Type 0 error rates and MAEs for all classifiers using 4 GLCM features with quantisation levels 8 and 64, for the tea images; results are averaged over 10 runs.

	Type 0 error rates							
Class	Quantisation at level 8				Quantisation at level 64			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.427	0	0.000	0.083	0.400	0	0.000	0.110
2	0.687	0	0.353	0.487	0.933	0	0.600	0.633
3	0.873	0	0.680	0.660	0.920	0	0.573	0.757
4	0.967	0	0.413	0.713	0.993	0	0.440	0.630
5	1.000	0	0.887	0.830	1.000	0	0.833	0.877
6	0.993	0	0.920	0.817	1.000	0	0.940	0.800
7	0.993	0	0.787	0.793	1.000	0	0.673	0.727
8	0.380	0	0.433	0.677	0.560	0	0.560	0.707
Overall	0.783	0	0.559	0.632	0.851	0	0.578	0.655
	MAEs							
Class	Quantisation at level 8				Quantisation at level 64			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	1.493	0	0.000	0.213	1.147	0	0.000	0.207
2	1.867	0	0.967	1.360	4.073	0	1.507	1.763
3	2.667	0	1.293	1.437	2.467	0	1.047	1.557
4	2.860	0	0.747	1.303	3.087	0	0.853	1.203
5	2.833	0	1.813	1.513	3.047	0	1.640	1.643
6	2.727	0	1.693	1.467	2.827	0	1.607	1.403
7	2.853	0	1.653	1.600	2.833	0	1.607	1.567
8	2.200	0	1.233	1.527	3.460	0	1.720	2.013
Overall l	2.438	0	1.175	1.042	2.867	0	1.248	1.136

was for SAR sea ice imagery, so the choice of optimum quantisation level may depend on the application. Again using GLCM features generally give much poorer results than using the granulometric moments.

8.4 Wavelet-based Features for the Tea Images

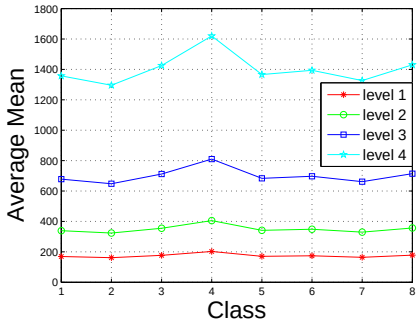
To compare with the performance of granulometric moments and GLCM features for texture image classification, wavelet-based features were also computed from the tea images. A 2-D DWT was applied to decompose the input images into four sub-bands, i.e. the approximation sub-band, and horizontal, vertical and diagonal sub-bands. In the second and higher decomposition levels only the approximation was decomposed into four sub-bands and so on. Among the many possible wavelet bases the Daubechies wavelet with 45 vanishing moments was used here as the mother wavelet, as it is a widely used wavelet basis (Borah et al. (2007), Chaplot et al. (2005), and Salari and Ling (1995)). A brief account of DWTs is given in Section 3.3.

The DWT at four different resolutions was applied on the foreground of the 256^2 grey scale tea images, and statistical features were computed, namely the mean and sd, maximum, energy and entropy from each of the four sub-bands. We computed the maximum feature as the maximum intensity of the sub-bands.

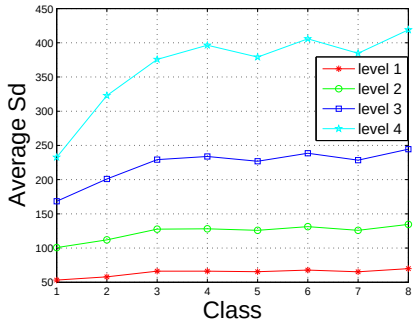
Since there are 50 sub-images in each of the 8 classes and 5 features for each of the 4 sub-bands, 20 features were computed at each level from each class. There are 4 levels of decomposition, so the feature matrix is of size $400 \times 20 \times 4$, where rows represent the class of the 50 sub-images, columns represent different features and the third dimension represents decomposition level. Average features were obtained by averaging over the 50 sub-images in each class, and are plotted against class in Figures 8.11, 8.12 and 8.13. The means of any sub-band for the first two levels of resolution are almost constant (near-zero) over the classes, but do change with class for the level 3 and 4 resolutions, especially for level 4.

The sd of the approximation sub-band increases with class for all four resolutions, especially for level 3 and 4, although the rate of increase is lower for the lower level of decompositions. All four are expected to be useful for classification. The sds of all the detail sub-bands show constant trends. The maxima of the detail sub-bands show a very similar pattern to that of the sds. However the maxima of the approximation sub-band varies with class, although they do not look useful as they do not show any clear trend.

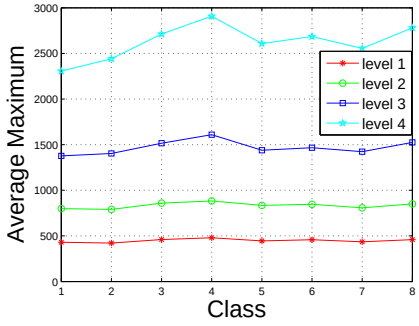
Entropy is a measure closely related to randomness. For the first two levels of decomposition the entropies are almost constant. However they vary at a higher level of decomposition (Figures 8.13 (a)–(d)). At level 4 decomposition, class 4 has the highest entropy of the approximation sub-band, whereas the entropy of the detail sub-bands is high for class 3.



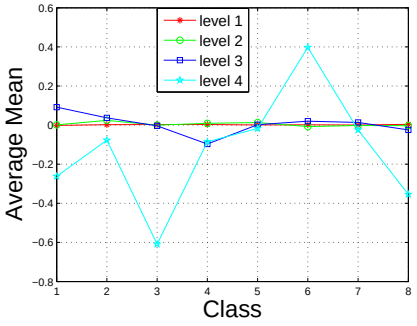
(a) Mean of Approximation sub-band



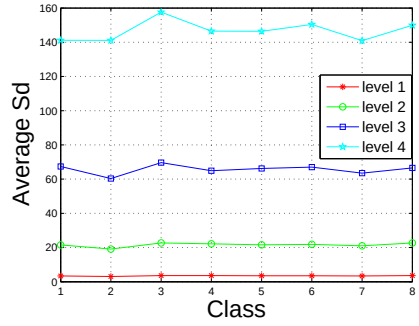
(b) Sd of Approximation sub-band



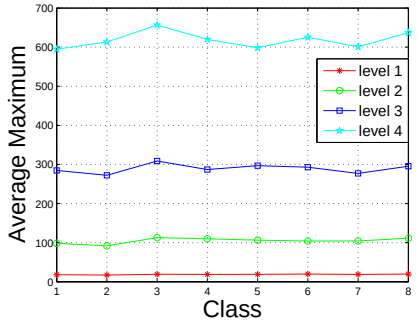
(c) Maximum of Approximation sub-band



(d) Mean of Horizontal sub-band



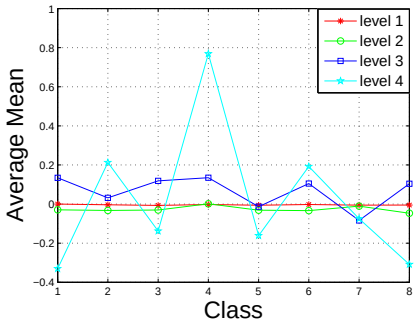
(e) Sd of Horizontal sub-band



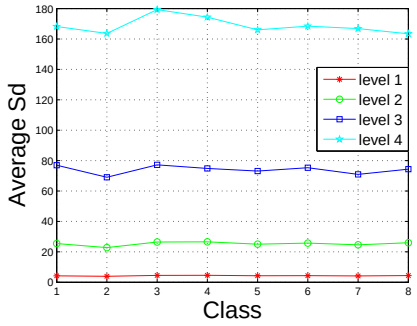
(f) Maximum of Horizontal sub-band

Figure 8.11: Average wavelet-based coefficients from the 256^2 grey scale tea images for the first 4 levels.

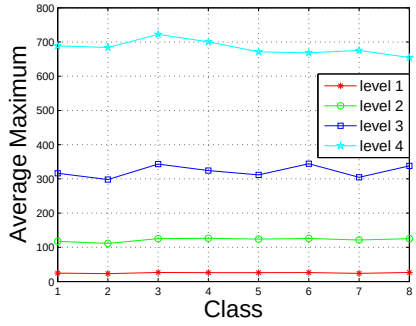
Energy is a measure closely related to information content in an image. We computed energy for all sub-bands. The amount of information is higher for the lower levels of decomposition of the approximation sub-band and is almost constant at around 100 for all classes at the first level and decreases with subsequent decomposition. It is clearly visible in Figure 8.13 (e) that class 4 contains the highest energy and class 8 the second highest for all four levels of decomposition. However, the energy of any detail sub-band increases with the level of decomposition. They are almost constant at 0 for level 1 but vary at further levels of decomposition.



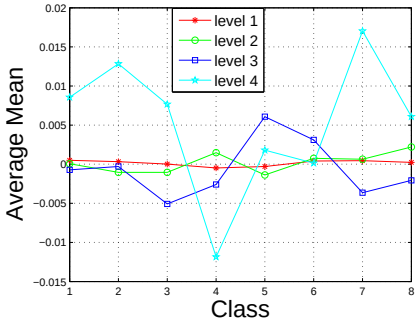
(a) Mean of Vertical sub-band



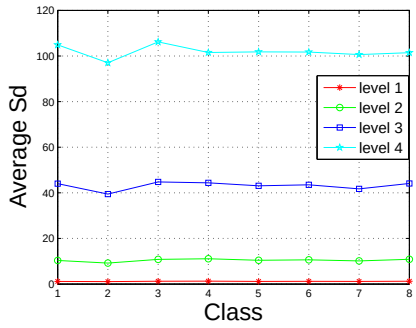
(b) Sd of Vertical sub-band



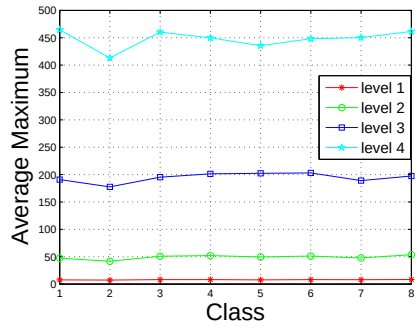
(c) Maximum of Vertical sub-band



(d) Mean of Diagonal sub-band



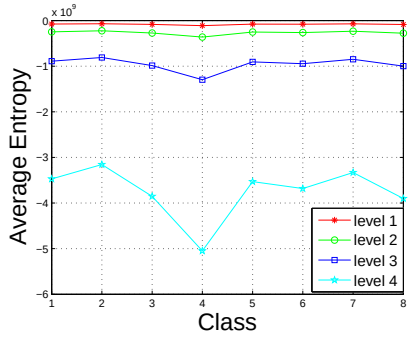
(e) Sd of Diagonal sub-band



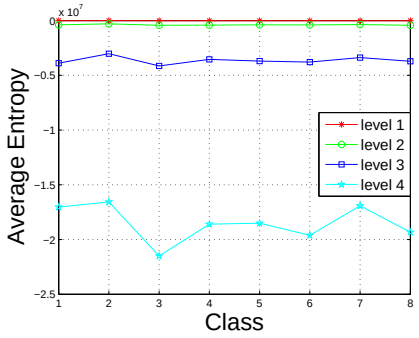
(f) Maximum of Diagonal sub-band

Figure 8.12: Average wavelet-based coefficients from the 256^2 grey scale tea images for the first 4 levels.

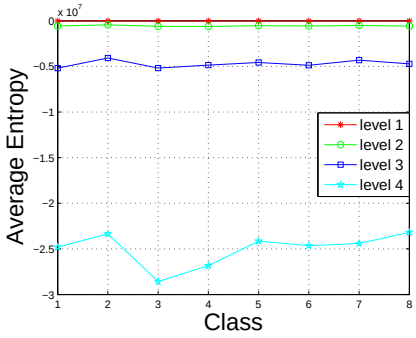
The means, sds, maximums, entropies and energies of all sub-bands at levels 3 and 4 differ most with class although they do not show any clear trend with class.



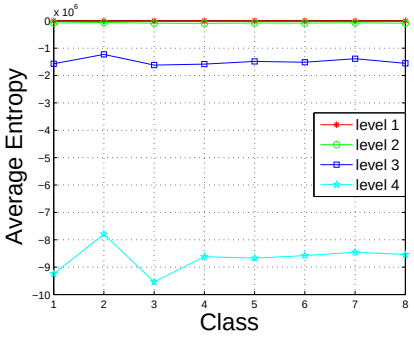
(a) Entropy of Approximation sub-band



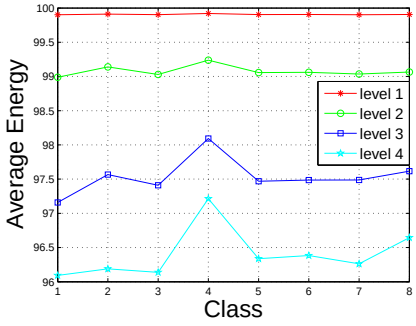
(b) Entropy of Horizontal sub-band



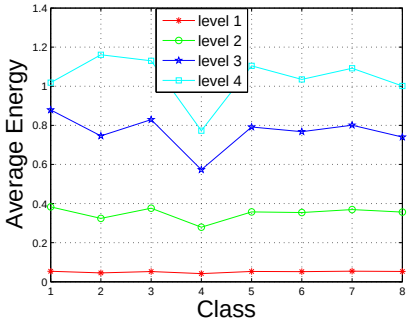
(c) Entropy of Vertical sub-band



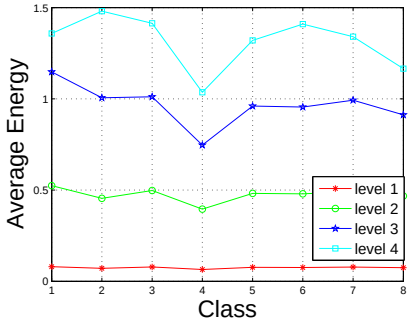
(d) Entropy of Diagonal sub-band



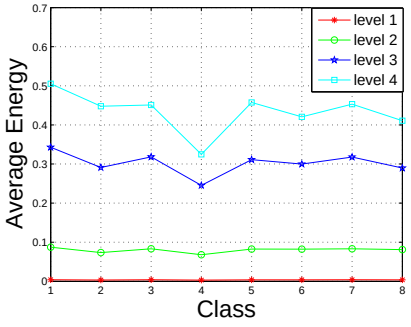
(e) Energy of Approximation sub-band



(f) Energy of Horizontal sub-band



(g) Energy of Vertical sub-band

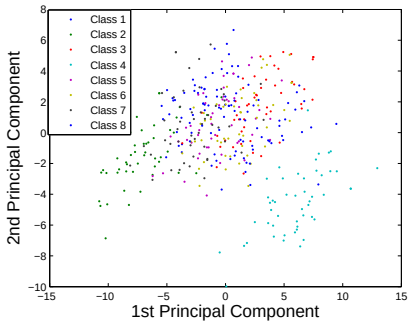


(h) Energy of Diagonal sub-band

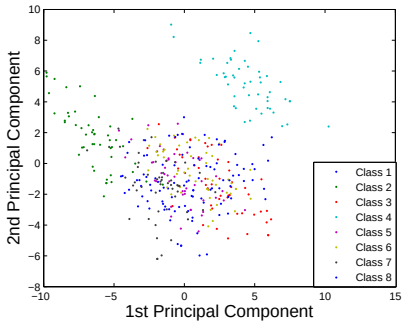
Figure 8.13: Average wavelet-based coefficients from the 256^2 grey scale tea images for the first 4 levels.

8.4.1 PCA on wavelet features for the tea images

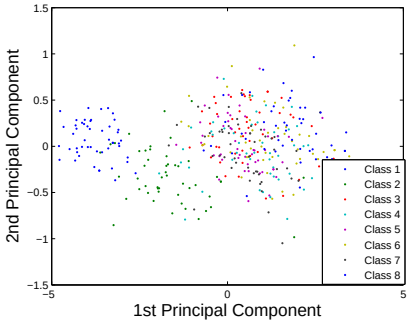
There are 20 features (5 features from 4 sub-bands) extracted from each level of decomposition, giving a total of 80. Different combinations of the features were considered for classification. Figures 8.11, 8.12 and 8.13 suggest that the means, sds and maximums of all sub-bands at level 1 (12 features), entropies of all 4 sub-bands at level 1 and 2 (8 features) and energies of all sub-bands at levels 1 (4 features) are almost constant over class. Hence these were not considered at this stage, giving $80 - 12 - 8 - 4 = 56$ features for classification. The means, sds, maximums, entropies and energies of all sub-bands at level 1 and 2 were also excluded as the features at level 1 are constant and only slightly vary with class for level 2, leaving $80 - 5 * 4 * 2 = 40$ features for classification. It is also obvious from the figures that only the sds of the approximation sub-band increase with class, so we also used these 4 features alone for classification as a third feature set.



(a) PCs using 56 features



(b) PCs using 40 features



(c) PCs using sds of approximation sub-band

Figure 8.14: Scatter plots of first two PCs using different sets of wavelet features for the tea images.

PCA was first applied to the 56 and 40 selected features, and then 4 features, to observe the clustering ability of the feature sets. Figure 8.14 shows that PCA using 56 features distinguishes classes 2 and 4, and using 40 features

distinguishes only class 4 but the rest overlap, whereas using 4 features from the approximation sub-band differentiates the first two classes while leaving the rest indistinguishable.

8.4.2 Using different classifiers

The different sets of features were used in all the classifiers. Again 70% of the features, randomly sampled, were used to train each classifier and each was tested on the other 30% of the features. Parameter settings were chosen for SVM and FF-NNET using a single training set. The process was repeated 10 times and average error rates and average MAEs were computed.

Table 8.12: Training set error rates for different combinations of cost and γ for the radial basis kernel and for a linear kernel using 56, 40 and 4 wavelet-based features from the tea images.

	Radial basis kernel										
	Cost [56 wavelet-based features]										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.43	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45
0.5	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73
0.9	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74
	Cost [40 wavelet-based features]										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48
0.5	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
0.9	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79
	Cost [4 wavelet-based features]										
	1	10	20	30	40	50	60	70	80	90	100
0.1	0.11	0	0	0	0	0	0	0	0	0	0
0.5	0.02	0	0	0	0	0	0	0	0	0	0
0.9	0.03	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	Linear kernel										
	1	10	20	30	40	50	60	70	80	90	100
56 features	0.86	0.86	0.88	0.87	0.87	0.83	0.88	0.88	0.86	0.84	**0.84**
40 features	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	**0.30**
4 features	0	0	0	0	0	0	0	0	0	0	**0**

Table 8.12 shows that neither the linear kernel nor the radial basis kernel attained 100% correct classification on the training set for any cost between 1 and 100 using 56 and 40 wavelet-based features in the SVM, but using only 4 features 100% correct results can be obtained for a linear kernel with any cost

between 1 and 100, and with the radial basis kernel with a cost of 10 or more and γ between 0.1 and 0.5 inclusive. A linear kernel with a cost of 100 was used for all sets of features. For all 3 feature sets, a FF-NNET with decay as 10^{-4} and rang as $1/\max(|\boldsymbol{x}|)$ was best, using normalised features. With these parameter values the training set error rate for different numbers of hidden units was computed and it was found that 10 units were appropriate for all sets of features (Table 8.13).

Table 8.13: Training set error rates for different numbers of hidden units in a FF-NNET, using all 56, 40 and 4 wavelet-based features from the tea images.

	Number of units									
	1	2	3	4	5	6	7	8	9	10
56 features	0.28	0.25	0.03	0	0	0	0	0	0	**0**
40 features	0.51	0.31	0.49	0.05	0	0	0	0	0	**0**
4 features	0.52	0.36	0.33	0.26	0.25	0.26	0.18	0.13	0.13	**0.02**

Type 0 errors and MAEs for all classifiers are shown in Table 8.14 using 56, 40 and 4 features. REG has the same error rate of 85.7% for both 56 and 40 features and the error rates from LDA are similar for both sets of features, but SVM and FF-NNET work better with 56 features than 40 features. However SVM gives 100% correct classification using 4 features and REG, LDA and FF-NNET have 63.2%, 40.1% and 50.2% error rates respectively, so using only the sd of the approximation sub-band from each level of decomposition provides improved classification results for REG and SVM. The MAEs indicate the same. SVM is clearly the best classifier overall.

8.5 Conclusion

For the pyramid images, the regression approach, LDA and FF-NNET are equally effective for classifying these images, though LDA and FF-NNET produced lower error rates than the regression approach for the ellipse images.

GLCM features produced slightly lower classification error rates for the regression approach but slightly higher rates for LDA and FF-NNET for the pyramid images compared to the granulometric moments (Table 5.10). For both feature sets, SVM with a linear kernel and a cost of 20 or more produced 100% correct classification and is superior.

Again the linear kernel was the best choice of kernel for SVM with GLCM features from the corrosion images, with a small test set error rate (2.7% for

quantisation level 8, 1.7% for quantisation 64 (Table 8.6) and 1.3% with no quantisation (Table 8.7)). Error rates for the other classifiers are high. Use of additional GLCM features does not have benefits, rather it produced worse results in some cases. For the tea images, error rates for LDA and FF-NNET are quite comparable, while the regression approach has a higher error rate and SVM with a linear kernel classifies perfectly.

LDA works better for the wavelet-based features than the GLCM features for both quantisations, but FF-NNET produced lower error rates for 4 wavelet-based features than the GLCM feature for both quantisations. However SVM produced 11.8% error with 56 wavelet-based features. The regression approach worked better with 4 wavelet-based features than the GLCM features but worse with 56 wavelet-based features. Nevertheless error rates for all classifiers are much higher with the wavelet-based features compared to the granulometric moments. Therefore, for both sets of real images granulometric moments provide very useful features compared to the GLCM and wavelet-based features for shape-based texture image classification.

In the next chapter hyperspectral images of six different types of Chinese teas are used with several different band selection techniques and feature sets in another classification problem.

Table 8.14: Average test set Type 0 error rates and MAEs for all classifiers using 56, 40 and 4 wavelet-based features from the tea images; results are averaged over 10 runs.

	56 features							
Class	Type 0 errors				MAEs			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.427	0.000	0.000	0.100	0.427	0.000	0.000	0.203
2	0.847	0.060	0.073	0.127	0.847	0.060	0.260	0.387
3	0.953	0.127	0.387	0.530	2.220	0.220	0.967	1.473
4	1.000	0.027	0.047	0.133	4.000	0.027	0.107	0.380
5	1.000	0.127	0.547	0.647	3.553	0.200	0.933	1.250
6	1.000	0.227	0.487	0.715	3.787	0.240	0.773	1.318
7	1.000	0.173	0.420	0.410	5.327	0.173	1.073	1.103
8	0.627	0.200	0.453	0.513	3.953	0.327	1.393	1.503
Overall	0.857	0.118	0.302	0.340	3.014	0.155	0.688	0.952
	40 features							
Class	Type 0 errors				MAEs			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.427	0.053	0.035	0.117	0.427	0.063	0.085	0.303
2	0.847	0.140	0.085	0.403	0.847	0.187	0.275	1.317
3	0.953	0.220	0.410	0.573	2.220	0.367	1.175	1.677
4	1.000	0.057	0.045	0.227	4.000	0.077	0.075	1.620
5	1.000	0.440	0.620	0.773	3.553	0.653	1.140	1.593
6	1.000	0.487	0.640	0.763	3.787	0.670	1.065	1.520
7	1.000	0.407	0.445	0.707	5.327	0.583	1.155	2.030
8	0.627	0.310	0.465	0.613	3.953	0.560	1.305	1.920
Overall	0.857	0.264	0.343	0.522	3.014	0.395	0.784	1.373
	4 features							
Class	Type 0 errors				MAEs			
	REG	SVM	LDA	FF-NNET	REG	SVM	LDA	FF-NNET
1	0.000	0	0.000	0.013	0.000	0	0.000	0.030
2	0.307	0	0.533	0.213	0.373	0	0.107	0.590
3	0.540	0	0.267	0.567	1.347	0	0.553	1.277
4	0.960	0	0.533	0.567	2.820	0	0.940	1.170
5	1.000	0	0.687	0.667	2.333	0	1.127	1.127
6	1.000	0	0.793	0.827	2.480	0	1.327	1.467
7	0.927	0	0.573	0.663	2.913	0	1.213	1.437
8	0.320	0	0.300	0.503	1.453	0	0.780	1.233
Overall	0.632	0	0.401	0.502	1.715	0	0.756	1.041

Chapter 9

Hyperspectral Image Classification

This chapter describes hyperspectral images and reviews some applications of hyperspectral imaging. It applies some of the existing methods for selecting appropriate spectral bands to hyperspectral images of six different varieties of Chinese teas, and explores the benefits of using hyperspectral images over colour and grey scale images for classifying these datasets. It also investigates the comparative performance of shape-based texture features, i.e. pattern spectrum (PS) moments, and several intensity-based texture features, i.e. grey level co-occurrence matrix (GLCM) features, wavelet-based features and wavelet-based GLCM features.

9.1 Hyperspectral Imaging

A digital image is a 2D finite array of pixels, each with a pixel intensity. For a binary image, pixel intensities are either 0 or 1, whilst for a grey scale image, intensities usually range from 0 to 255. Only spatial information is available in these images but not spectral information. A colour image consists of three layers of grey scale images corresponding to three wavelength regions corresponding broadly to red, green and blue. In the electromagnetic spectrum (EMS) the RGB region is known as the *visible region* ranging from wavelength 400 nanometres (nm) to 700nm. A multispectral image consists of up to tens of layers of grey scale images, each corresponding to a specific spectral band in the EMS, whereas a hyperspectral image contains hundreds to thousands of grey scale images covering a wider range of wavelengths, e.g. 2nm to 2500nm (Geladi et al. (2007)).

Image generation

A hyperspectral image is a 3D dataset generated by photographing the same image through a series of wavelength bandpass filters. There are two main approaches to collecting hyperspectral images, namely *push-broom imaging* and *tunable filtering imaging*. Both approaches require a 2D optical sensor array to separate wavelengths from each other. A push-broom imaging technique consists of a suitable objective lens matched to the spatial and spectral requirements of the application, an imaging spectrograph and a 2D detector to simultaneously collect the spectral versus spatial information. Three basic camera configurations, point scan, line scan and focal plane scan, are used based on the type of spatial information required. For x, y and λ data, (where x and y are the coordinates of a spectral band and λ is the central wavelength value of that band), push-broom collects x and λ by scanning y, whereas tunable filtering collects x, y by scanning λ. A detailed description is given in Geladi et al. (2007).

Line scanning imaging was used to capture the Chinese tea images used in this chapter. This records sequential y values, building up a hyperspectral image with two spatial dimensions considering all pixels and the central wavelength values λ of each band in an image. The scanning can be done either by camera movement (e.g. aircraft) or by movement of the scene to be imaged (e.g. conveyor belt, Balas et al. (2003)).

9.2 Overview of Band Selection Methods

Hyperspectral imaging acquires hundreds of spectral bands within a wide range of the EMS region, hence provides rich spectral information which is very often advantageous for classification. However, the high dimensionality of hyperspectral data makes the classification task more challenging and computationally expensive. Moreover, the contiguous bands in hyperspectral data are very often captured at a narrow spatial resolution (few nm) (Gilchrist and Hyvarinen (2006)), consequently, some adjacent bands represent similar spectral and spatial information. Use of such redundant information not only increases computational expense but may also decrease classification efficiency. Therefore, it is essential to select appropriate spectral bands of the hyperspectral data before classification. Different authors have adopted different strategies for selecting bands, some of which are discussed below.

9.2.1 PCA reduction

PCA is the most widely used dimensionality reduction technique in this domain. PCA maximises the variance of the input data in the new uncorrelated coordinate system. Many researchers have applied PCA to select appropriate spectral bands for classification purposes. For example, Zhao et al. (2009) applied PCA to extract 3 dominant bands from 150 bands of five types of Chinese green tea to discriminate tea categories. Benediktsson et al. (2005) proposed using the number of PC images that account for more than 90% of the variation in the data, and used the first 2 PC images extracted from the first 40 out of 80 spectral bands (the last 40 bands were excluded due to heavy noise) of hyperspectral images of urban areas of Pavia, Italy, whilst Tan and Du (2010) used the first 7 PCs from the same dataset using all spectral bands. Zhang et al. (2010) used airborne hyperspectral images of Shanghai, to classify 8 different kinds of vegetation area using the first 3 PC images extracted from the original data.

9.2.2 Using all bands

Although hyperspectral images consist of hundreds of spectral bands, some authors have suggested using all available spectral bands. For example Jia et al. (2010a) used a wavelet approach on all 220 bands of Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) images of Indiana's Indian Pines (scenes taken over northwest Indiana) for classification. Prasad et al. (2010) used all bands of a 2151-band handheld ASD dataset and a 128-band airborne SpecTIR dataset, to separate seven classes of corn crops. Eight regions of different bodies of water, ranging from clear water to an evaporation pond, and 2 regions of dense and sparse vegetation were classified using 224 spectral bands of AVIRIS data in Subramanian et al. (1997). Huang and Zhang (2008) used all 126 spectral bands of hyperspectral images taken over the West Lafayette campus of Purdue University to extract roads within images of the campus.

All 220 spectral bands of AVIRIS images of Indiana's Indian Pines were used to classify 10 out of 16 terrain classes based on a sparse conditional random field model in Zhong and Wang (2008), whereas Liguo et al. (2009) and Kuo et al. (2010) used all bands of the same dataset to classify all 16 terrain classes. All spectral bands (210) of the Washington DC Mall dataset were used in Yin et al. (2010) and Yan et al. (2010). Kim et al. (2008) used all spectral bands of hyperspectral images taken over Okavango Delta, Botswana. Rellier et al. (2004) used all 224 spectral bands of AVIRIS images of Moffett Field, California. Moshou

et al. (2005) used all 4 available spectral bands of hyperspectral images of wheat plants to detect plant disease, using data fusion. Different anomaly detectors were developed and applied to the 224 bands of AVIRIS scenes and 15-panel hyperspectral digital imagery collection experiment (HYDICE) scenes in Chang (2002). Jensen et al. (2008) used all spectral bands of the Rosis dataset (81 bands taken at Fontainebleau forest, Paris) and an airborne sensor over Kennedy Space Center at Cape Canaveral, Florida, consisting of 224 bands, to verify their proposed regression model. Guo et al. (2008) used all spectral bands of AVIRIS 92AV3C hyperspectral images. Haq et al. (2010) used all 210 spectral bands of the Washington DC Mall dataset to validate their model. Chang (2002) evaluated his method on all the bands of AVIRIS images of the Lunar Crater Volcanic Field of North Nye County.

9.2.3 Band selection by inspection or prior knowledge

However many researchers identified some damaged and/or less useful bands by visual inspection and discarded them. For example, among 224 spectral bands 4 noisy bands were discarded from AVIRIS images from Purdue University, in Hsu and Yang (2007). Rabe et al. (2010) used 114 spectral bands after radiometric correction and eliminating noisy bands from the HyMap sensor acquired over the city of Berlin. Huang and Zhang (2008) removed 18 water-absorption bands from the Washington DC Mall data set (210 bands) and applied a mean-shift to obtain an object-oriented representation of hyperspectral data. Chen and Tran (1994) selected 50 bands (41-90) out of 224 spectral bands of AVIRIS images as they contained more terrain features. Jensen et al. (2008) used spectral bands 1-71 out of 80 bands of an image of Pavia, of which the last 8 captured thermal infrared. They also selected some bands manually from another hyperspectral image taken over Okavango Delta, Botswana by removing noisy bands.

Bazi and Melgani (2006), Demir and Ertürk (2007, 2011), Haq et al. (2010), Camps-Valls et al. (2007), Jia et al. (2010a), and Ratle et al. (2010) discarded 20 out of 220 spectral bands of the Indiana Indian Pines data, as they were affected by an atmospheric problem, and applied their methodology on the selected bands for classification. Serpico et al. (2004) and Serpico et al. (2007) discarded 18 bands of the same dataset and used 202 for classification. Ratle et al. (2010) also used 176 bands out of 224 bands of the Kennedy Space Center data after omitting the water absorption bands. Du and Chang (2001) used 196 spectral bands after removing some noisy bands of HYDICE images of scenes. Chang (2002) evaluated the proposed methodology on HYDICE images after removing some noisy bands.

Nidamanuri and Zbell (2011) eliminated 4 bands due to noise and the presence of faulty detectors from a airborne hyperspectral images consisting of 128 spectral bands.

9.2.4 Other band selection approaches

Apart from these simplistic techniques of band selection, a few authors have suggested some more sophisticated approaches. For example, Zhao et al. (2011) selected appropriate bands by computing the correlation matrix of all bands and selecting the less correlated bands. Zhu et al. (2010) proposed a memetic ant colony optimisation (MACO) band selection approach for hyperspectral image classification, based on the foraging behaviour of ants searching for the shortest path between a food source and their nest. Bajcsy and Groves (2004) proposed combining unsupervised and supervised methods for selecting appropriate hyperspectral bands for classification. Seven unsupervised methods including information entropy, first and second spectral derivatives, spatial contrast, spectral ratio, correlation and PCA were applied separately to prepare seven rank-ordered lists. Three supervised methods, namely regression, an instance-based method (using an inverse Euclidean distance weighting of the k-nearest neighbours) and a regression tree were employed using those rank-ordered lists as well as random and incremental ranking of all spectral bands. Lower error rates were obtained using the unsupervised rank-ordered lists of bands. Chen and Zhang (2011) proposed several dimension-reduction techniques using prior knowledge of pairwise constraints (pairs of samples belonging to the same class are referred to as *must-link constraints* and ones belonging to different classes are called *cannot-link constraints*).

A mutual information (MI) based band selection approach is proposed in Ren et al. (2011). MI between each pair of adjacent bands is computed in terms of Shannon entropy. Zero MI indicates complete independence of the corresponding band images, and the smaller the MI the greater the chance of bands being in different clusters. All MIs are ordered in ascending order and the corresponding band labels are recorded. The first m elements of the ranked series and their band labels are used for grouping all spectral bands. This splits up all available bands of the hypercube from left to right into groups. One representative band with the smallest summed mean squared difference or summed mean absolute difference from each of the other bands in the group was selected from each group. So there are $m - 1$ bands chosen for feature extraction.

9.3 Different Approaches for Classification

Hyperspectral imaging is used extensively in remote sensing image classification but has rarely been reported so far for discriminating tea varieties, except in Zhao et al. (2009). They used hyperspectral images of five grades of roasted Chinese green tea leaves where the grades were determined using average scores from 3 skilful tea tasters. Each taster scored on appearance, taste and aroma. The hyperspectral image data was gathered through a system consisting of 3 modules, namely the sensor module, the lighting source module and the conveyer module. The sensor module was an ImSpector V10E HSI camera with a spectral range of 408-1117nm and spectral resolution of 2.8nm. The lighting source module had two 150W quartz-halogen DC stabilised fiber-optic illuminators and the conveyer module consisted of a translation stage and a controller.

Corrected images were obtained by calibrating the original images with a white and a dark reference image as $R = (I - B)/(W - B)$, where B is the dark image obtained by closing completely the aperture, i.e. using approximately 0% reflectance, W is the white image obtained using approximately 99% reflectance, and I is the original image. The spectral reference R values range from 0 to 1. The spectral profiles among the five grades were very similar in the spectral region below 700nm, but were distinct in the spectral region 700-850nm.

PCA was applied to select 3 dominant wavebands from the spectral range 700-850nm. The first PC explained 79.1% of the variation in the original images. Images at spectral bands 762, 793 and 838nm were found to be the dominant bands. Sample images of size 300^2 were extracted from the original images of size 1280×600. Four texture descriptors, namely mean, sd, energy and entropy, were computed from each optimum sub-band image and used in a SVM. Among 700 sample images (140 for each grade), 500 images (100 for each grade) were used to train the classifier and the remaining 200 sample images were used to test it. They obtained 98% and 95% correct classification rates for the training and test set respectively. A back-propagation neural network (BP-NNET) also produced 98% accuracy for the training set, but 90% for the test set. Classification accuracy using LDA was 88% and 85% for the training and test sets respectively.

Morphological techniques were applied to hyperspectral images in Benediktsson et al. (2005). They proposed using PCs that account for around 90% of the variation in the data. The first two PC images (accounting for more than 90% variation) from the first 40 spectral bands of 80, of the urban areas of Pavia, Italy, were used. The last 40 bands were excluded due to heavy noise. They derived an *opening profile* which constitutes opening granulometry (as we use) but using

opening by reconstruction, and a *closing profile* which is anti-granulometry (also known as closing granulometry) from closing by reconstruction. The opening and closing profiles were constructed from both PCs and were used to develop an extended morphological profile. A NNET using the morphological profile based on the first 2 PCs produced more than 99% classification accuracy.

Mathematical morphology was also used for detecting different ground cover in agricultural and urban classification by Plaza et al. (2005). An extended differential morphological profile (EDMP) and a scale-orientation morphological profile (SOMP) were constructed by simultaneously considering the spatial and spectral information of the image data. Morphological features were extracted from real HSI AVIRIS images taken over Salinas Valley, CA, consisting of 192 spectral bands, and from 40 spectral bands of DAIS 7915 image data of Pavia, Italy. SOMP with a NNET classifier achieved 94% and 96% classification accuracy for the AVIRIS and DAIS datasets respectively.

Jia et al. (2010b) proposed de-noising each hyperspectral band first and then computing wavelet coefficients. They applied wavelet shrinkage on all 220 bands of AVIRIS images of Indiana's Indian Pines and then applied a discrete wavelet transform on the spectral signature to obtain a number of wavelet features. Then they used an affinity propagation algorithm to select optimum features. Using a k-NN classifier, 83% classification accuracy was obtained.

Prasad et al. (2010) proposed a confusion-driven adaptation technique consisting of estimating confusion matrices from the training sample and iteratively finding features that best separate the most confused classes. In the classification phase, a stepwise LDA (SLDA) and a quadratic maximum likelihood (ML) classifier were combined, and improved classification (78%) was obtained from multiple-classifiers (SLDA+ML) compared to a single classifier for classifying seven classes of corn crops. The method was tested on a 2151-band handheld ASD dataset and 128-band airborne SpecTIR dataset.

Rabe et al. (2010) proposed simplifying an SVM to reduce the number of support vectors, hence reducing computation time. Firstly, an SVM was trained on the original training samples and then used to classify them. Afterwards, the class membership was replaced by the SVM class decision and a second SVM was applied on the modified training data. Much improved results were obtained for classifying HSI data consisting of 114 spectral bands from the HyMap sensor used over Berlin, using their simplified SVM compared to the regular SVM.

Zhang et al. (2010) classified airborne hyperspectral images of Shanghai, to classify 8 different kinds of vegetation area. The first 3 PCs were plugged into the

ML classifier. Then *easy mixed* classes were identified by investigating the classification results. For the easy mixed classes, a *projection pursuit* (PP) algorithm was applied to find the optimal projection directions, projecting the full dataset on the calculated directions and constructing the PP feature subspace. Then they combined the PP feature subspace with PCs and applied the ML classifier again. The combined features produced higher classification accuracy of 67%, whereas PCs alone produced 60% accuracy.

Bazi and Melgani (2006) applied a SVM classifier directly on hyperspectral AVIRIS images of land cover classes consisting of 220 bands, of which 20 were discarded as they were affected by an atmospheric problem. They used 1800 training samples and 4588 test samples, and 87.66% overall classification accuracy was obtained. Two other modified versions of SVM were also used to classify the same data with improved accuracy.

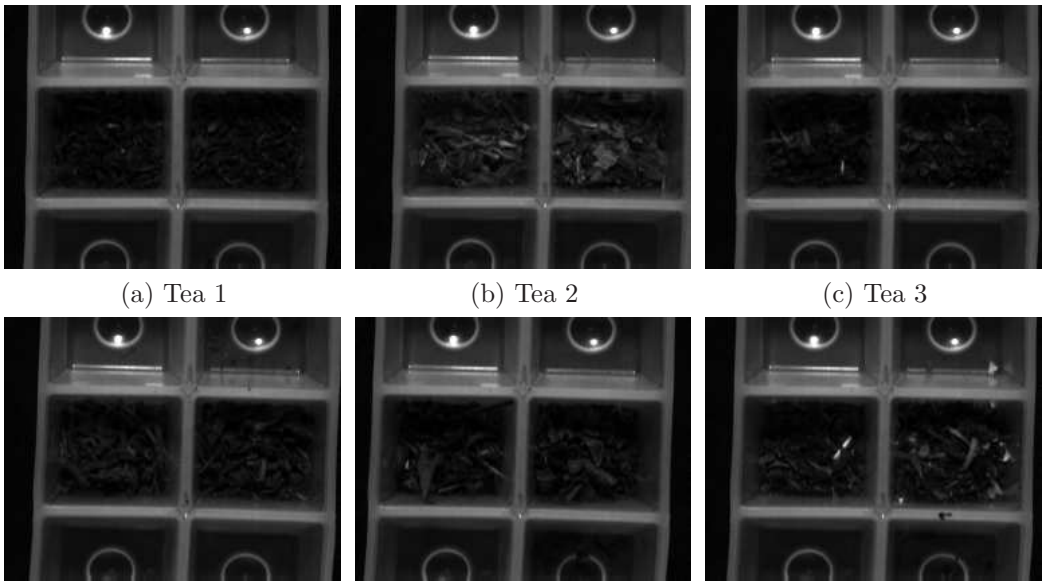
A FF-NNET using eigenvalues from the region of interest from 224 spectral bands of AVIRIS data was proposed in Subramanian et al. (1997) and 100% accurate classification was attained. Zhao et al. (2011) proposed partitioning the hyperspectral data cube into several nearly uncorrelated sub-sets as the varying data quality and discrimination ability across bands may affect the accuracy of classification. An eigenvalue-based approach was proposed to evaluate the confidence of each subset. A fuzzy c-mean clustering method and a k-NN classifier produced nearly the same accuracy for classifying real HYDICE scenes. Huang and Zhang (2008) used a SVM for classification of two sets of real hyperspectral image data. One consisted of HYDICE images taken over the Mall, Washington DC, using 210 bands of which 18 water-absorption bands were deleted, to classify roads, grass, water, shadow, trees, trails and roofs. The second dataset from a flight over Purdue University contained 126 spectral bands used to extract roads within the images.

9.4 Image Description

There are hundreds of different varieties of tea from the tea bush Camellia Sinensis, but almost all of them fall into three main categories. These are black teas, which are fully oxidised before drying by a careful withering process; green teas, which are dried quickly while unoxidised; and Oolongs, which are somewhere in the middle and combine the best qualities of both (http://oolong.co.uk/tea.html). Six varieties of Chinese tea, namely China Black, Lung Ching, Tikuanyin, Yunnan, Oolong and Jasmine are studied in this chapter.

Black tea leaves come from the same tea plant, Camellia Sinensis, as does all tea. The first step after plucking the leaves is to let them wither. Then there are three additional processing steps that the leaves are subjected to before becoming black tea. They are rolled, allowed to fully oxidize (ferment), and lastly they are dried. After rolling, they are sifted to separate the different leaf particle sizes (http://folk.uio.no/gisle/cache/blacktea.html). Lung Ching, also known as Dragon Well tea, is one of the most popular Chinese green teas, mainly produced in the Zhejiang province of China. It has a green colour, mellow taste, fragrant aroma, and graceful shape. Lung Ching tea is characterised by its unusual shape and characteristics of the dry leaves, i.e. long, flat, and very lightweight. Oolong teas are produced in Fujian province in China. Tikuanyin tea, also originating in Fujian province, is a variety of Chinese oolong tea and has been very popular for centuries. The tea leaves are dark green and rolled into tight balls. Tikuanyin tea is also known as Ti Kwan Yin, Anxi Tie Guan Yin, Iron Goddess of Mercy etc. (http://www.chinese-tea-culture.com/tieguanyin-tea.html). Yunnan is one of the best Chinese black teas, mainly produced in Yunnan province. Yunnan tea is highly regarded for its unique malt and peppery taste. Jasmine tea is a famous tea made from Chinese green tea leaves that are scented with jasmine flowers (http://www.wisegeek.com/what-is-yunnan-tea.html).

There is no ordering that can be established visually based on the size of the tea leaves among the different tea types. Hence, we arbitrarily refer to China Black, Lung Ching, Tikuanyin, Yunnan, Oolong and Jasmine tea as Tea 1, Tea 2, Tea 3, Tea 4, Tea 5 and Tea 6 respectively. Hyperspectral images of each tea type were acquired in Kelman et al. (2011) using an Andor Luca EMCCD camera with Specim V8E spectrograph. Each tea type was placed into a compartment of an ice cube tray and the tray was placed on a Zolix KSA 11-200S4N motorised stage, illuminated with an Armley 150W Halogen lamp and imaged simultaneously to generate a hypercube consisting of 196 scan lines, with 251 pixels in each line and 250 spectral bands in the 282nm to 865nm region of the EMS. Each hypercube was calibrated with a white and dark reference image such that the pixel intensities range from 0 to 1. A dark reference was recorded by putting the lens cap on the camera and a white reference was obtained by placing a white tile under the camera. Occasionally there may be values above 1, due to specular reflection from the material being imaged (tea leaves here). The spectraSENS software used to capture the images has a function to allow corrections. Figure 9.1 shows the raw hypercube images, one from each of the six hypercubes taken at spectral band 630nm.

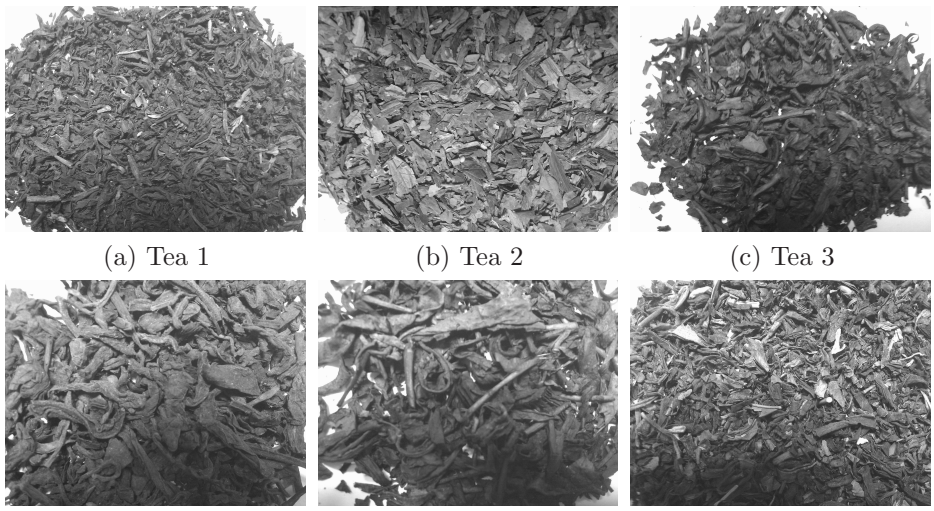


(a) Tea 1 (b) Tea 2 (c) Tea 3

(d) Tea 4 (e) Tea 5 (f) Tea 6

Figure 9.1: Hyperspectral images of 6 types of Chinese tea at wavelength 630nm.

Corresponding RGB images of each tea type were captured by uniformly placing the same tea sample on a white sheet of paper and imaging it using a CCD camera and saving in RGB format. Although the tea types are visually indistinct with respect to the hypercube images, they are distinguishable from each other in terms of leaf sizes, shapes and colour (Figure 9.2). We are interested in investigating whether hyperspectral imaging provides useful information compared to RGB images for classifying these tea types.

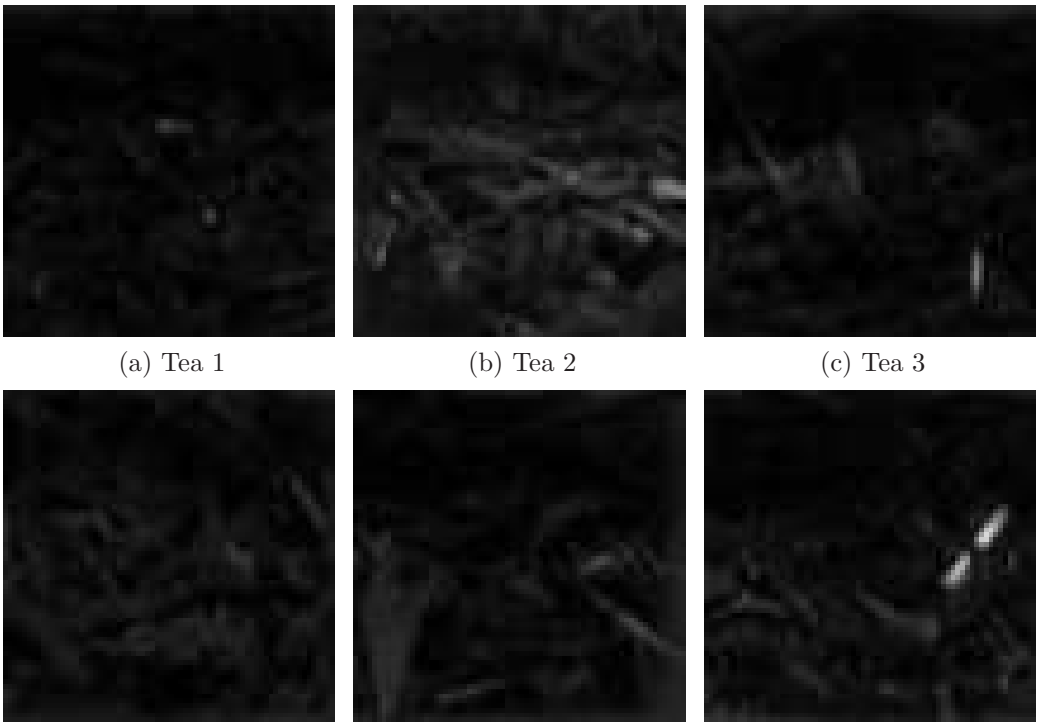


(a) Tea 1 (b) Tea 2 (c) Tea 3

(d) Tea 4 (e) Tea 5 (f) Tea 6

Figure 9.2: Colour images of size 1952×2592 of 6 types of Chinese tea. There are no bright colours in the images, so even printed in colour they look grey.

A considerable area of each slice of the hypercube contains the image of the ice cube tray rather than the tea sample. To obtain representative sample images, we selected the region of interest by manually cropping two images to the largest possible size of 70^2, one from each part of the ice cube tray, from each of the 250 spectral bands. One extracted image at spectral band 150 for each tea type is shown in Figure 9.3.



(a) Tea 1 (b) Tea 2 (c) Tea 3

(d) Tea 4 (e) Tea 5 (f) Tea 6

Figure 9.3: Sample images of size 70^2, one from each type of Chinese tea at wavelength 630 nanometers (spectral band 150).

9.5 Band Selection

Here we investigate on these tea images some of the simpler approaches used for band selection, namely using all of the bands, visual inspection to select some of the bands, and identifying the more informative bands using PCA, entropy and mutual information.

Approach 1: Firstly, we consider all 250 bands for all tea types.

Approach 2: Secondly, we select 150 bands out of the 250 spectral bands from each hypercube by discarding the first 100 bands, as 70 of them are very dark

and the next 30 are rather noisy.

Approach 3: Thirdly, PCA was applied on all 70^2 images extracted from all spectral bands for each type separately to select the first few PC images, then to select representative spectral bands corresponding to the strongest coefficients. We chose not to normalise the data as the pixel intensities all range between 0 to 1 (see Section 9.4).

PCA centring (subtracting the mean of each variable) is very common to avoid larger scale variables dominating PC1. Both Matlab and R centred the data before applying PCA and as a result, some of the centred data values become negative. Applying PCA on the centred data produces coefficients of each PC which may be a mixture of positive and negative. Multiplying the PC coefficients with the centred data produces PC scores, the representation of the original data in the principal component space. The PC scores have some negative values, which makes no sense when we deal with PC images, as images cannot have negative intensity. However, applying PCA on the original data does not always produce PC scores with all positive entries. PCA was also applied to the centred but un-scaled, non-centred but scaled and normalised (both centred and scaled) data to explore whether any options produced PC scores with positive entries. Nevertheless, none of the options was found to guarantee positive PC scores.

PCA can be applied to the covariance or correlation matrix of the data. It was found that applying PCA on the covariance matrix yields the same coefficients, with arbitrary sign, as PCA on the centred but unscaled data and the PC score is calculated by multiplying the coefficients with the centred but unscaled data. Similarly, applying PCA on the correlation matrix is the same as applying PCA on the normalised data and the PC score can be found by multiplying the PC coefficients with the normalised data. But none of the options generates PC scores with all positive entries.

The PC score is considered as the PC image, so its entries must be non-negative. To deal with the negative intensities, the PC score images are shifted along so their minimum is zero. Each PC image is checked for negative intensities and adjusted by adding the most negative intensity, e.g. add on 10 if the smallest value is -10. Clipping the negative intensities at zero may be an alternative, but this will affect the variation of the PC image intensities, hence limiting the information.

The first 10 PCs express around 85% of the variation among any of the tea types. The cumulative proportions of variation explained by each PC for all tea

types are shown in Table 9.1.

Table 9.1: Cumulative proportion of variation explained by each of the first 10 PCs from each tea type.

PCs	Tea 1	Tea 2	Tea 3	Tea 4	Tea 5	Tea 6
PC 1	0.70	0.60	0.72	0.73	0.71	0.66
PC 2	0.76	0.77	0.76	0.75	0.76	0.76
PC 3	0.78	0.79	0.78	0.77	0.78	0.78
PC 4	0.79	0.80	0.79	0.79	0.79	0.79
PC 5	0.80	0.81	0.80	0.80	0.80	0.80
PC 6	0.81	0.82	0.82	0.81	0.81	0.81
PC 7	0.82	0.83	0.83	0.82	0.82	0.82
PC 8	0.83	0.84	0.83	0.83	0.83	0.83
PC 9	0.84	0.85	0.84	0.84	0.84	0.84
PC 10	0.85	0.86	0.85	0.85	0.85	0.85

Then we select 10 spectral bands containing the higher PC coefficients in PC1 and use them for feature extraction. The coefficients of PC1 were sorted, and the highest 10 coefficients and the corresponding bands for all tea types are recorded in Table 9.2. The 10 highest coefficients correspond to the last 10 bands (241-250) for Teas 1, 3, 4 and 5, and for Teas 2 and 6 they also come from the upper end of the spectral region.

Table 9.2: Highest 10 coefficients from the first PC with the corresponding band for each tea type.

	Tea 1		Tea 2		Tea 3		Tea 4		Tea 5		Tea 6	
	Band	Coef.	Band	Coef.	Band	Coef.	Band	Coef.	Band	Coef.	Band	Coef.
1	241	0.139	232	0.117	241	0.132	241	0.121	241	0.131	234	0.112
2	242	0.139	233	0.117	242	0.133	242	0.121	242	0.131	235	0.112
3	243	0.139	234	0.117	243	0.133	243	0.121	243	0.131	236	0.112
4	244	0.140	235	0.117	244	0.133	244	0.122	244	0.131	237	0.112
5	245	0.142	236	0.117	245	0.134	245	0.124	245	0.132	238	0.112
6	246	0.144	237	0.117	246	0.136	246	0.125	246	0.134	239	0.112
7	247	0.145	238	0.117	247	0.136	247	0.126	247	0.134	240	0.112
8	248	0.146	239	0.117	248	0.136	248	0.127	248	0.135	241	0.113
9	249	0.146	240	0.117	249	0.137	249	0.127	249	0.135	246	0.113
10	250	0.147	241	0.117	250	0.138	250	0.128	250	0.135	247	0.113

Approach 4: Fourthly, we consider use of image entropy to select appropriate bands. The entropy measure was developed in Shannon (1948), and is sometimes known as the Shannon entropy. The entropy of a random variable is a numerical measure of its randomness or uncertainty or unpredictability, defined in terms of its probability distribution. For a random variable $X \in \mathbb{X}$, e.g. grey level pixel

intensity, with probability distribution $p(x)$, the entropy $H(X)$ is defined as

$$H(x) = -\Sigma_{x\in\mathbb{X}}\, p(x) log_2 p(x). \tag{9.1}$$

Low entropy of a random variable means that the value is easily predictable, whereas a high entropy means that the value is hard to predict, and zero entropy means there is no uncertainty. For example, in the case of a uniform distribution, all possible values of a random variable have equal probability. Hence it has the maximum entropy value, which means more uncertainty. On the other hand, if prior knowledge about the non-uniformity of the outcomes is available, the random variable will have lower entropy, i.e. less uncertainty, and more predictability. In the context of an image, larger entropy corresponds to greater variability in the image intensities, i.e. an image with a wider range of intensities produces higher entropy. An image containing a more limited range of intensities produces smaller entropy, and an image with constant intensities (perfectly flat) has zero entropy.

For two random variables $X \in \mathbb{X}$ and $Y \in \mathbb{Y}$, if $p(i,j)$ represents the probability of joint occurrence of $X = i$ and $Y = j$, the entropy of this joint event can be defined as:

$$H(x,y) = -\Sigma_{i,j}^{m,n} p(i,j) log_2 p(i,j) \tag{9.2}$$

where m and n are the number of possible values of X and Y respectively. The entropy of X and Y separately can be derived from the joint entropy as

$$H(x) = -\Sigma_{i,j}^{m,n} p(i,j) log_2 \Sigma_j p(i,j) \text{ and } H(y) = -\Sigma_{i,j}^{m,n} p(i,j) log_2 \Sigma_i p(i,j). \tag{9.3}$$

If X and Y are independent, their joint entropy is equivalent to the sum of their marginal entropies.

We investigated different approaches for computing entropy, using all 250 sub-images extracted from all available spectral bands, shown in Figure 9.4.

Method I: Matlab's 'entropy' function computes histogram-based entropy measures. The steps involved are:

1. convert any class other than logical to an 8-bit image, so that the pixel values are discrete (ranging from 0 to 255) and directly correspond to a bin value of the histogram,

2. specify number of bins and compute the grey level frequency histogram, which provides a vector p of length equivalent to the number of bins,

3. replace 0 entries of p by 10^{-4}, to allow computation of log 0,

4. divide p by the total number of pixels in the original image to obtain a relative frequency vector,

5. compute entropy using p as given in (9.1).

Figure 9.4 was produced using method I with 16, 32, 64, 128 and 256 bins, to see the effect of number of bins on entropy. Comparing entropy patterns in Figure 9.4, we see that number of bins affects the value of the entropy but not the pattern of entropy plotted against spectral band. We use different scaling on the y-axes, to preserve the patterns of the curves, but these are easily comparable with each other.

In general lower entropies correspond to the bands around 68 to 170 for all tea types except Tea 2. Bajcsy and Groves (2004) used spectral bands corresponding to high entropy and argued that bands with higher entropy provide more information. Therefore we would like to select spectral bands with higher entropy. Although the first 55 bands have higher entropies, we do not consider them as these may be due to noise and considered the last 50 bands for all tea types for features extraction as they have the highest entropies.

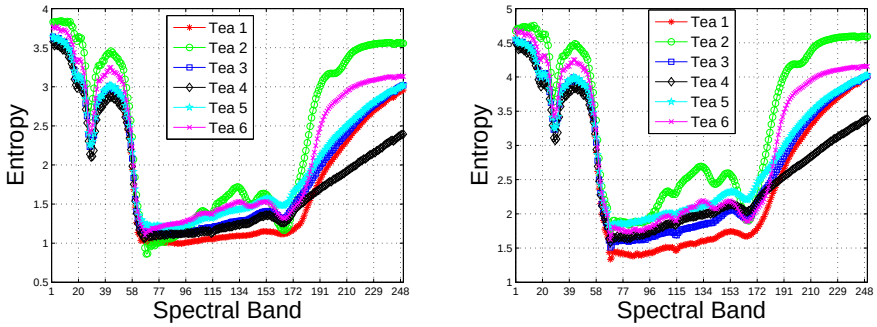
Method II: The entropy of an individual image can also be derived from the joint entropy of two images as given in (9.3). Matlab's 'hist2' function was used to compute the joint histogram of the two images and the individual entropies were derived afterwards. The steps are:

1. a scaling step, i.e. quantise and round the image intensities, so that the intensities are integer and directly fall into one of the bins {0,1,...,L-1}, as $I = round((I - ma) * (L-1)/(MA - ma))$ and $J = round((J - mb) * (L-1)/(MB - mb))$ where L is number of bins, ma and mb are the minimum intensities and MA and MB are the maximum intensity of image I and J respectively,

2. compute the joint histogram h of the scaled images, which is a 2-D array,

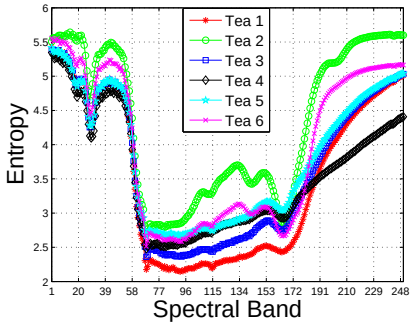
3. replace the 0 entries in h by 10^{-4},

4. divide h by the total number of pixels in either of the original images to obtain a relative frequency matrix,

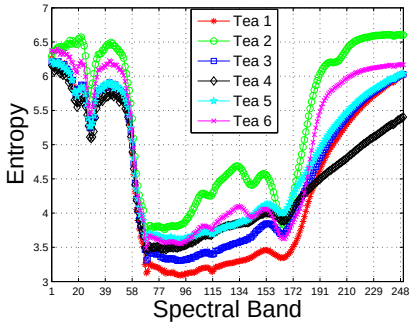


(a) Entropy using 16 bins

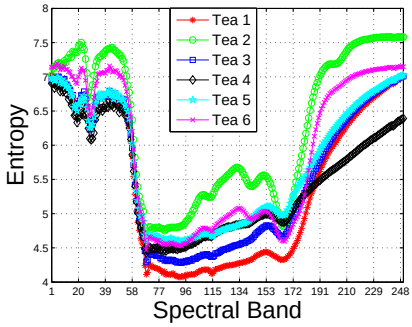
(b) Entropy using 32 bins



(c) Entropy using 64 bins



(d) Entropy using 128 bins



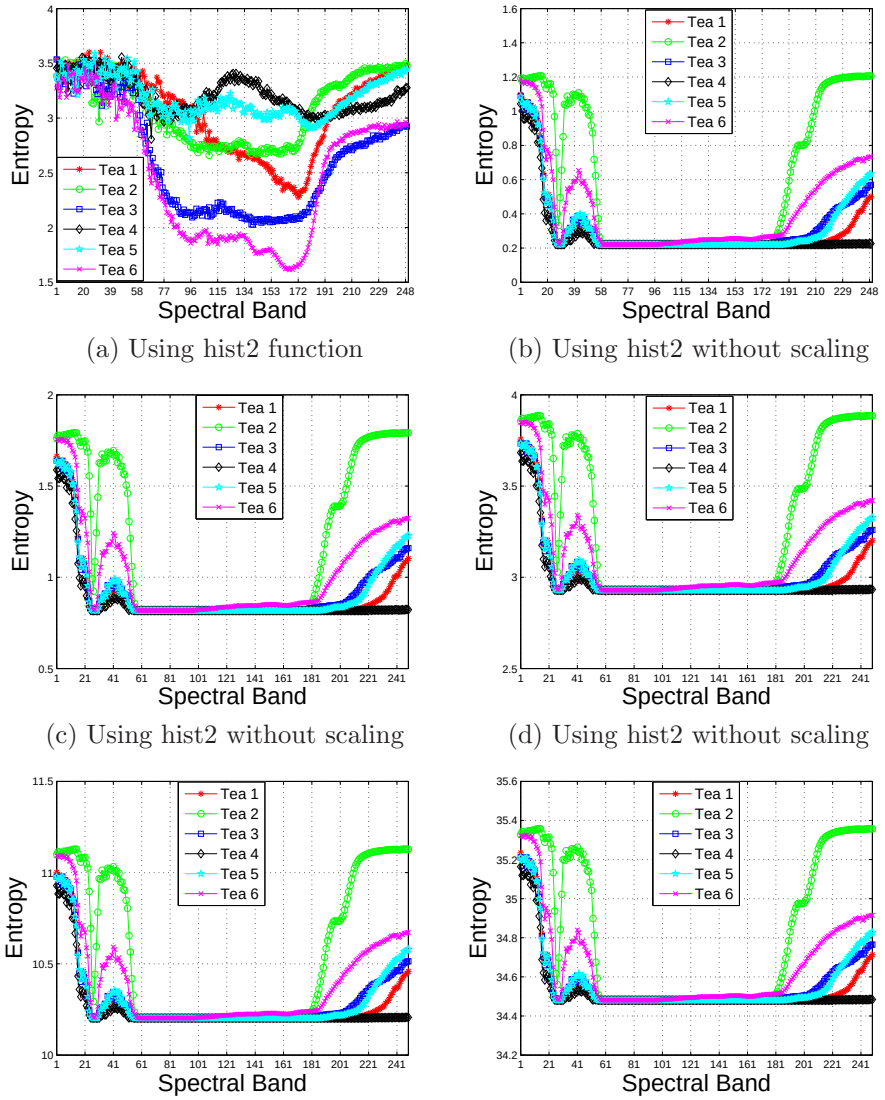
(e) Entropy using 256 bins

Figure 9.4: Entropy information against spectral band using Method I for bin sizes 16, 32, 64, 128 and 256.

5. compute marginal entropies as in (9.3).

Figure 9.5 (a) represents entropies computed using Method II for 16 bins, where the patterns are not similar to those in Figure 9.4. However, the entropies in Figure 9.5 (b) were also computed using Method II but without scaling the image intensities (step 1 of Method II), and these show very similar patterns to those in Figure 9.4.

We observed the effect of scaling used in Method II in simple examples, shown in Appendix V. Consider a 10^2 image with intensity ranging from 0 to 232. Scaling to $I = round((I - ma) * (L-1)/(MA - ma))$, where $L = 16$ is the number of bins, ma and MA are the minimum and maximum intensities of image I respectively,



(a) Using hist2 function

(b) Using hist2 without scaling

(c) Using hist2 without scaling

(d) Using hist2 without scaling

(e) Using hist2 without scaling

(f) Using hist2 without scaling

Figure 9.5: Entropy information against spectral band using Matlab's hist2 function (a) with scaling for 16 bins and (b)–(f) without scaling for 16, 32, 64, 128 and 256 bins.

gives image intensities scaled from 0 to 15. The histograms of the original image and the scaled image are very similar. Although the scaling was sensible, we prefer applying 'hist2' but without scaling of the intensities.

Approach 5: Fifthly, we computed the mutual information (MI) for adjacent pairs of bands, as used in Ren et al. (2011). The MI between bands i and $i+1$, $I(x_i, x_{i+1})$, is defined as

$$I(x_i, x_{i+1}) = H(x_i) + H(x_{i+1}) - H(x_i, x_{i+1}) \tag{9.4}$$

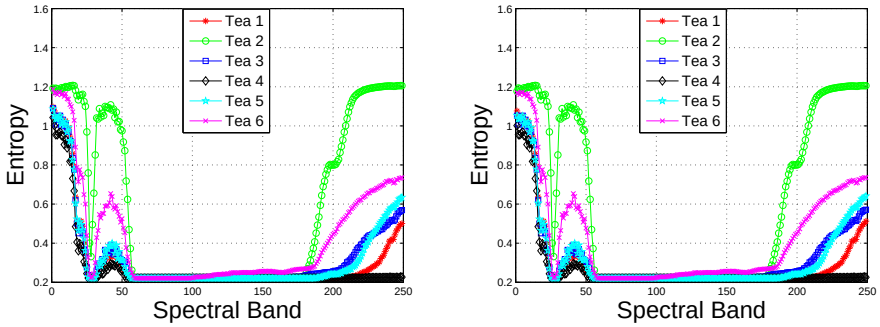
where $H(x_i)$, $H(x_{i+1})$ and $H(x_i, x_{i+1})$ are the individual entropies of bands i and $i+1$ and their joint entropy respectively.

We first consider Method II but without scaling the intensities, to compute individual entropies and joint entropies. Since there are 250 bands, we obtained 249 MIs, which are plotted against spectral band-pair in Figure 9.6. Figures 9.6 (a) and (b) represent the entropies of spectral bands $1, 2, \ldots, 249$ and $2, 3, \ldots, 250$ respectively, (c) shows the joint entropy and (d) represents the MI between adjacent pairs of bands, for 16 bins. For all tea types the MI is almost constant at about 0.1 up to spectral band 110, except Tea 2 which has some higher entropies around band 30. The MI starts to increase at spectral band 110, except for Tea 4, but the rate of increase is higher for Tea 2.

Figure 9.7 shows MIs using different numbers of bins, namely, for bin size 32, 64, 128 and 256. Any number of bins gives a similar pattern. However 128 and 256 bins produced negative MIs, so the individual entropies and joint entropies using 128 bins were computed and are shown in Figure 9.8. For 128 bins, the joint entropies are higher than using fewer bins. Therefore, the MIs are negative as the MI is the sum of the individual entropies minus the joint entropy.

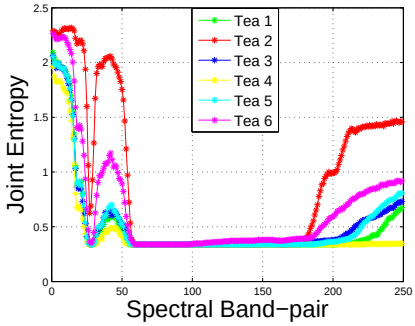
For comparison, the MIs using method II and with scaling of the intensities were also computed for 16 bins. Figure 9.9 shows the entropies for bands $1, 2, \ldots, 249$ and $2, 3, \ldots, 250$, their joint entropies and MIs for 16 bins. In general the MIs for all tea types increase with spectral band-pair after 60 (Figure 9.9(d)).

Ren et al. (2011) suggested ranking the MIs and selecting the lowest m values from the ranked MIs, which can split the total available bands into $m+1$ groups using the bands corresponding to the lowest MI as boundaries separating the groups. Then one representative band from each group with the smallest summed mean squared difference or summed mean absolute difference from each of the other bands in the group was selected. In our case, the MI for each tea type increases with spectral band after around 60, if we consider scaling the image intensities (Figure 9.9). However, if we do not scale the image intensities, for any number of bins, the MI starts to increase from the beginning for all tea types, peaks at around band 25 and then drops again. After band 50 the MIs again increase with band number for all tea types. Selecting the smallest m

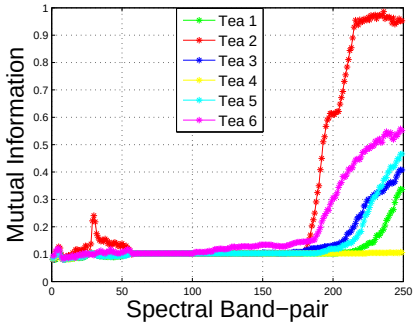


(a) Entropy of $1, 2, \ldots, 249$

(b) Entropy of $2, 3, \ldots, 250$



(c) Joint Entropy of adjacent pairs of bands



(d) MI of adjacent pairs of bands

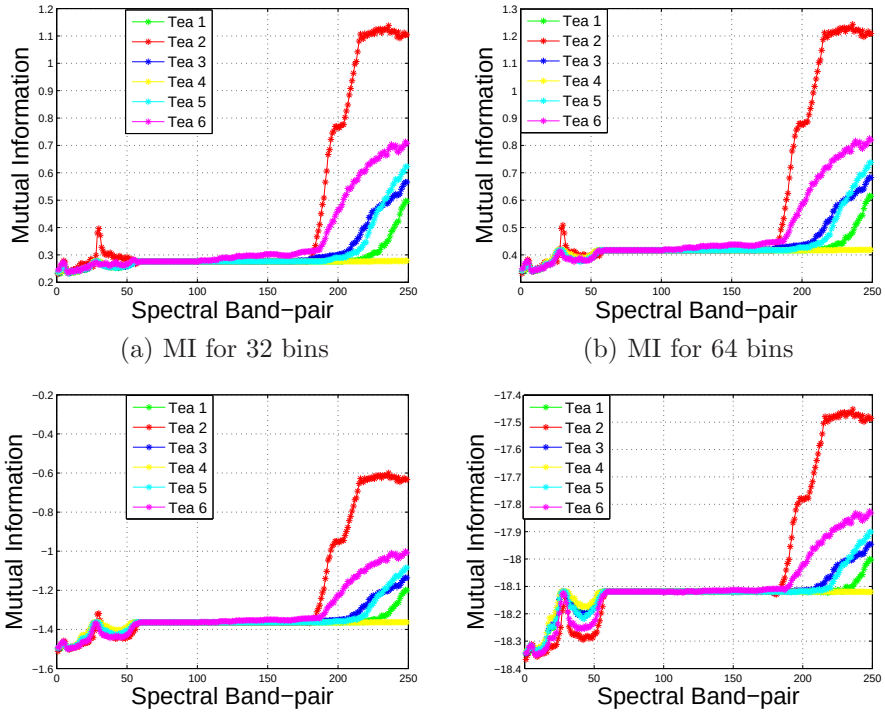
Figure 9.6: Entropy and mutual information (MI) for 16 bins using Matlab's 'hist2' function but without scaling the image intensities; (a) Entropy of bands $1, 2, \ldots, 149$; (b) Entropy of bands $2, 3, \ldots, 250$; (c) Joint entropy of adjacent pairs of bands, i.e. 1 and 2, 2 and 3, and so on; (d) MI of adjacent pairs of bands.

MIs in either case will not divide the bands into groups unlike the situation in Ren et al. (2011). Therefore, we consider the 10 bands with lowest MIs for feature extraction. Again we avoid the first 50 bands due to the presence of noise, although some of them have lower MIs.

9.6 Computation of PS Moments

Granulometry is now applied on all 70^2 sample images extracted from selected spectral bands for all tea types using three different SEs, i.e. a square, disk and rectangle. We used square SEs of base length sizes $2j-1,\ j = 1, 2, \ldots$, and disk SEs of radius $2j-1,\ j = 1, 2, \ldots$ and rectangle SEs of size $j \times 2j,\ j = 1, 2, \ldots$, where j is the number of rows and $2j$ is the number of columns.

Firstly we computed the first four PS moments from each SE from all 250 sub-images extracted from all available bands. The average PS moments are plotted against tea type in Figure 9.10. Tea types are quite distinguishable in



(a) MI for 32 bins

(b) MI for 64 bins

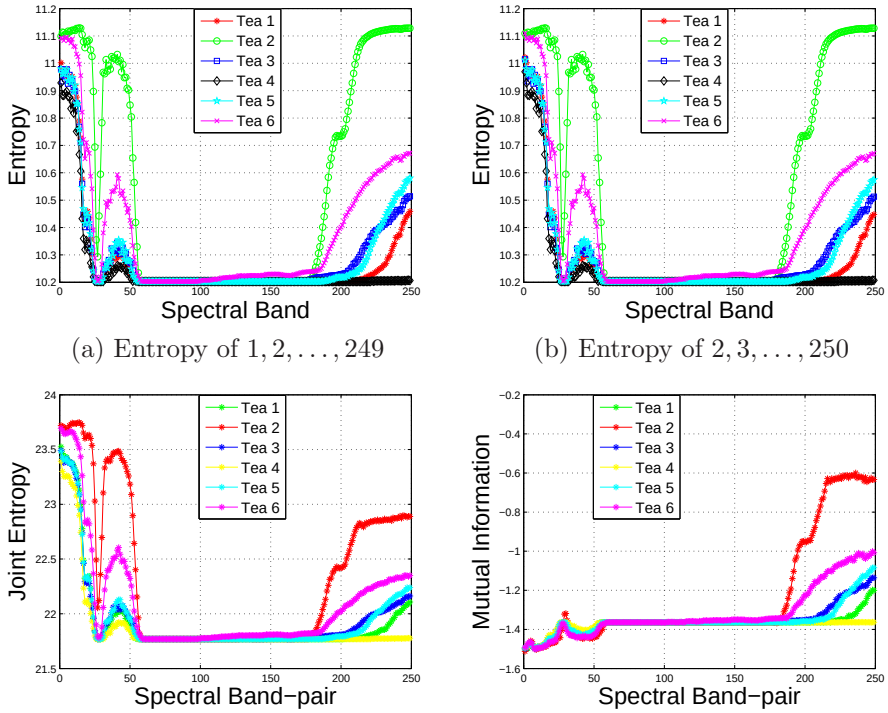
(c) MI for 128 bins

(d) MI for 256 bins

Figure 9.7: MI of adjacent pairs of bands using Matlab's 'hist2' function but without scaling the image intensities for (a) 32; (b) 64; (c) 128 and (d) 256 bins.

terms of PS mean from each SE, i.e. Teas 1, 2 and 6 are clearly distinct from each other for any PS mean but Teas 3, 4 and 5 are similar as their PS means are only slightly different from each other. A similar situation is observed for PS sd. PS skewness from a disk SE clearly distinguished the tea types, however the square and rectangle SEs produced similar skewness for all tea types. A rectangle SE produced the same kurtosis for all tea types but they are distinguishable in terms of kurtosis from the square and disk SEs, except teas 3 and 4. Although the average PS skewness is not very close to zero, most of the individual skewness values are close to zero. Consequently, only the PS mean, sd and kurtosis are used in classification.

PS moments were computed from all 70^2 sub-images of the first 10 PC images derived from each tea type, and are shown in Table 9.3. The PS mean and sd for Tea 1 is similar to Tea 4, whereas Tea 3 is similar to Tea 5, and Tea 6 has the highest PS mean and sd except for the sd from the rectangle SE. Skewness is zero (up to two decimal place) for all SEs, hence it is not shown in the table. Tea 6 has negative kurtosis for all SEs while the other tea types have positive kurtosis. The PS moments from the 10 bands with the strongest PC coefficients were also computed. The moments for Teas 3 and 4 are very similar and Tea 2 has the



(a) Entropy of $1, 2, \ldots, 249$

(b) Entropy of $2, 3, \ldots, 250$

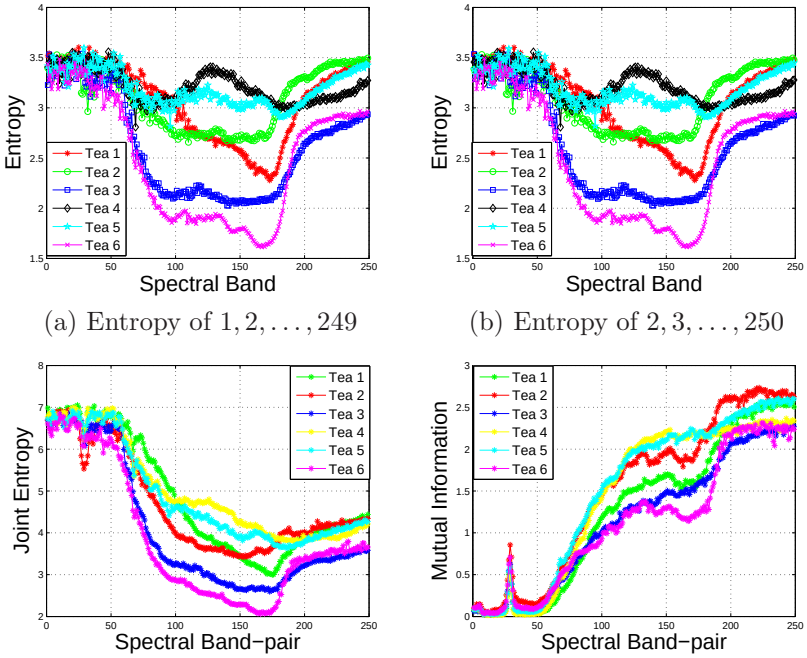
(c) Joint Entropy of adjacent pairs of bands

(d) MI for 128 bins

Figure 9.8: Entropy and MI for 128 bins using 'hist2' function but without scaling the image intensities; (a) Entropy of bands $1, 2, \ldots, 149$; (b) Entropy of bands $2, 3, \ldots, 250$; (c) Joint entropy of adjacent pairs of bands, i.e. 1 and 2, 2 and 3, and so on; (d) MI of adjacent pairs of bands.

highest PS moments for all SEs. We use the first two moments and kurtosis from all SEs in classification.

The average PS moments from the 10 bands with highest entropies and the 10 bands with lowest MI, using 3 SEs, are also shown in Table 9.3. The PS means and sds for the bands with highest entropies are higher compared to the means and sds from the 10 bands with lowest MIs. Again skewness from all SEs is zero (up to 4 decimal places), hence these are not shown. Therefore, again we use 9 PS moments (mean, sd, and kurtosis from each SE) from both sets of bands for classification.



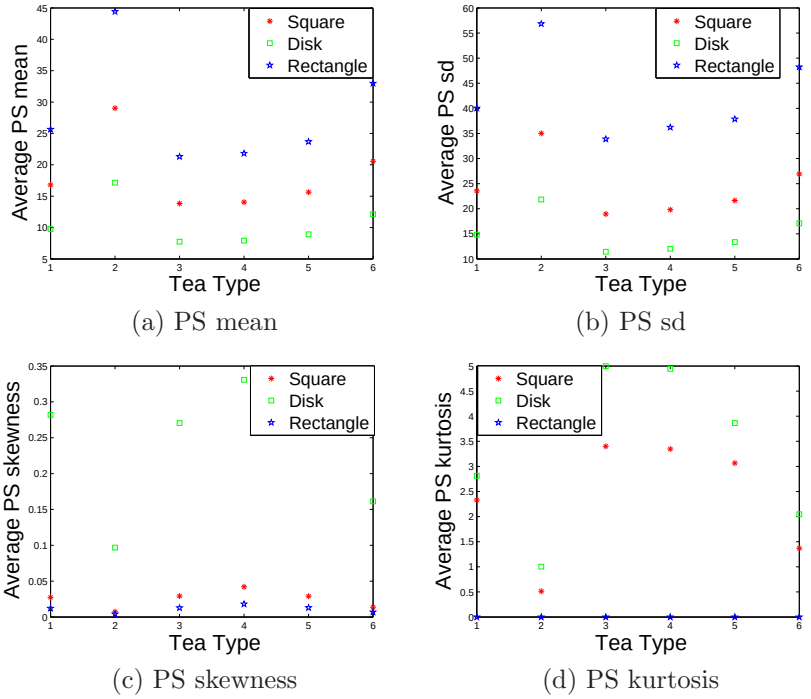
(a) Entropy of $1, 2, \ldots, 249$

(b) Entropy of $2, 3, \ldots, 250$

(c) Joint Entropy of adjacent pairs of bands

(d) MI for 16 bins

Figure 9.9: Entropy and MI for 16 bins using 'hist2' function with scaling of the image intensities; (a) Entropy of bands $1, 2, \ldots, 149$; (b) Entropy of bands $2, 3, \ldots, 250$; (c) Joint entropy of adjacent pairs of bands, i.e. 1 and 2, 2 and 3, and so on; (d) MI of adjacent pairs of bands for 16 bins.



(a) PS mean

(b) PS sd

(c) PS skewness

(d) PS kurtosis

Figure 9.10: Plots of the average PS moments against tea type, for square, disk and rectangular SEs, using all 250 sample images from each tea type.

Table 9.3: Average PS moments computed from different sets of spectral bands.

PS moments from the first 10 PCs									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	24.79	14.05	35.98	34.31	20.24	53.13	1.51	1.73	2.44
Tea 2	31.62	18.57	45.39	35.37	21.48	52.27	0.16	0.42	0.12
Tea 3	29.95	17.61	44.43	37.56	23.06	59.62	0.88	1.11	2.09
Tea 4	24.76	14.29	35.40	33.47	20.17	50.28	1.34	1.51	2.54
Tea 5	30.41	17.40	44.56	38.28	22.65	60.57	0.70	0.96	1.67
Tea 6	35.30	20.63	50.82	39.73	24.10	58.00	-0.25	-0.01	-0.23
PS moments from 10 bands with highest PC1 coefficients									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	29.98	18.02	46.86	37.86	24.44	66.25	0.08	0.36	0.74
Tea 2	51.66	30.13	68.76	53.00	31.60	67.98	-1.43	-1.40	-1.23
Tea 3	22.63	12.82	36.19	30.43	18.22	57.59	2.66	2.92	5.10
Tea 4	22.30	12.84	36.31	32.26	19.65	61.95	2.26	2.43	3.97
Tea 5	25.92	15.06	41.99	35.06	21.81	66.03	1.04	1.58	2.11
Tea 6	34.40	21.12	54.72	40.21	26.93	72.04	-0.52	-0.15	0.24
PS moments from the 10 bands with highest entropies									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	29.98	18.02	46.86	37.86	24.44	66.25	0.08	0.74	0.74
Tea 2	53.56	31.27	71.10	53.58	31.99	68.89	-1.54	-1.29	-1.29
Tea 3	22.63	12.82	36.19	30.43	18.22	57.59	2.66	5.10	5.10
Tea 4	22.30	12.84	36.31	32.26	19.65	61.95	2.26	3.97	3.97
Tea 5	25.92	15.06	41.99	35.06	21.81	66.03	1.04	2.11	2.11
Tea 6	34.95	21.48	55.69	40.50	27.14	72.41	-0.57	0.16	0.16
PS moments from the 10 bands with lowest MI									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	17.35	9.78	25.45	25.22	15.09	40.72	4.61	5.34	8.03
Tea 2	30.62	18.54	49.58	40.41	26.07	70.15	0.38	0.63	0.79
Tea 3	14.27	7.84	19.98	18.94	10.85	26.46	6.37	6.22	8.99
Tea 4	15.95	9.10	23.30	22.03	13.49	35.43	5.67	6.04	10.48
Tea 5	16.05	8.93	22.69	21.17	12.18	34.13	5.84	5.55	8.33
Tea 6	23.85	13.97	38.16	33.72	21.13	59.62	1.23	1.94	2.41

9.6.1 Separability measures

Different scatter matrix-based class separability measures, as described in Section 7.5.1, are computed to indicate the separation ability of different sets of bands. The value of $Tr\{\boldsymbol{S_W}\}$, $Tr\{\boldsymbol{S_B}\}$, J_1, J_2 and J_3 for each set of bands are recorded in Table 9.4. A large $Tr\{\boldsymbol{S_B}\}$ indicates good separation of the classes and a small $Tr\{\boldsymbol{S_W}\}$ indicates that the data points are well clustered around their mean within each class. A smaller value of $Tr\{\boldsymbol{S_W}\}$ or $\boldsymbol{S_W}$ compared to $Tr\{\boldsymbol{S_B}\}$ or $\boldsymbol{S_B}$ generates larger values of J_1, J_2 and J_3. Larger values of these measures indicates stronger clustering of the feature set vector for each case around the mean vector in that class, hence greater class separability.

The smallest $Tr\{\boldsymbol{S_W}\}$ corresponds to the 10 bands with the lowest MI, which also produced the largest $Tr\{\boldsymbol{S_B}\}$, and so the highest value of J_3. The 10 bands with the strongest PC1 coefficients produced a higher $Tr\{\boldsymbol{S_W}\}$ but quite a high value of $Tr\{\boldsymbol{S_B}\}$, and produced the largest values of J_1 and J_2, confirming the greater separability of the data. The second best set of features correspond to the 10 bands with highest entropies as they produced the second largest values of J_1 and J_2.

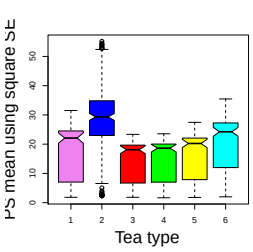
Table 9.4: Separability measures for the PS moments computed from different sets of bands $J_1 = Tr\{\boldsymbol{S_W}^{-1}\boldsymbol{S_B}\}$, $J_2 = \frac{|\boldsymbol{S_T}|}{|\boldsymbol{S_W}|}$ and $J_3 = \frac{Tr\{\boldsymbol{S_B}\}}{Tr\{\boldsymbol{S_W}\}}$.

Set of bands	$Tr\{\boldsymbol{S_W}\}$	$Tr\{\boldsymbol{S_B}\}$	J_1	J_2	J_3
All bands	287.421	6.246	0.851	1.851	0.022
First 10 PC images	410.540	1.772	0.406	1.406	0.004
10 bands with strongest PC1 coefficients	483.800	9.859	**4.172**	**5.172**	0.020
10 bands with highest entropies	492.148	11.190	**3.633**	**4.633**	0.023
10 bands with lowest MI	**227.818**	**11.376**	1.957	2.957	**0.050**

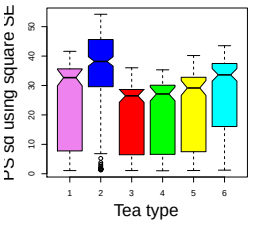
9.6.2 Graphical presentation of different sets of PS moments

The first four PS moments using a square, a disk and a rectangular SE from the different sets of bands are presented using box plots. This provides an impression about the distribution and variability of the moments from the different tea types as well as the discrimination ability of different sets of PS moments.

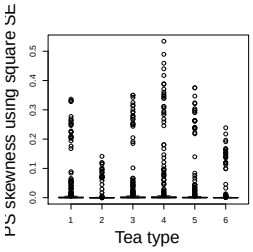
Using all 250 images: The box plots of the PS moments computed using 3 SEs are shown in Figure 9.11. Based on the PS means and sds using all SEs, only Tea 2 is reasonably different from the other tea types, although outliers are present in some sets of moments. The levels of the boxes for other tea types, especially for Teas 3 and 4, are quite similar, indicating lower discrimination ability. The lower and upper quartiles and the median of the PS skewness are close to zero for all SEs, with many outliers, as seen in Figure 9.11 (c), (g) and (k). The box plots of the PS kurtosis for the different tea types are more symmetrical compared to those of the PS mean and sd, and Teas 2 and 6 are dissimilar from Teas 3–5 and Tea 1 is a little different from the rest.



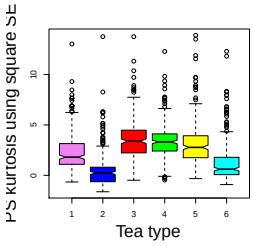
(a) PS mean using square SE



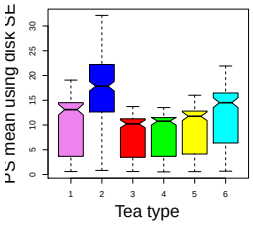
(b) PS sd using square SE



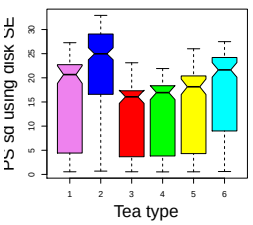
(c) PS skewness using square SE



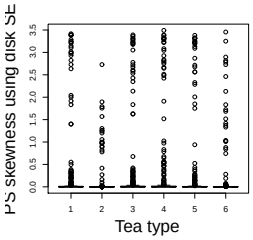
(d) PS kurtosis using square SE



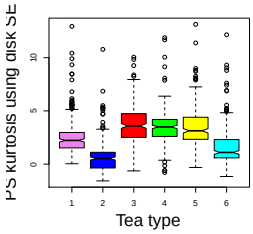
(e) PS mean using disk SE



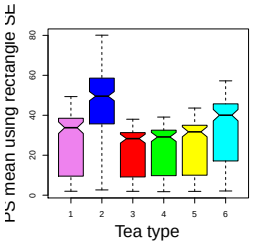
(f) PS sd using disk SE



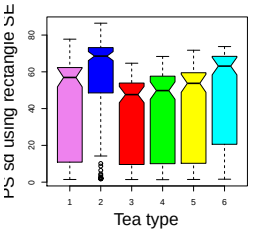
(g) PS skewness using disk SE



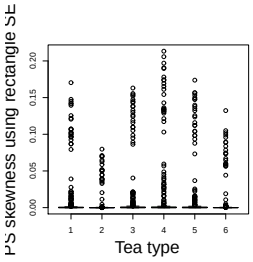
(h) PS kurtosis using disk SE



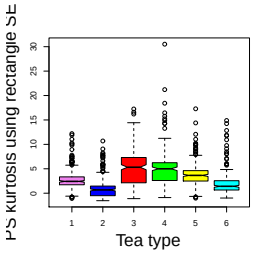
(i) PS mean using rectangle SE



(j) PS sd using rectangle SE



(k) PS skewness using rectangle SE

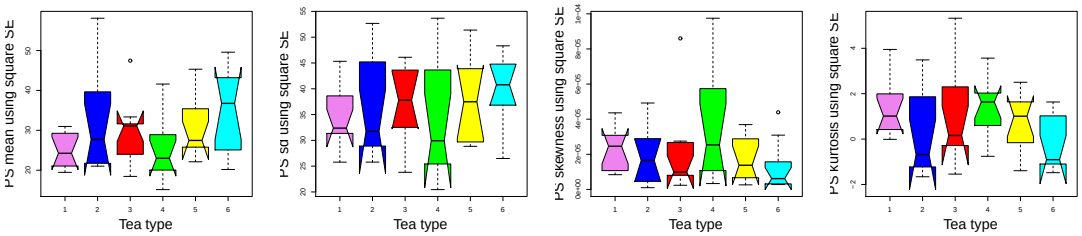


(l) PS kurtosis using rectangle SE

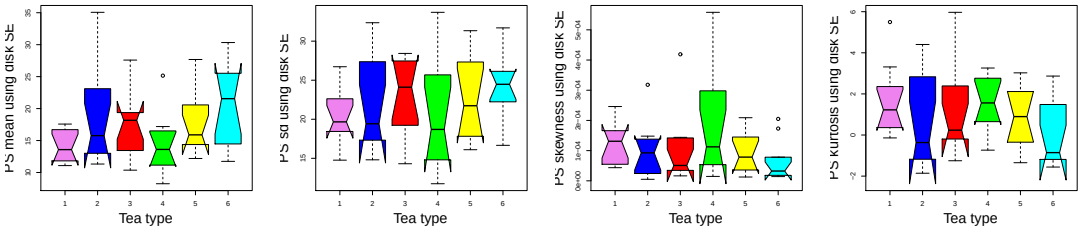
Figure 9.11: Box plots of the PS moments against tea type, for square, disk and rectangular SEs, using all 250 bands from each tea type.

Using the first 10 PC images: The box plots of the first 4 PS moments from all 3 SEs are shown in Figure 9.12. The moments are rarely symmetric, with a few outliers for some sets of moments. The levels for different teas differ

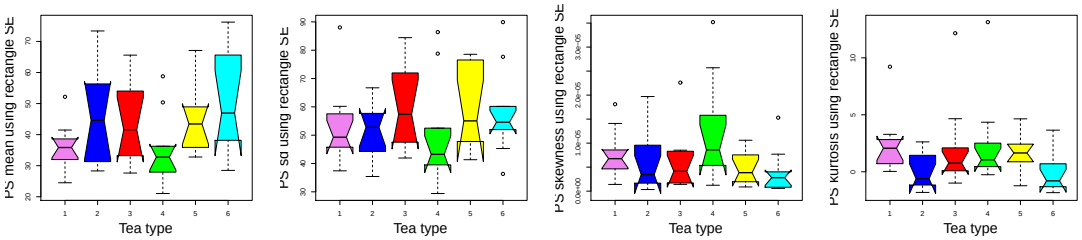
a bit but they do overlap, suggesting higher classification error using this set of moments.



(a) PS mean using square SE (b) PS sd using square SE (c) PS skewness using square SE (d) PS kurtosis using square SE



(e) PS mean using disk SE (f) PS sd using disk SE (g) PS skewness using disk SE (h) PS kurtosis using disk SE

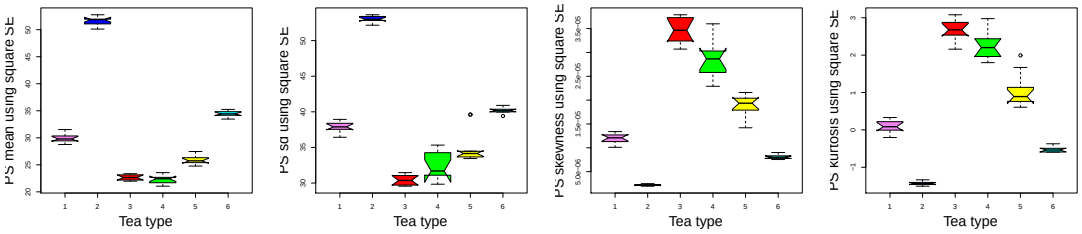


(i) PS mean using rectangle SE (j) PS sd using rectangle SE (k) PS skewness using rectangle SE (l) PS kurtosis using rectangle SE

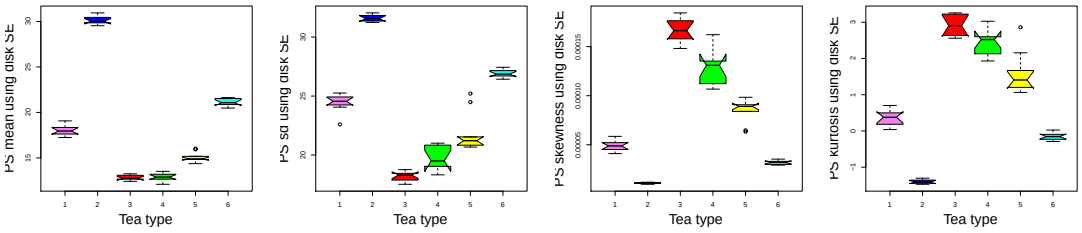
Figure 9.12: Box plots of the PS moments against tea type, for square, disk and rectangular SEs, using the first 10 PC images for each tea type.

Using the 10 bands with highest PC1 coefficients and with highest entropies: Figures 9.13 and 9.14 represent the box plots of these PS moments. The corresponding plots in both figures are similar, as both methods selected bands from the upper end of the spectral region. All tea types are less variable for both these sets of moments and are different from each other. Therefore these sets of moments are expected to give the best classification results, and these plots confirm the results seen in the separability measures. However many of the plots are not symmetric.

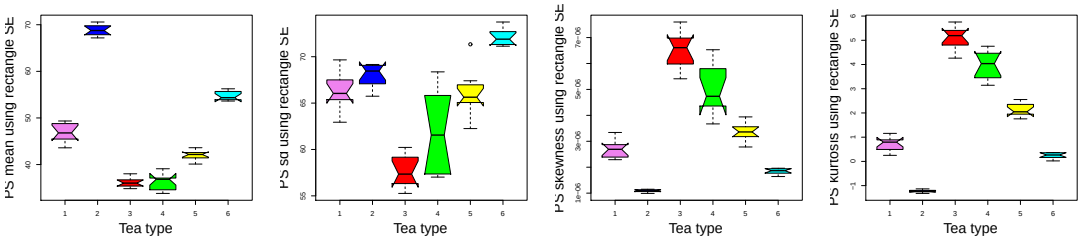
Using the 10 bands with lowest MI: The PS moments are able to distinguish clearly only Teas 2 and 6 (Figure 9.15). This set of moments provides better separation than the moments using all bands and the 10 PC images but



(a) PS mean using square SE (b) PS sd using square SE (c) PS skewness using square SE (d) PS kurtosis using square SE



(e) PS mean using disk SE (f) PS sd using disk SE (g) PS skewness using disk SE (h) PS kurtosis using disk SE

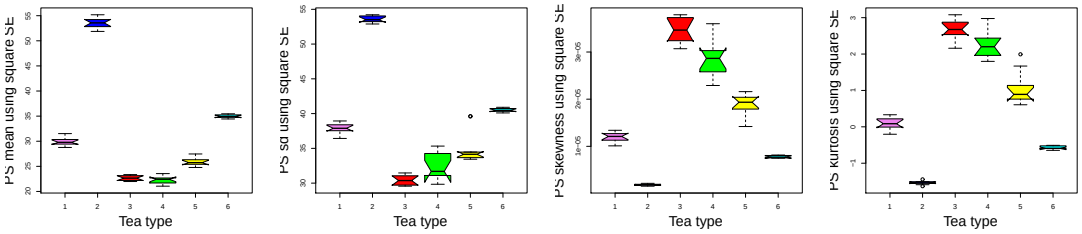


(i) PS mean using rectangle SE (j) PS sd using rectangle SE (k) PS skewness using rectangle SE (l) PS kurtosis using rectangle SE

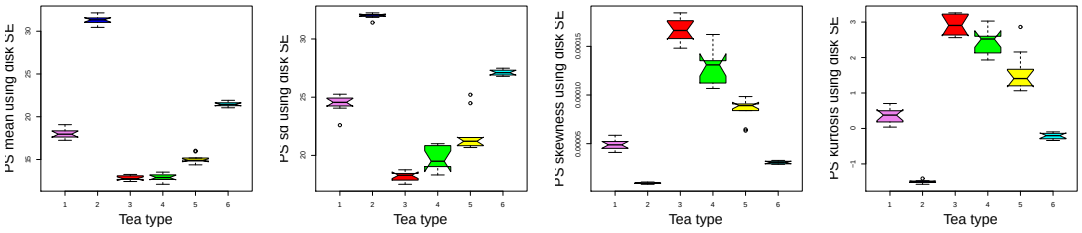
Figure 9.13: Box plots of the PS moments against tea type, for square, disk and rectangular SEs, using the 10 bands with highest PC1 coefficients for each tea type.

worse than the PS moments from the 10 bands with highest PC coefficients or entropies.

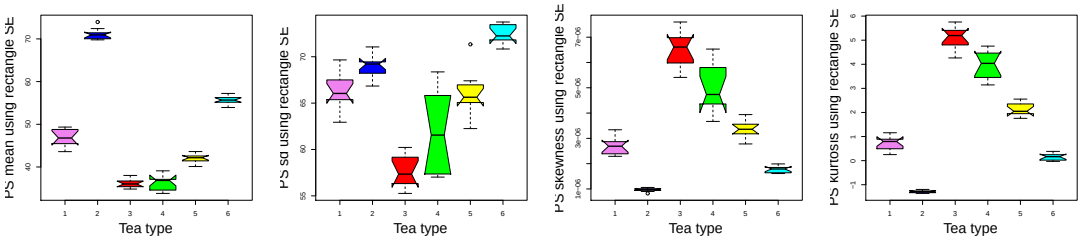
Most of the PS moments appear to have asymmetric distributions in Figures 9.11– 9.15. Also the adjacent bands are correlated, not independent. Consequently, ANOVA or MANOVA would not be appropriate tests for testing the differences between the means of the PS moments from different tea types. Therefore, we consider the Kruskal-Wallis non-parametric test as a robust alternative to ANOVA to test for differences between the tea types in terms of the PS moments. The Kruskal-Wallis test only investigates whether the different populations generating the samples have the same median. The p-values for different sets of moments from each set of bands are shown in Table 9.5. For all sets of bands except the first 10 PC images, the p-values for the test statistics from the PS



(a) PS mean using square SE (b) PS sd using square SE (c) PS skewness using square SE (d) PS kurtosis using square SE



(e) PS mean using disk SE (f) PS sd using disk SE (g) PS skewness using disk SE (h) PS kurtosis using disk SE



(i) PS mean using rectangle SE (j) PS sd using rectangle SE (k) PS skewness using rectangle SE (l) PS kurtosis using rectangle SE

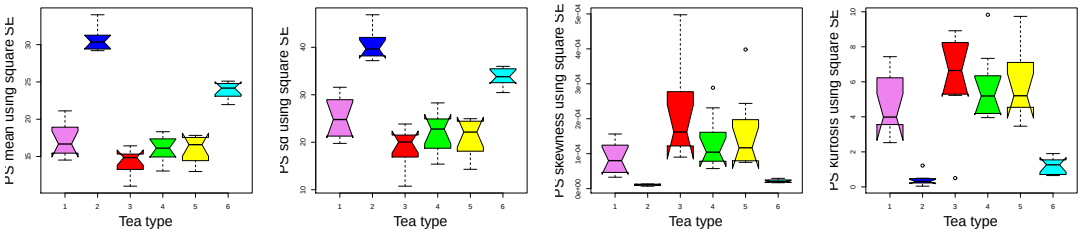
Figure 9.14: Box plots of the PS moments against tea type, for square, disk and rectangular SEs, using the 10 bands with highest entropies for each tea type.

mean, sd and kurtosis from all 3 SEs are near zero, strongly indicating different median values of the PS moments between the tea types.

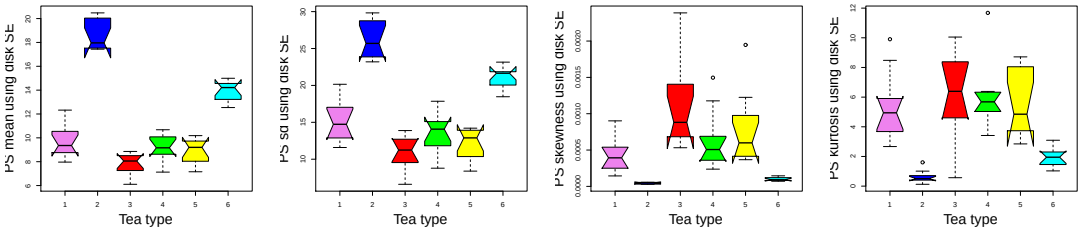
9.6.3 PCA on different sets of PS moments

We examined the clustering ability of PCA using different sets of moments before applying any supervised classification techniques. PCA was applied to the 9 PS moments from the different sets of bands, namely all 250 bands, the first 10 PC images and the 10 bands with the strongest PC1 coefficients, the 10 bands with the highest entropy and the 10 bands with the lowest MI. The first two PCs from each set of bands are plotted against each other in Figure 9.16.

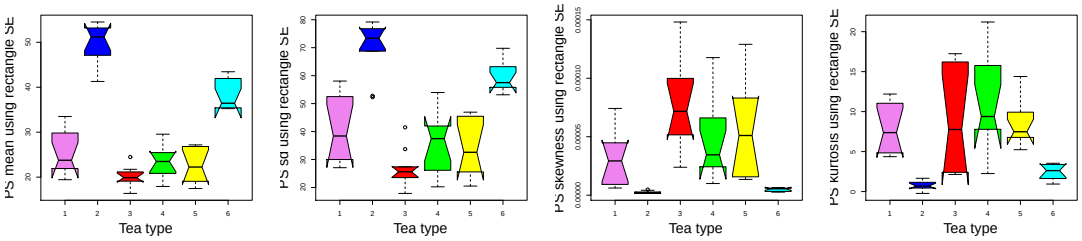
Neither the PCs from all bands nor the PCs from the first 10 PC images were able to discriminate the tea types. However, the first two PCs computed from the



(a) PS mean using square SE (b) PS sd using square SE (c) PS skewness using square SE (d) PS kurtosis using square SE



(e) PS mean using disk SE (f) PS sd using disk SE (g) PS skewness using disk SE (h) PS kurtosis using disk SE



(i) PS mean using rectangle SE (j) PS sd using rectangle SE (k) PS skewness using rectangle SE (l) PS kurtosis using rectangle SE

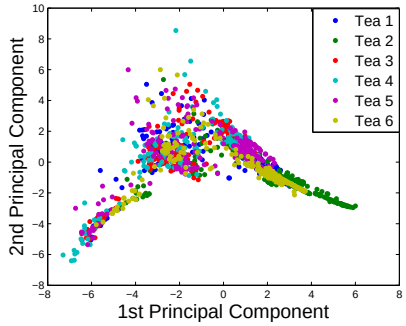
Figure 9.15: Box plots of the PS moments against tea type, for square, disk and rectangular SEs, using the 10 bands with lowest MI for each tea type.

10 bands with the strongest PC1 coefficients and the 10 bands with the highest entropies were able to distinguish all tea types (Figures 9.16 (c) and (d)), except some images of Teas 3 and 4. Both techniques selected bands from the upper end of the available spectral bands (see Table 9.2 and Figure 9.5). The 10 bands with minimum MI were selected from the middle of the spectral bands (Figure 9.7).

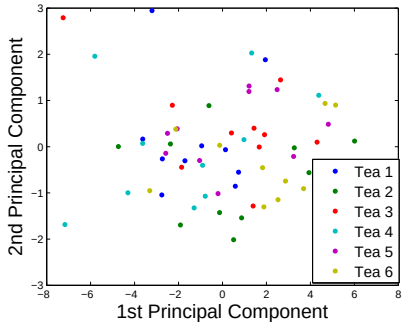
Table 9.5: P-values of Kruskal-Wallis tests; one for each PS moment from each SE and from each set of bands.

Bands	PS mean			PS sd		
	Square	Disk	Rect.	Square	Disk	Rect.
All bands	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
First 10 PC images	0.06	0.45	0.06	0.41	0.07	0.45
10 bands with strongest PC1 coefficients	$8.9e^{-11}$	$2.8e^{-10}$	$1.1e^{-10}$	$9.4e^{-11}$	$1.3e^{-11}$	$1.3e^{-8}$
10 bands with highest entropies	$8.9e^{-11}$	$2.5e^{-10}$	$9.1e^{-11}$	$1.1e^{-10}$	$9.4e^{-11}$	$6.6e^{-9}$
10 bands with lowest MI	$1.3e^{-8}$	$1.8e^{-8}$	$1.0e^{-8}$	$9.9e^{-9}$	$1.7e^{-8}$	$5.9e^{-8}$

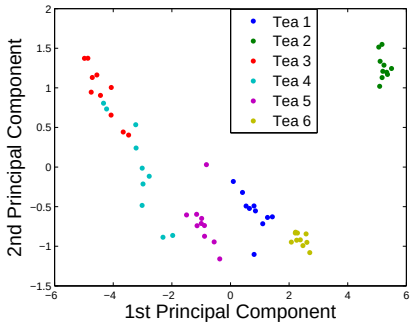
Bands	PS kurtosis		
	Square	Disk	Rect.
All bands	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
First 10 PC images	0.06	0.11	0.02
10 bands with strongest PC1 coefficients	$7.6e^{-11}$	$1.1e^{-10}$	$5.8e^{-11}$
10 bands with highest entropies	$7.6e^{-11}$	$1.1e^{-10}$	$5.4e^{-11}$
10 bands with lowest MI	$8.3e^{-8}$	$2.9e^{-7}$	$9.7e^{-7}$



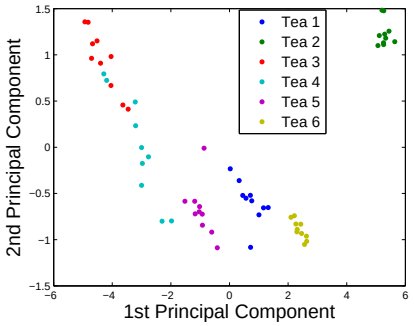
(a) PCs using all bands



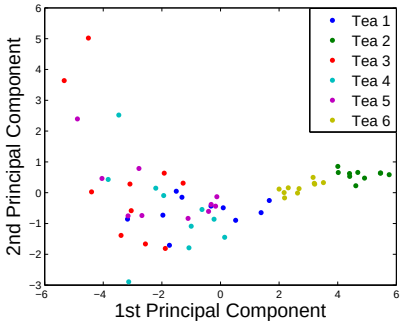
(b) PCs using first 10 PC images



(c) PCs using first 10 informative bands



(d) PCs using first 10 bands with highest entropies



(e) PCs using first 10 bands with lowest MI

Figure 9.16: Scatter plots of the first two PCs derived from different sets of bands.

9.7 Classification using PS moments

Although the separability measures and PCA clustering confirmed that the 10 bands with the strongest PC1 coefficients and the 10 bands with the highest entropies are the best sets of bands, we applied SVM, FF-NNET and LDA to all sets of PS moments and compare their performance. We randomly selected 70% of the available images and used the 9 PS moments from these images to train the classifiers and tested on the rest, repeated the process 10 times and averaged the results over 10 runs to compute the final results. As the ordering of the tea types are arbitrary (Section 9.4), use of Type 1, Type 2 error and MAE as used in previous chapters are meaning less. Therefore we only used Type 0 error, which represents the proportion of misclassification, to assess the performance of each classifier.

For SVM and FF-NNET the optimum parameter values were decided on based on a single set of training data. For SVM, a linear kernel produced 100% correct classification for any cost between 1 and 100 for the PS moments computed from all 250 bands, the last 150 bands, the first 10 PC images, the 10 bands with highest entropies and the 10 bands with lowest MI. However, we also computed the training set error rates using other kernels for some sets of PS moments. For example, a polynomial kernel with $\eta = 1$ produced 100% correct classification for any combination of cost between 1 and 100 and γ between 0.1 to 1 (Table 9.6), using 9 PS moments from all 250 images. The radial basis kernel was only slightly worse as the optimum pairs of cost and γ (any cost of 10 or more with $\gamma = 0.1$ (or $\gamma = 0.2$, not shown here)) produced 99% correct classification.

Table 9.6: Training set error rates for different combinations of cost and γ for the radial basis kernel and the polynomial kernel with $\eta = 1$, using 9 moments, where the training set consists of 70% of the 250 sub-images of each tea type.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.5	0.03	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
0.9	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0	0	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0

Table 9.7: Training set error rates for different combinations of cost and γ for the radial basis kernel and the polynomial kernel with $\eta = 1$, using 9 PS moments, where the training set consists of 70% of the 10 PC images of each tea type.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.22	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
0.5	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
0.9	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
0.5	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22
0.9	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22

We again sought other appropriate kernels and corresponding parameter values for the PS moments from the first 10 PC images. The radial basis kernel produced the lowest error rate of 11% with $\gamma = 0.2$ for any cost of 10 or more. The lowest error rate of 17% for a polynomial kernel with $\eta = 1$ corresponds to any value of cost between 1 and 100 with $\gamma < 0.5$ (Table 9.7). However for all sets of bands, SVM with a linear kernel produced 100% correct classification for any cost between 1 and 100 using a single training set, so we used the linear kernel and a cost of 100 in each case.

The error rates for FF-NNET on normalised PS moments using a single training set with different numbers of hidden neurons are shown in Table 9.8 for different sets of bands. The values of decay and rang were set to be 10^{-4} and $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, as these were the best choice for many sets of features in previous chapters. For all sets of bands 10 hidden units were used as this produced the lowest training set error rate.

Both the average test set and training set error rates (averaged over 10 runs) for different sets of bands were computed to examine whether there was any overfitting of any classifiers. The error rates are shown in Table 9.9. Training and test set error rates for SVM and LDA are comparable but for FF-NNET occasionally the test set error rates are very high compared to the training set error rates, e.g. for the first 10 PC images and the 10 bands with lowest MI, which suggests overfitting.

Considering the test set error rates, SVM yielded better classification results for all 250 bands, however, the performance of LDA and FF-NNET is relatively poor, although FF-NNET is better than LDA. For bands 101-250, again SVM

Table 9.8: Training set error rates from a grid search approach for finding the optimum number of hidden neurons for FF-NNET with rang of $1/\max(|\boldsymbol{x}|)$ using 9 PS moments from different sets of bands.

	Number of units									
Bands	1	2	3	4	5	6	7	8	9	10
1-250	0.57	0.46	0.36	0.34	0.31	0.25	0.29	0.21	0.20	**0.19**
101-250	0.39	0.13	0.09	0.04	0.03	0.02	0	0	0	**0**
First 10 PC images	0.69	0.57	0.14	0.14	0	0	0	0	0	**0**
10 bands with strongest PC1 coefficients	0	0	0	0	0	0	0	0	0	**0**
10 bands with highest entropies	0	0	0	0	0	0	0	0	0	**0**
10 bands with lowest MI	0.40	0.17	0.05	0	0	0	0	0	0	**0**

produced 100% correct classification, LDA a 34.9% error rate and FF-NNET a 7.4% error rate, and all classifiers performed better than using all bands. Using the PS moments from the first 10 PC images, SVM with the linear kernel attained 89% correct classification, but the performance of FF-NNET and LDA was very poor. SVM, LDA and FF-NNET produced very low error rates using the PS moments from both the 10 bands with the strongest PC1 coefficients (1.1% for SVM, 5% for LDA and FF-NNET) and the 10 bands with highest entropies (1.1% for SVM, 6.1% for LDA and 4.4% for FF-NNET). Selecting 10 bands either with the strongest PC1 coefficients or with the highest entropies was the best choice overall.

Table 9.9: Overall test and training set error rates for all classifiers using PS moments; averaged over 10 runs.

No. of bands	Test set error rate			Training set error rate		
	SVM	LDA	FF-NNET	SVM	LDA	FF-NNET
1-250	0.000	0.490	0.310	0.000	0.552	0.200
101-250	0.000	0.349	0.074	0.000	0.339	0.000
First 10 PC images	0.111	0.839	0.772	0.000	0.548	0
10 bands with strongest PC1 coefficients	0.011	0.050	0.050	0.000	0.000	0
10 bands with highest entropies	0.011	0.061	0.044	0.000	0.000	0
10 bands with lowest MI	0.017	0.294	0.411	0.000	0.217	0

Table 9.10 shows the Type 0 error rates for each tea type using all classifiers for one of the two best sets of bands, i.e. the 10 bands with highest entropy. Teas 4 and 5 were harder to classify than the other tea types. SVM is best and LDA is poorest.

Table 9.10: Average test set error rates for SVM, LDA and FF-NNET using 9 PS moments from the 10 bands with highest entropies; results are averaged over 10 runs.

Tea type	SVM	LDA	FF-NNET
Tea 1	0.000	0.067	0.000
Tea 2	0.000	0.000	0.000
Tea 3	0.000	0.000	0.000
Tea 4	0.067	0.100	0.167
Tea 5	0.000	0.200	0.100
Tea 6	0.000	0.000	0.000
Overall	0.011	0.061	0.044

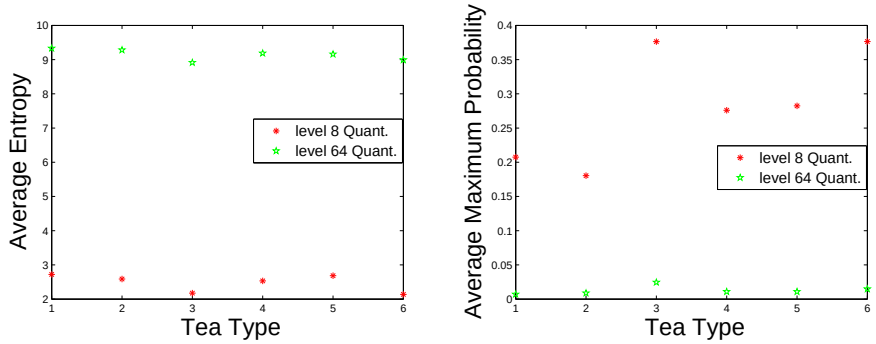
9.8 Classification using Other Features

Since the 10 bands with highest entropies provided good classification results for all different classifiers using PS moments, we investigated the performance of different features, i.e. GLCM features, wavelet-based features and wavelet-based GLCM features from this set of bands.

9.8.1 GLCM features

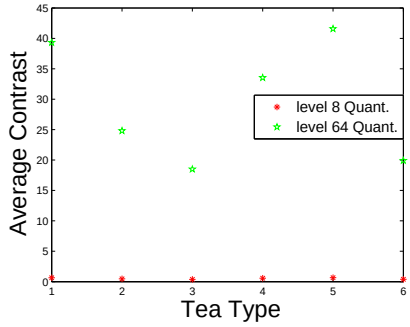
First we computed 6 GLCM features, namely maximum probability, energy, entropy, contrast, homogeneity and correlation (as in Section 3.5) for four orientations, namely 0°, 90°, 45° and 135°. We averaged the GLCM features over all orientations to obtain rotationally invariant features. Then we computed the average of these rotationally invariant features over the 10 bands, which are plotted against tea type in Figure 9.17 and shown in Table 9.11. Average entropy using quantisation level 8 is lower than that of level 64 for all tea types, although they are not very distinct for different tea types. Level 64 produces lower maximum probabilities (closer to zero) than level 8 and the differences are apparent for each tea type except Teas 4 and 5. Average contrasts for level 8 look almost identical in the plot because of the y-axis scale (the numerical values range from 0.39 to 0.66), however level 64 produced considerably different contrast for different tea

types. Tea types are easily distinguishable based on the average correlation from both quantisation levels. Average energies from level 64 are more or less identical and near zero, whereas energies from level 8 are higher and are different for different tea types. Average homogeneities for different tea types are not very distinct for either level of quantisation, but level 8 produces higher homogeneities than level 64. We used all 6 features from either quantisation for classification in Section 9.8.5.

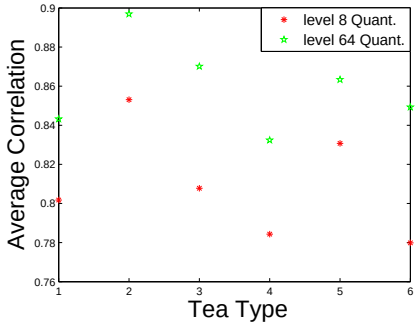


(a) Average Entropy

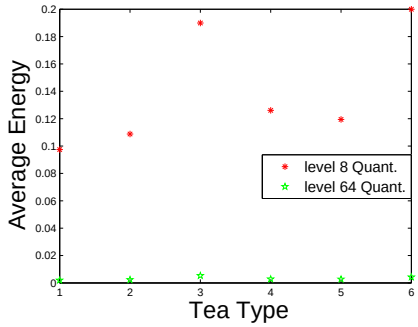
(b) Average Maximum Probability



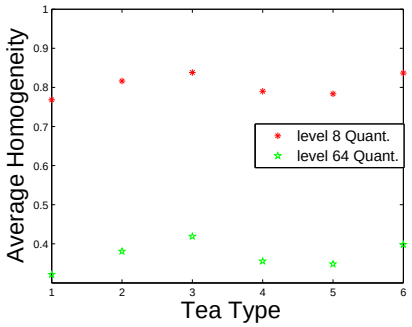
(c) Average Contrast



(d) Average Correlation



(e) Average Energy



(f) Average Homogeneity

Figure 9.17: Plots of the rotationally invariant GLCM features, averaged over 10 bands, against tea type, from the 10 bands with highest entropies.

Table 9.11: Rotationally invariant GLCM features, averaged over 10 bands, computed at quantisation levels 8 and 64.

Quantisation level 8						
Tea type	Entropy	Max. Pr.	Contrast	Correlation	Energy	Homogeneity
Tea 1	2.7205	0.2074	0.6452	0.8017	0.0974	0.7682
Tea 2	2.5879	0.1804	0.4636	0.8531	0.1088	0.8163
Tea 3	2.1749	0.3762	0.3827	0.8078	0.1899	0.8382
Tea 4	2.5327	0.2760	0.5725	0.7843	0.1261	0.7901
Tea 5	2.6855	0.2825	0.6596	0.8307	0.1195	0.7836
Tea 6	2.1375	0.3764	0.3928	0.7799	0.2000	0.8368
Quantisation level 64						
Tea type	Entropy	Max. Pr.	Contrast	Correlation	Energy	Homogeneity
Tea 1	9.3300	0.0071	39.3196	0.8432	0.0019	0.3212
Tea 2	9.2821	0.0087	24.8093	0.8969	0.0023	0.3806
Tea 3	8.9118	0.0245	18.4885	0.8701	0.0053	0.4190
Tea 4	9.1871	0.0108	33.5330	0.8324	0.0028	0.3555
Tea 5	9.1576	0.0108	41.5692	0.8633	0.0027	0.3482
Tea 6	8.9924	0.0148	19.9095	0.8493	0.0043	0.3982

9.8.2 Wavelet-based features

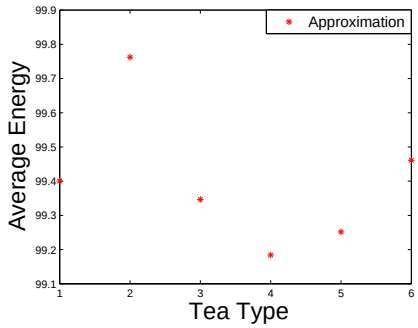
We applied a 2-D DWT on each of the 70^2 sub-images extracted from the 10 bands with highest entropies. We employed the first level of decomposition using a Daubechies wavelet with 45 vanishing moments as the mother wavelet. Among many wavelet-based features, the mean and sd were computed from the approximation sub-band as well as from each detail sub-band. Energy from the approximation sub-band was also computed. The features were averaged over all sub-images in each tea type and are shown in Figure 9.18 and Table 9.12. It is clear that the means of the detail sub-bands are not informative as they are all near-zero, hence they are not considered for use in classification. Although energy, mean and sd from the approximation sub-band and the sds from the detail sub-bands do not vary substantially over different tea types they show at least some differences, and we used all of these for classification in Section 9.8.5.

9.8.3 Wavelet-based GLCM features

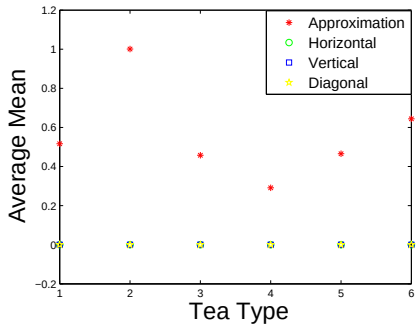
We also computed GLCM features from the wavelet sub-bands of the images rather than the original 70^2 images, as these features have produced good classification results in some applications, e.g. in Li et al. (2011). First a level 1 DWT was obtained from the 70^2 images and approximation and detail sub-bands obtained, and then the same 6 GLCM features were computed from all 4 sub-bands. Again four different orientations were considered while computing wavelet-based

Table 9.12: Average wavelet-based features from the 10 bands with highest entropies.

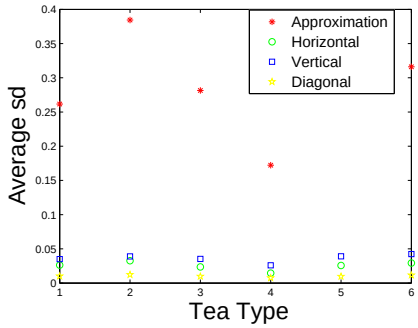
	Approximation			Horizontal		Vertical		Diagonal	
	Energy	Mean	Sd	Mean	Sd	Mean	Sd	Mean	Sd
Tea 1	99.4004	0.5164	0.2616	0.0000	0.0262	0.0001	0.0351	0.0000	0.0102
Tea 2	99.7624	1.0011	0.3844	0.0001	0.0326	0.0002	0.0390	0.0000	0.0122
Tea 3	99.3468	0.4579	0.2815	0.0000	0.0236	0.0001	0.0353	0.0000	0.0098
Tea 4	99.1843	0.2912	0.1723	0.0000	0.0146	0.0000	0.0259	0.0000	0.0075
Tea 5	99.2518	0.4653	0.2946	0.0000	0.0257	0.0000	0.0391	0.0000	0.0097
Tea 6	99.4611	0.6439	0.3161	0.0001	0.0293	0.0001	0.0424	0.0000	0.0117



(a) Average Energy



(b) Average Mean

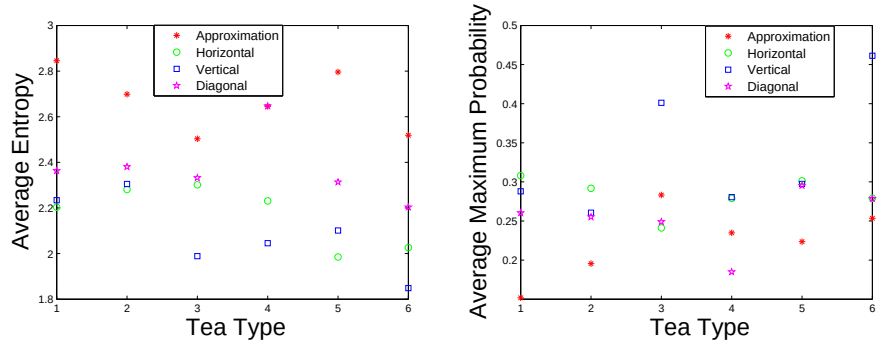


(c) Average sd

Figure 9.18: Plots of the wavelet-based features, averaged over 10 bands, against tea type from the 10 bands with highest entropies, for each tea type.

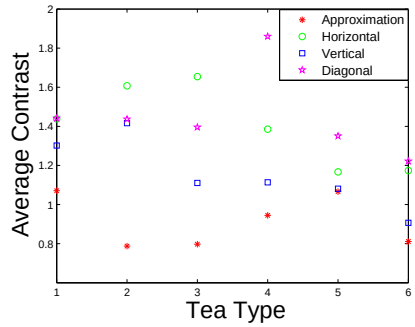
GLCM features and the results were averaged over the different orientations to obtain rotationally invariant features. Wavelet-based GLCM features were computed from the 10 bands with highest entropies with quantisation levels 8 and 64. The average features are shown in Figures 9.19 and 9.20 and Table 9.13. For both quantisation levels, the approximation sub-band produced higher entropy and correlation than any detail sub-band, whereas contrast and energy are lower than in the detail sub-bands. The energies from all sub-bands for quantisation level 64 are zero up to 2 decimal places, except from the vertical sub-band for

Tea 6, but they all vary at further decimal places, as seen in Figure 9.20(e). For both levels of quantisation, tea types look distinct based on all 6 features from all sub-bands except correlation from the detail sub-bands, though they vary slightly over tea types. These are more obvious from the figures (Figures 9.19 and 9.20) than the table (Table 9.13). However, none of the features from either quantisation level is constant over tea types, hence all features are used for classifying these tea types.

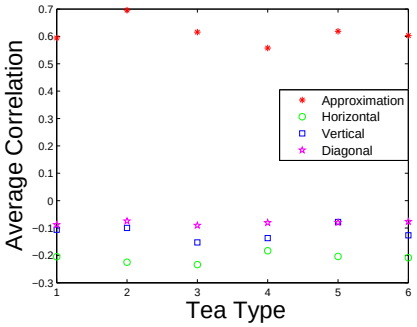


(a) Average Entropy

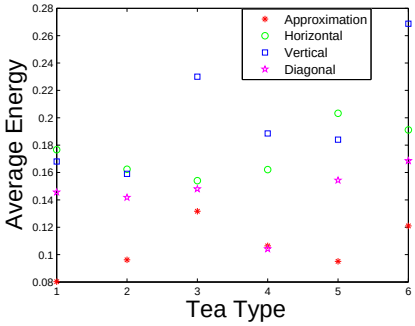
(b) Average Maximum Probability



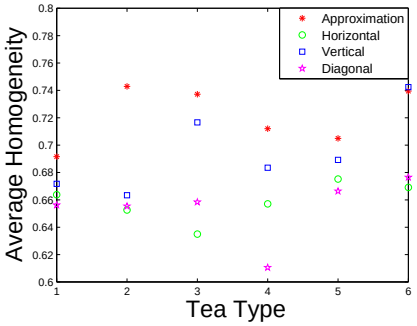
(c) Average Contrast



(d) Average Correlation

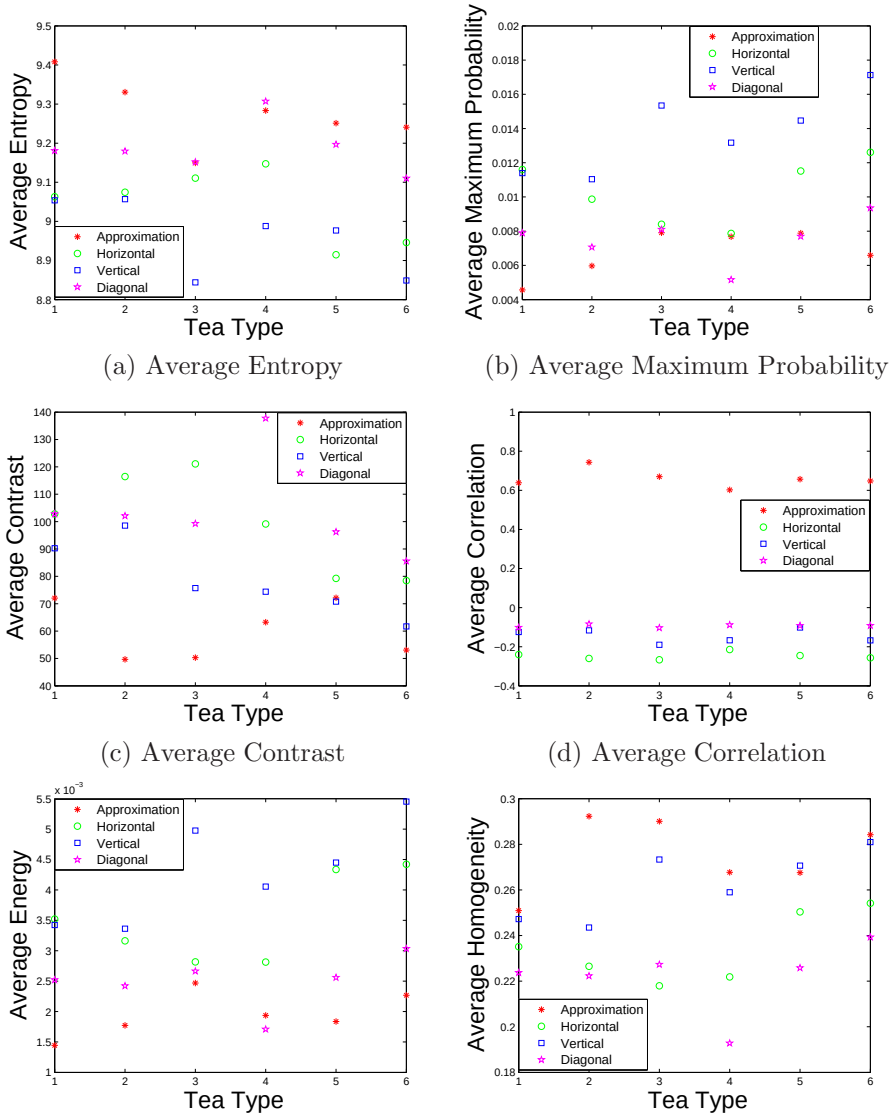


(e) Average Energy



(f) Average Homogeneity

Figure 9.19: Plots of the average wavelet-based GLCM features computed at quantisation level 8 against tea type, from the 10 bands with highest entropies.



(a) Average Entropy (b) Average Maximum Probability

(c) Average Contrast (d) Average Correlation

(e) Average Energy (f) Average Homogeneity

Figure 9.20: Plots of the average wavelet-based GLCM features computed at quantisation level 64 against tea type, from the 10 bands with highest entropies.

Table 9.13: Rotationally invariant GLCM features from the wavelet decompositions, from the 10 bands with highest entropies.

	Wavelet-based GLCM features at quantisation level 8											
	Approximation						Horizontal					
	Ent	MaxPr	Cont	Corr	Ener	Hom	Ent	MaxPr	Cont	Corr	Ener	Hom
Tea 1	2.85	0.15	1.07	0.59	0.08	0.69	2.20	0.31	1.44	-0.20	0.18	0.66
Tea 2	2.70	0.20	0.79	0.70	0.10	0.74	2.28	0.29	1.61	-0.22	0.16	0.65
Tea 3	2.50	0.28	0.80	0.62	0.13	0.74	2.30	0.24	1.65	-0.23	0.15	0.64
Tea 4	2.65	0.23	0.94	0.56	0.11	0.71	2.23	0.28	1.39	-0.18	0.16	0.66
Tea 5	2.80	0.22	1.07	0.62	0.10	0.70	1.99	0.30	1.17	-0.20	0.20	0.68
Tea 6	2.52	0.25	0.81	0.60	0.12	0.74	2.03	0.28	1.17	-0.21	0.19	0.67
	Vertical						Diagonal					
	Ent	MaxPr	Cont	Corr	Ener	Hom	Ent	MaxPr	Cont	Corr	Ener	Hom
Tea 1	2.23	0.29	1.30	-0.11	0.17	0.67	2.36	0.26	1.44	-0.09	0.15	0.66
Tea 2	2.30	0.26	1.42	-0.10	0.16	0.66	2.38	0.26	1.44	-0.07	0.14	0.66
Tea 3	1.99	0.40	1.11	-0.15	0.23	0.72	2.33	0.25	1.40	-0.09	0.15	0.66
Tea 4	2.05	0.28	1.11	-0.14	0.19	0.68	2.65	0.18	1.86	-0.08	0.10	0.61
Tea 5	2.10	0.30	1.08	-0.08	0.18	0.69	2.31	0.30	1.35	-0.08	0.15	0.67
Tea 6	1.85	0.46	0.91	-0.13	0.27	0.74	2.20	0.28	1.22	-0.08	0.17	0.68
	Wavelet-based GLCM features at quantisation level 64											
	Approximation						Horizontal					
	Ent	MaxPr	Cont	Corr	Ener	Hom	Ent	MaxPr	Cont	Corr	Ener	Hom
Tea 1	9.41	0.00	72.06	0.64	0.00	0.25	9.06	0.01	102.81	-0.24	0.00	0.24
Tea 2	9.33	0.01	49.68	0.74	0.00	0.29	9.07	0.01	116.47	-0.26	0.00	0.23
Tea 3	9.15	0.01	50.31	0.67	0.00	0.29	9.11	0.01	121.08	-0.27	0.00	0.22
Tea 4	9.28	0.01	63.24	0.60	0.00	0.27	9.15	0.01	99.12	-0.21	0.00	0.22
Tea 5	9.25	0.01	72.16	0.66	0.00	0.27	8.91	0.01	79.27	-0.25	0.00	0.25
Tea 6	9.24	0.01	53.07	0.65	0.00	0.28	8.95	0.01	78.42	-0.26	0.00	0.25
	Vertical						Diagonal					
	Ent	MaxPr	Cont	Corr	Ener	Hom	Ent	MaxPr	Cont	Corr	Ener	Hom
Tea 1	9.05	0.01	90.25	-0.12	0.00	0.25	9.18	0.01	102.77	-0.10	0.00	0.22
Tea 2	9.06	0.01	98.55	-0.12	0.00	0.24	9.18	0.01	102.10	-0.08	0.00	0.22
Tea 3	8.84	0.02	75.71	-0.19	0.00	0.27	9.15	0.01	99.26	-0.10	0.00	0.23
Tea 4	8.99	0.01	74.38	-0.17	0.00	0.26	9.31	0.01	137.73	-0.09	0.00	0.19
Tea 5	8.98	0.01	70.78	-0.10	0.00	0.27	9.20	0.01	96.26	-0.09	0.00	0.23
Tea 6	8.85	0.02	61.69	-0.17	0.01	0.28	9.11	0.01	85.58	-0.09	0.00	0.24

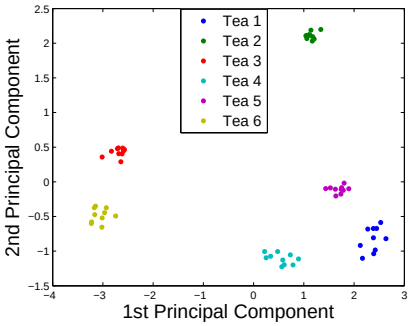
9.8.4 PCA clustering using other features

PCA with normalisation of the features was applied to the different feature sets above to observe the separation of the six Chinese teas. First we applied PCA on the 6 rotationally invariant GLCM features for both quantisations separately. For quantisation levels 8 and 64 the first two PCs explain 95.27% and 93.51% of the variation in the feature sets respectively. (These very high figures may suggest that the data has been over-simplified by quantisation.) The corresponding scatter plots of the first two PCs are shown in Figure 9.21 (a) and (b). Both sets of PCs correctly separate the six tea types without any misclassifications. Points belonging to each tea type are well separated from the other tea types in the plots.

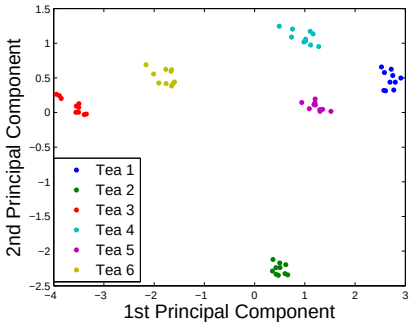
PC1 from PCA on 6 wavelet-based features expresses 88.7% of the variation in the feature set and together with PC2 it explains 98.17%. The scatter plot shows that all the tea types are also clearly distinguishable using the first two PCs (Figure 9.21 (e)).

PCA was also applied to the wavelet-based GLCM features from both quantisation levels. For level 8, the first PC explains only 34.37% of the variation in the features and the first two PCs explain 60.12%. For quantisation level 64, PC1 expresses 38.25% of the variation and the first two PCs explain 63.33%. Since the proportion of variation explained by the first two PCs is low we would not expect such good separation using them (Figure 9.21 (c) and (d)). The features from quantisation level 8 were able to separate only Teas 3, 5 and 6 but the others overlap with each other. However, the features computed at quantisation level 64 classified the tea types into different groups without any overlap, although the points within the groups are quite scattered compared to Figure 9.21 (a)–(b).

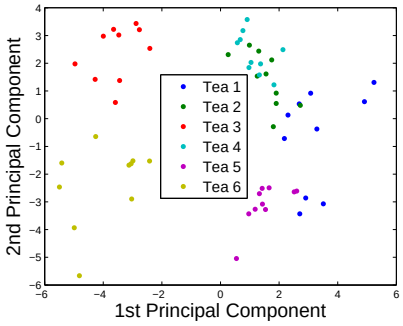
The PC loadings for the 6 GLCM features from both levels of quantisations and wavelet-based features are shown in Table 9.14. In PC1 from the GLCM features at level 8, entropy has the highest coefficient while the second strongest corresponds to energy and correlation is least strong. Correlation has the highest coefficient in PC2 whereas maximum probability is the most influential feature in PC3. At level 64, energy, correlation and contrast have the strongest coefficients in PC1, PC2 and PC3 respectively. Therefore, none of the features uniquely dominates the others. For the wavelet-based features, sd from the diagonal, vertical and approximation sub-bands has the highest coefficients in PC1, PC2 and PC3 respectively.



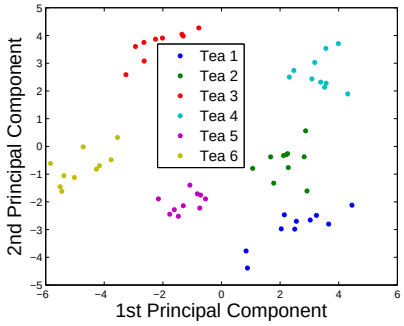
(a) PCs from quantisation level 8



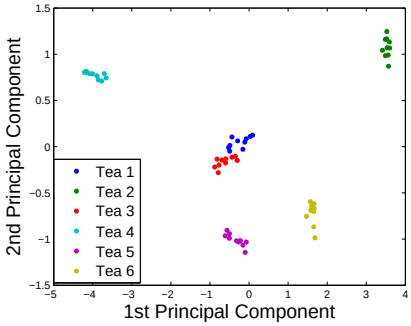
(b) PCs from quantisation level 64



(c) PCs from wavelet-based GLCM features at level 8



(d) PCs from wavelet-based GLCM features at level 64



(e) PCs from wavelet-based features

Figure 9.21: Scatter plots of PC2 versus PC1 from 6 rotationally invariant GLCM features at quantisation levels 8 and 64 ((a) and (b)), 6 rotationally invariant wavelet-based GLCM features ((c) and (d)) and 6 wavelet-based features (e) from the 10 bands with highest entropies from each type.

9.8.5 Classification using supervised classifiers

As the PCA clustering was able to distinguish the tea types 100% correctly, for both sets of GLCM features and wavelet-based features, we applied SVM, LDA and FF-NNET to all the feature sets to classify the tea images, and compared their performances.

For SVM, any kernel, i.e. a radial basis kernel, a polynomial kernel and a linear kernel, produced 100% correct classification for a single training set with

Table 9.14: Principal component scores for GLCM and wavelet-based features from the 10 bands with highest entropies for the Chinese tea images.

	Level 8			Level 64		
GLCM features	PC1	PC2	PC3	PC1	PC2	PC3
Entropy	0.4678	-0.0095	0.0945	0.4418	-0.2725	-0.2294
Max.Probability	-0.4137	-0.2949	0.6573	-0.4436	0.1652	0.4371
Contrast	0.4128	-0.3775	0.4562	0.4158	0.2327	0.7307
Correlation	0.2157	0.7819	0.5347	-0.1190	-0.8720	0.4181
Energy	-0.4623	-0.0764	0.2409	-0.4601	0.2117	0.1023
Homogeneity	-0.4234	0.3914	-0.0836	-0.4575	-0.1979	-0.1927
Wavelet features	PC1	PC2	PC3			
Energy from approximation sub-band	0.3944	0.5414	-0.2148			
Mean from approximation sub-band	0.4082	0.4229	0.2180			
Sd from approximation sub-band	0.4243	-0.0604	0.7178			
Sd from horizontal sub-band	0.4255	-0.1799	-0.0927			
Sd from vertical sub-band	0.3674	-0.6985	-0.0159			
Sd from diagonal sub-band	0.4262	-0.0641	-0.6183			

any cost of 1 or above, any value of γ between 0 and 1 and for the polynomial kernel $\eta = 1$, using the GLCM features from either quantisation level 8 or 64. For the FF-NNET on normalised features with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, any number of units between 1 and 10 produced 100% correct classification using a single training set. However we used a linear kernel with cost of 100 in SVM and 10 hidden neurons in the FF-NNET for both sets of GLCM features. The same parameter settings were used for the wavelet-based features and wavelet-based GLCM features.

For the rotationally invariant GLCM features at quantisation levels 8 and 64, all classifiers gave 100% correct classification. All classifiers also perfectly classified all tea types using the wavelet-based features and wavelet-based GLCM features. Therefore the GLCM features, wavelet-based features and the wavelet-based GLCM features are slightly more informative than the PS moments for classifying these images using any of the classifiers.

9.9 Colour Chinese Tea Images

We now consider the use of colour images rather than the hyperspectral images to explore whether the use of hyperspectral images is more advantageous (provides more useful information) than colour images for classifying these tea images. Twenty non-overlapping sub-images of size 256^2 were extracted from each of the colour images shown in Figure 9.2. Both the RGB and HSV (hue-saturation-value) colour spaces are considered for computation of PS moments. The PS moments using square, disk and rectangle SEs were computed from both sets of colour representations. The average PS moments computed from the red, green and blue planes and also from a HSV representation of the colour Chinese tea images are shown in Table 9.15. PS skewness from all SEs is zero up to 4 decimal places, therefore is not shown in the table. There is some difference between the teas.

In the red plane of the RGB colour representation, Tea 4 has the highest and tea 6 has the lowest PS mean from a square and a rectangle SEs, whereas a disk SE produced the highest mean for tea 5 and lowest for tea 6. In the green plane, the PS means for tea 3 are highest and lowest for tea 6 for all SEs, and in the blue plane the lowest PS mean corresponds to tea 2 for all SEs but the highest for tea 4 from a square and disk SEs and tea 3 for a rectangle SE. The tea types are also different in terms of PS sd and kurtosis. A rectangle SE gave higher first two PS moments than a square and a disk SE and the first two PS moments from a disk SE are generally lowest for all tea types.

The PS means and sds from the hue and saturation images are much lower for all SEs than those of the intensity images. For the hue images, Tea 2 has the highest PS mean and sd and the lowest kurtosis and Tea 6 has the lowest PS mean. For the saturation images, Tea 2 has the highest mean and sd whereas Tea 1 has the lowest. Overall the PS mean, sd and kurtosis are different for the different tea types for both sets of colour maps, hence are likely to be useful for classification.

We used both sets of PS moments in the SVM, LDA and FF-NNET for classifying the teas. There are 3 PS moments (mean, sd and kurtosis) from each SE for each of the 3 colour planes in both colour representations, giving a total of 27 PS moments to be used in any classifier. Table 9.16 shows the training set error for SVM (for a single training set) with different kernels and different values of γ and cost. For the polynomial kernel any value of η between 1 and 5 gave slightly higher error rate than $\eta = 0$. Although a polynomial kernel with $\eta = 0$ does as well as the linear kernel for the PS moments from the RGB colour planes, a

linear kernel with cost 100 was used for classification as it produced 100% correct classification for both sets of PS moments. Ten hidden units with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$, where $\boldsymbol{x}$ is the input data, were used for both the PS moments from the RGB HSV colour maps in FF-NNET (with normalisation of features), as they produced lower training set error rates (Table 9.17).

The average test set error rates for all classifiers using the PS moments from both colour representations are shown in Table 9.18. For the RGB images, the error rates for SVM, LDA and FF-NNET are 7.2%, 23.9% and 21.7%, whereas for the HSV model the error rates are 7.8%, 22.8% and 22.2% respectively, so the error rates are not very different for the RGB and HSV colour spaces. None are nearly as good as using the hyperspectral images, as would be expected since there is less information in the RGB and HVS data.

Table 9.15: Average PS moments computed from the RGB and HSV colour maps of the Chinese tea images using a square, disk and rectangular SEs.

PS moments from the red plane of the Chinese tea images									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	132.54	94.18	224.05	154.62	106.06	284.52	-0.63	-1.24	-0.30
Tea 2	139.04	81.56	225.10	147.44	85.23	268.40	-0.11	-0.34	0.47
Tea 3	184.04	114.07	292.40	163.47	101.62	276.62	-1.60	-1.67	-1.56
Tea 4	195.66	107.87	306.69	165.16	103.11	275.08	-1.52	-1.53	-1.50
Tea 5	182.10	120.33	283.24	167.41	122.17	276.35	-1.38	-1.25	-1.35
Tea 6	125.50	78.29	197.07	154.51	93.36	260.09	-0.19	-0.66	0.22
PS moments from the green plane of the Chinese tea images									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	120.10	86.67	196.69	146.48	105.00	259.93	0.12	-0.43	0.94
Tea 2	117.77	71.63	176.86	128.68	79.20	208.29	0.97	0.96	1.58
Tea 3	198.64	129.00	330.80	178.49	119.31	321.04	-1.64	-1.71	-1.57
Tea 4	197.57	122.90	325.77	168.98	106.62	303.88	-1.53	-1.54	-1.46
Tea 5	175.04	104.08	278.15	164.52	94.71	279.46	-1.39	-1.36	-1.27
Tea 6	104.61	66.78	158.02	133.79	85.64	215.74	0.96	0.59	1.82
PS moments from the blue plane of the Chinese tea images									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	144.75	103.43	245.58	156.43	107.55	288.82	-0.91	-1.42	-0.53
Tea 2	114.73	72.31	170.60	123.65	78.59	198.55	1.28	0.74	2.32
Tea 3	221.96	136.39	391.05	168.01	103.03	321.09	-1.81	-1.85	-1.84
Tea 4	232.00	150.40	386.89	177.31	118.25	317.31	-1.77	-1.82	-1.79
Tea 5	205.39	134.70	325.16	171.47	112.39	286.28	-1.73	-1.76	-1.70
Tea 6	149.27	96.17	241.14	169.64	106.36	295.95	-0.77	-1.01	-0.31
PS moments from the Hue images									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	11.28	6.01	15.12	8.30	4.78	11.83	1.31	0.79	1.08
Tea 2	50.21	28.97	69.91	46.71	27.54	65.02	-0.11	-0.28	-0.38
Tea 3	12.46	6.71	16.74	10.22	5.93	14.83	2.66	2.40	2.56
Tea 4	14.85	8.10	20.10	14.40	8.28	20.43	5.86	5.04	5.07
Tea 5	12.20	6.47	16.42	11.96	6.72	16.90	7.82	6.87	8.50
Tea 6	10.86	5.81	14.55	8.60	5.04	12.25	3.68	3.36	3.99
PS moments from the Saturation images									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	21.84	11.28	30.11	21.44	11.68	31.42	4.07	5.14	5.18
Tea 2	103.30	54.20	149.57	96.36	49.16	145.14	0.98	0.89	0.36
Tea 3	33.17	17.66	45.79	33.09	17.38	46.41	3.87	3.43	4.20
Tea 4	58.36	31.93	80.93	58.94	33.54	82.12	2.48	3.47	2.43
Tea 5	66.51	37.53	96.06	58.11	32.70	87.25	0.48	0.06	1.05
Tea 6	33.76	17.30	46.25	35.65	18.95	49.36	4.62	5.15	4.84
PS moments from the Intensity images									
Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	147.45	96.22	256.30	157.24	95.47	298.10	-1.15	-1.64	-0.88
Tea 2	139.20	77.64	225.62	145.64	86.63	265.79	-0.25	-0.16	0.32
Tea 3	206.17	141.77	342.76	164.83	112.33	292.47	-1.81	-1.82	-1.80
Tea 4	216.65	132.37	376.09	175.83	115.17	337.19	-1.68	-1.78	-1.66
Tea 5	193.63	137.14	314.43	172.36	122.17	301.46	-1.58	-1.67	-1.54
Tea 6	141.07	96.46	229.87	161.49	105.38	282.36	-0.79	-0.99	-0.53

Table 9.16: Training set error rates for different combinations of cost and parameter γ for a radial basis kernel and a polynomial kernel with $\eta = 0$ using 27 PS moments; 9 from each of the RGB colour plane and a linear kernel for RGB and HSV colour maps of the Chinese tea images, using square, disk and rectangle SEs.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.36	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
0.5	0.67	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61
0.9	0.75	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72
	Polynomial kernel with $\eta = 0$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.5	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.9	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
	Linear kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
RGB	0	0	0	0	0	0	0	0	0	0	**0**
HSV	0	0	0	0	0	0	0	0	0	0	**0**

Table 9.17: Training set error rates from a grid search approach for finding the optimum number of hidden neurons for FF-NNET with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$, using 27 PS moments from RGB and HVS colour maps of the Chinese tea images using square, disk and rectangle SEs.

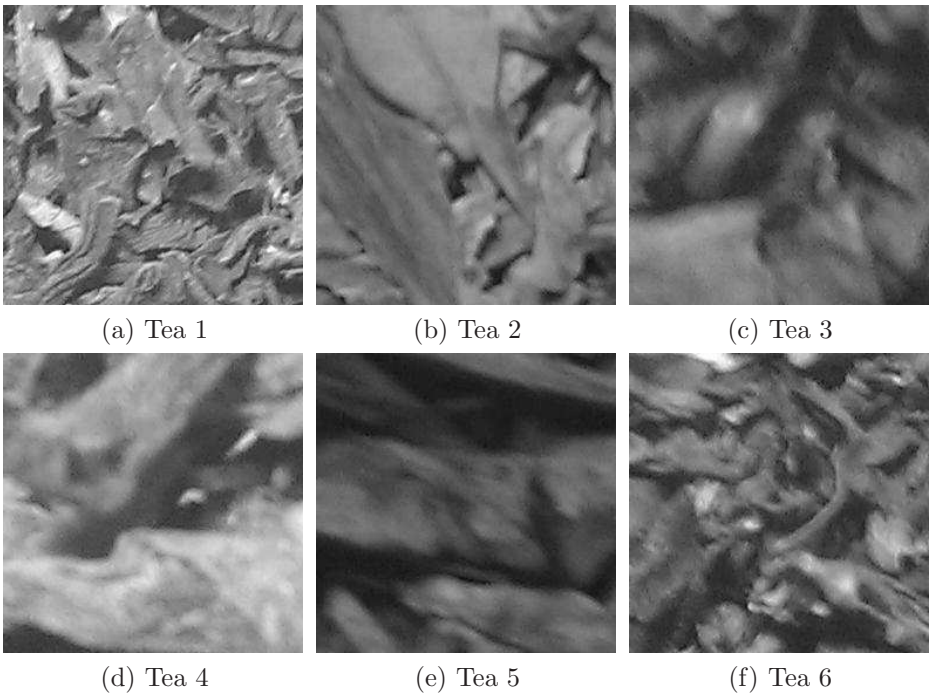
	Number of units									
	1	2	3	4	5	6	7	8	9	10
RGB	0.08	0	0.01	0	0	0	0	0	0	**0**
HSV	0.37	0.06	0	0	0	0	0	0	0	**0**

Table 9.18: Average test set error rates for SVM, LDA and FF-NNET using 27 PS moments, 9 from each colour plane of the Chinese tea images using square, disk and rectangle SEs; results are averaged over 10 runs.

Tea type	PS moments from RGB images			PS moments from HSV images		
	SVM	LDA	FF-NNET	SVM	LDA	FF-NNET
Tea 1	0.017	0.133	0.150	0.000	0.100	0.233
Tea 2	0.000	0.083	0.083	0.000	0.050	0.000
Tea 3	0.083	0.267	0.183	0.033	0.183	0.167
Tea 4	0.167	0.400	0.467	0.267	0.417	0.300
Tea 5	0.133	0.467	0.317	0.167	0.367	0.283
Tea 6	0.033	0.083	0.100	0.000	0.250	0.350
Overall	0.072	0.239	0.217	0.078	0.228	0.222

9.10 Grey Scale Chinese Tea Images

The grey scale versions of the RGB images were also considered for classification of the images instead of the RGB or HSV colour spaces. The RGB images (shown in Figure 9.2) were converted to grey scale images and twenty non-overlapping sub-images of size 256^2 were extracted from each image. One such sub-image from each tea type is shown in Figure 9.22.



(a) Tea 1 (b) Tea 2 (c) Tea 3

(d) Tea 4 (e) Tea 5 (f) Tea 6

Figure 9.22: Grey scale images of size 256×256 from 6 types of Chinese tea.

The PS moments from the grey scale Chinese tea images using a square, disk and a rectangular SEs were computed and averaged over all sub-images. Average PS moments are shown in Table 9.19. Tea 3 has the highest mean and sd using any SE, whereas Tea 6 has the lowest mean and sd. Skewness from all SEs is zero up to 5 decimal places, so they are not shown in the table and therefore again we use only the mean, sd and kurtosis from all SEs for classification.

We used SVM, LDA and FF-NNET (on normalised features) to classify the grey scale images. For all classifiers 70% randomly chosen sub-images from each tea type were used as the training set and the remaining are used for testing. The process was repeated 10 times and the results are averaged over 10 runs.

For SVM, the training set error rates from a single training set for different values of γ and cost for a radial basis kernel and a polynomial kernel are shown in Table 9.20. The radial basis kernel produced the lowest error rate of 8% for $\gamma = 0.1$ and cost of 10, whereas the lowest error rate of 5% from a polynomial

Table 9.19: Average PS moments computed from the grey scale Chinese tea images using a square, disk and rectangular SEs.

Tea type	PS mean			PS sd			PS kurtosis		
SEs	Square	Disk	Rect.	Square	Disk	Rect.	Square	Disk	Rect.
Tea 1	132.77	92.99	219.49	157.05	104.55	279.30	-0.38	-1.11	0.06
Tea 2	125.37	82.29	192.96	134.29	86.32	227.60	0.40	-0.23	1.04
Tea 3	206.04	129.34	338.05	178.88	111.81	314.60	-1.71	-1.78	-1.69
Tea 4	200.41	120.01	325.21	163.71	107.13	287.63	-1.61	-1.66	-1.59
Tea 5	189.04	118.83	310.95	172.56	115.56	306.85	-1.50	-1.40	-1.42
Tea 6	117.29	78.05	179.16	144.56	92.36	236.58	0.12	-0.53	0.81

kernel with $\eta = 1$ can be obtained for $\gamma = 0.1$ and any cost between 1 and 100. The table also shows the error rate for a linear kernel, which produced 100% correct classification for any cost between 1 and 100. In FF-NNET, 10 hidden units produced the lowest error rate (Table 9.21), hence 10 units are used.

Table 9.20: Training set error rates for different combinations of cost and parameter γ for a radial basis kernel and a polynomial kernel with $\eta = 1$ and for a linear kernel with different cost using 9 PS moments from the grey scale Chinese tea images using square, disk and rectangle SEs.

	Radial basis kernel										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.27	0.08	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
0.5	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22	0.22
0.9	0.42	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40
	Polynomial kernel with $\eta = 1$										
	Cost										
γ	1	10	20	30	40	50	60	70	80	90	100
0.1	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
0.5	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12
0.9	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
	Linear kernel										
	Cost										
Cost	1	10	20	30	40	50	60	70	80	90	100
	0	0	0	0	0	0	0	0	0	0	**0**

Table 9.22 shows the average test set error rates for the different classifiers. SVM produced only a 1.4% error rate, while LDA and FF-NNET produced 51.6% and 27.5% error rates respectively, which are much poorer. LDA performed much worse with these images than with the colour ones, FF-NNET was slightly worse,

Table 9.21: Training set error rate from a grid search approach for finding the optimum number of hidden neurons for FF-NNET with decay of 10^{-4} and rang of $1/\max(|\boldsymbol{x}|)$ using 9 PS moments from the grey scale Chinese tea images using square, disk and rectangle SEs.

	Number of units									
	1	2	3	4	5	6	7	8	9	10
Error rate	0.32	0.21	0.02	0	0	0	0	0	0	**0**

Table 9.22: Average test set error rates for SVM, LDA and FF-NNET using 9 PS moments from grey scale Chinese tea images; results are averaged over 10 runs.

Tea type	SVM	LDA	FF-NNET
Tea 1	0.000	0.700	0.283
Tea 2	0.033	0.400	0.133
Tea 3	0.000	0.446	0.217
Tea 4	0.000	0.667	0.333
Tea 5	0.000	0.533	0.333
Tea 6	0.050	0.350	0.350
Overall	0.014	0.516	0.275

but SVM was (surprisingly) better. All are worse than with the hyperspectral images.

9.11 Conclusion

Although all available spectral bands of hyperspectral images are used in many classification problems, selecting the optimum set of bands is important. We have discussed different approaches for selecting the optimum set of bands, among which choosing bands with the highest entropy and the bands with the strongest PC1 coefficients appeared to be the best criteria. A set of bands either with maximum entropies or with the strongest PC1 coefficients produced the lowest error rates for all classifiers for these tea images. However it is not guaranteed that these are the best criteria for selecting appropriate bands for every application, as these may depend on the images of interest. The performance of all classifiers using PS moments from the hyperspectral images is satisfactory, although SVM outperforms LDA and FF-NNET.

The rotationally invariant GLCM features, wavelet-based features and wavelet-based average GLCM features provide very useful information for classifying these tea images and are all better than PS moments, as these features gave 100% cor-

rect classification using any classifier with appropriate parameter settings.

The usefulness of two different colour representations and grey scale images, compared to the hyperspectral images, was also examined. SVM, LDA and FF-NNET showed similar efficiency using the PS moments from both colour maps. Although SVM worked slightly better with the PS moments from the grey scale images, the performance of LDA was much better with the PS moments from either the RGB or HSV images. The FF-NNET had slightly better results for the RGB and HSV images than for the grey scale images.

Nonetheless, use of the optimum set of bands from the hyperspectral images is much more advantageous (as the error rates for SVM, LDA and FF-NNET were only 1.1%, 6.1% and 4.4% respectively) than either colour plane representation or the grey scale version for any classifier.

Overall conclusions of the thesis are drawn in Chapter 10.1.

Chapter 10

Conclusions

In this chapter a summary of the important findings from Chapters 4–9 is presented in Section 10.1 and future potential research directions are discussed in Section 10.2.

10.1 Key Findings

In this thesis, the use of regression modelling based on PS moments as image texture features from multiple SEs has been explored as a means of classifying texture images. GLCM features and wavelet-based features were also used for comparison. Other approaches to classification, i.e. SVM, LDA and FF-NNET, were also used for comparison.

We considered several different sets of texture images. The PS moments from a disk SE provided more useful features than a square or a line SE at any of the four angles (0°, 90°, 45° and 135°) for the synthetic images as well as for the real images of corrosion and Indian black tea granules. However computing PS moments using a disk SE is much more computationally expensive (at least 15 times more) than a square SE and horizontal or vertical line SEs for all sets of synthetic and real images. A line SE at 45° or 135° also required more time than the above SEs for computing PS moments but less than one fifth of the time needed for a disk SE.

Both the foreground and background PS mean and sd of the synthetic images containing pyramids or ellipses showed a clear relationship with evolution time of the texture but PS skewness or kurtosis did not. However, due to the high variability of the PS means and sds across the images at a specific point in time, none of the classifiers except SVM produced good classification results. The regression model was built using average PS moments (central moments) and

the prediction was computed from each image separately, which also affected the classification results. However, building a regression model using the PS moments from every single image did not improve the results. Also the use of the PS median instead of the PS mean in the model building was not found to be beneficial. We considered as well the use of raw moments instead of the central moments for the synthetic images, but this was not advantageous and hence the results are not included in the thesis.

The training times for the SVM and LDA were very rapid (a few seconds per time state) whereas the FF-NNET required a few minutes per time state for the synthetic images. However the total time needed for FF-NNET was not significant for any of the real images used here, as they consist of either 10 time points or 8 or 6 classes.

For classifying the corrosion images, the new regression-based classification approach worked better than FF-NNET and almost as well as LDA using 8 foreground PS moments, and was more robust using fewer features. It outperformed the previously published results on these images in McKenzie et al. (2003), Gray et al. (2005) and Gray et al. (2006) where the lowest error rate is about 5 times higher than our highest error rate. Performance might be improved further by clipping any predictions to the range of times observed in the training image set. Our methodology again performed better for classifying texture images of Indian black tea granules according to granule size, compared to the published work on these images in Borah et al. (2007). The PS moments produced very good results, with the highest error rate (8.1%) being less than half of the error rate (20%) obtained in Borah et al. (2007), who used wavelet-based features coupled with LVQ and MLP. For these images wavelet-based features do not work well. A key step for getting successful results in both classification problems was the use of a disk SE of increasing radius in the pre-processing (top-hat or bottom-hat transformation) of the images, since the use of a fixed size disk in the hat transform led to high error rates for all classifiers.

Comparing the performance of different texture features for the synthetic images, neither PS moments nor GLCM features were better overall than the other, as in some cases the former worked better and in other cases the opposite was observed. However for the real images of corrosion and tea granules, the shape-based PS moments were much better than the GLCM features for classifying those images. Furthermore, for the images of tea granules, PS moments provided very good classification results compared to the wavelet-based features.

For the SVM, the simple linear kernel was the best kernel for our images

as it gave a lower error rate for all sets of features except for the PS moments from the corrosion images, for which the polynomial kernel was best. The widely used radial basis kernel did not produce the lowest error rate for any of the sets of features from any of the sets of images used here. In general SVMs were much superior for classification compared to any of the other classifiers used here, and were extremely robust to the choice of texture features. However, SVMs (and also FF-NNET) require proper training to give optimum results, as an inappropriate choice of kernel function and its parameters can lead to a large error rate (Section 5.5). If correctly trained, the state-of-the-art classifier SVM can provide 100% or near 100% correct classification rate in many classification problems, as for example in Li et al. (2011), Chen et al. (2009a) and Chen et al. (2006).

Several existing techniques for selecting informative bands from hyperspectral images were investigated here for use in texture classification of images of Chinese teas. We considered the first 10 PC images and 10 spectral bands with strongest coefficients in the first PC, from PCA, 10 bands with highest entropies and 10 bands with lowest MIs. PCA to select bands with strongest PC1 coefficients and entropy-based methods were equally effective here for choosing informative spectral bands. Using the bands with the strongest coefficients in the first PC or with the highest entropy provided very good classification results for any of SVM, LDA and FF-NNET. SVM was equally effective with a 1.1% error rate for classifying these images using PS moments computed from either set of bands. LDA and FF-NNET produced equal error rates of 5% using the PS moments from the 10 bands with the strongest coefficients in the first PC, but LDA produced a slightly higher error rate of 6.1% than that of FF-NNET (4.4%) for the PS moments from the bands with highest entropies.

The other sets of features, i.e. rotationally invariant GLCM-based features at quantisation levels 8 and 64, wavelet-based features and wavelet-based average GLCM features at quantisation levels 8 and 64 from the 10 bands with highest entropies were very robust (100% correct classification) for classifying these hyperspectral tea images using any of the three classifiers considered. Even the choice of different kernels and the cost for SVM and the different parameter values (including number of hidden units) for FF-NNET were not crucial.

Although the PS moments were more effective for classifying the images of corrosion and Indian black teas than the GLCM features, and wavelet-based features showed worse performance for the Indian tea images, for the Chinese tea images the intensity-based GLCM features gave 100% accuracy. For the Chinese

tea images, the GLCM features were averaged over four different orientations to produce the results in Section 9.8, but the features from any of the orientations separately produced the same results.

The PS moments from the RGB and HSV colour representations of the images provided similar information, as SVM and LDA produced similar results for both sets of PS moments, though the HSV colour representation was slightly better in FF-NNET. Surprisingly, the grey scale version of the images was more informative than either of the colour representations, but only for SVM.

For all sets of features from the Indian black tea images as well as from the hyperspectral images of Chinese tea, the results of any of the supervised classifiers reflected the degree of class separation expected from the plots of PC2 against PC1, from PCA.

10.2 Further Work

For further work, the use of 3-D (grey scale) SEs rather than flat (binary) 2-D SEs could be examined for the grey scale image morphology, as the 3-D SEs may extract more useful information from the images and hence lead to more accurate classification. In particular it could potentially overcome the need for hat transformation or other suitable pre-processing to reduce intensity variations within the image texture primitives. However the use of 3-D SEs would be very expensive computationally.

Also the use of higher order features, instead of the first and second order features used here, may be expected to give better classification results, hence it would be interesting to investigate the performance of these for texture classification. The autocorrelation function of an image and run length matrix-based features (described in Section 3.5) can be regarded as higher order texture features. Some of the more recent feature extraction techniques, e.g. the local binary pattern (Ojala et al. (1996)) and coordinated cluster representation (Kurmyshev and Cervantes (1996)) could also be applied. Combining different features and using variable selection procedures could also be investigated.

Combining different classifiers may provide better classification results, at least for the synthetic images, since none of the classifiers except SVM works well either with the PS moments or GLCM features from the synthetic images. Combined classifiers have achieved high classification accuracy in many applications, for example in Prasad et al. (2010).

A more systematic study of when different image features do well would be

interesting, although for the SVM the choice of features does not greatly affect the results, at least for the images used here. We used GLCM features only at 135° for the synthetic images as well as for the real corrosion and tea granule images, since the features from other orientations produced very similar time or class trends, however averaging them over different orientations, as we have done later in Section 9.7, may lead to better classification results. In the wavelet decomposition we used the Daubechies wavelet with 45 vanishing moments, as this is a widely used wavelet basis. Different wavelet bases such as Haar wavelets, or using a different number of vanishing moments with the Daubechies wavelet, may alter the results, so it would be interesting to investigate this also.

For classifying the hyperspectral images of Chinese tea, we a considered selection of different sets of bands using several different approaches, and extracted different features from these sets of bands for classification. In future work spectral signature could also be used as a feature, as it is used in many applications of hyperspectral image classification, e.g. in Camps-Valls et al. (2007).

APPENDICES

Appendix I: Matlab code

The Matlab code to generate the synthetic texture images of pyramids and ellipses discussed in Section 4.2 is given here.

Generation of synthetic images of pyramids of different widths.

```
%parameters of the program

time =100;                  %number of iterations
start= 40;                  %control the colour of new object
objectcode = 50;            %initial mean object code
imsize=256;                 %size of the image
pr=0.99;                    %probability of adding a new object
updatepr = 0.2;             %probability of updating an existing object

%parameters to control the colour of the objects
updateinc = 10;             %determine increment of pixel intensities
grey = 0.25;                %scaling factor to compute the greylevel
                            %at the edges of the objects

%parameters to control the size of the object
initstep = 1;               %to create a random stepsize
stepinc = 1;                %to create a random stepsize
stepsize=zeros(time,1);     %store stepsize of each object
phi= 45;                    %control orientation of the objects

%arguments of the program
nobjects=0;                 %initial number of object
im=zeros(imsize, imsize);   %initial 2D blank image
%figure, imshow(im);
output=zeros(imsize,imsize,1,time);  %4D image to store images.
ex_num= zeros(time,1);      %vector to store iterations when new objects added
ex_numc= zeros(time,2);     %matrix to store centre coordinates of objects
size = zeros(time,1);       %vector to store the current size of each object
```

```
greylevel = zeros(time,1);  %store current greylevel of the central section

%body of the programm
for t = 1:time
   output(:,:,:,t+1) = output(:,:,:, t);
                              %subsequent stages are built on the previous stage
if t==1;                      %force to add the first object at the
    t;                        %beginning of the evolution
   cx=round(unifrnd(1,imsize,1,2)); %set random location within image
   newobjectcode = round(start + 2*(objectcode-start)*unifrnd(0,1));
                    %generate objectcode for the new object
   im(cx(1),cx(2)) = newobjectcode;
                    %assign the objectcode at the centre of the new object.
   increment = round(updateinc + unifrnd(0,1)*updateinc);
    for i = max(cx(1)-1,1): min(cx(1)+1,imsize);
        for j =  max(cx(2)-1,1): min(cx(2)+1,imsize);
            im(i, j) = min(im(i,j) + increment, 255);
        end; % loop for j
    end% loop for i
else
  if unifrnd(0,1)  <= pr        %condition of adding a new object
      cx=round(unifrnd(1,imsize,1,2)); %set random location
      newobjectcode = round(start + 2*(objectcode-start)*unifrnd(0,1));
                              %generate the greylevel of the object

      if im(cx(1),cx(1))==0
      %check to see if im matrix is empty where a new object appears
disp('Adding a new object at iteration');
                              %adding a new point at this time
               t;
    disp('Centre is at location')
               cx;
       im(cx(1),cx(2)) = newobjectcode;
                              %fill the centre with the newobjectcode
       nobjects= nobjects+1;
       greylevel(nobjects)=newobjectcode;
       ex_num(nobjects)=t;      %stores iteration number when new object added
       ex_numc(nobjects,:)=cx;  %store (x,y) location of new object
       nobjects;
      end %end for if im(cx(1),cx(1))
    end  %end for unifrnd(0,1)<= pr
  end  %end if t==1
                       %now revisit existing objects for possible update
  for k = 1: nobjects-1
   if  unifrnd(0,1) <= updatepr %updating the object using updatepr
```

```
                  t;          %at which iteration update is taking place
        nobjects;             %how many objects have been updated
  disp('Updating object');    %updating existing object
                  k;          %display which object is updating
   increment = round(updateinc + unifrnd(0,1)*updateinc);
    i1 = max(ex_numc(k,1) - size(k),1);
    i2 = min(ex_numc(k,1) + size(k), imsize);
    j1 = max(ex_numc(k,2) - size(k),1);
    j2 = min(ex_numc(k,2) + size(k), imsize);
     for i = i1: i2;
        for j = j1: j2;
ir = ex_numc(k,1)+(i-ex_numc(k,1))*round(cosd(phi))-(j-ex_numc(k,2))*
                                                  round(sind(phi));
jr = ex_numc(k,2)+(i-ex_numc(k,1))*round(sind(phi))+(j-ex_numc(k,2))*
                                                  round(cosd(phi));
    %computes the rotated x and y coordinates
    %fill the area with the minimum of newobjectcode + increment and 255.
        for inew = max(min(ir),1):min(max(ir),imsize);
          for jnew = max(min(jr),1):min(max(jr),imsize);
        im(inew, jnew) = min(im(inew, jnew) + increment, 255);
          end %end for jnew
        end  %end for inew
      end %end for j
    end %end for i
greylevel(k) = min(round(im(i,j)*grey),255);     %store old greylevel
stepsize(k)=initstep + round(stepinc*unifrnd(0,1));
                       %increase the size of the object
  size(k)=size(k)+stepsize(k);
for i=max(ex_numc(k,1)-size(k),1):min(ex_numc(k,1)+size(k), imsize);
  for j=max(ex_numc(k,2)-size(k),1):min(ex_numc(k,2)+size(k), imsize);
ir=ex_numc(k,1)+(i-ex_numc(k,1))*round(cosd(phi))-(j-ex_numc(k,2))*
                                                  round(sind(phi));
jr=ex_numc(k,2)+(i-ex_numc(k,1))*round(sind(phi))+(j-ex_numc(k,2))*
                                                  round(cosd(phi));
         %again computes rotated x and y coordinates
        for inew = max(min(ir),1):min(max(ir),imsize);
          for jnew = max(min(jr),1):min(max(jr),imsize);
        im(inew, jnew) = max(im(inew,jnew), greylevel(k));
          end %end for jnew
        end %end for inew
      end % end for j
     end % end for i
   end% if  unifrnd(0,1) <= updatepr
  end; %end for objects loop
 output(:,:,:,t) = im(:,:);%store im matrix at output matrix
```

```
 %figure,imshow(im2uint8(im(:,:)+1, 'indexed'));
 %display image at each evolving time
 %im2uint8 converts the intensity image to uint8 for the purpose of
 % display 1 is added as some of the pixel values are zero.
end; %end for time loop

%the following code is used to display different attributes and
%  evolution time of the objects
disp('Total number of objects added is');
nobjects;
disp('at iterations');
ex_num(1:nobjects,:);
disp('and at locations');
ex_numc(1:nobjects,:)
disp('current major axis of each object');
a(1:nobjects,:) ;
disp('current minor axis of each object');
b(1:nobjects,:);
disp('ratio of each object');
ratio(1:nobjects,:);
disp('orientation of each object');
phi(1:nobjects,:);
disp('stepsize of each object');
stepsize(1:nobjects,:);
```

```
%parameters of the program
time = 100;                    %number of iterations
start= 10;                     %control the colour of new object
objectcode = 15;               %initial mean object code
imsize=256;                    %size of the image
updateinc=10;                  %determine increment of pixel intensities
pr=0.99;                       %probability of adding a new object
updatepr = 0.3;                %probability of updating an existing object

%parameters to control shapes of the ellipses
rmin=0.3;                      %minimum of the ratio
rmax=.9;                       %maximum of the ratio
mu= 0.666;                     %mean of the Gaussian random number
sigma=.3;                      %sd of the Gaussian random number
%by keeping sigma at 0.3 ratios are more likely to fall between .4 to .9.
a = zeros(imsize,1);           %to store the major axis
b = zeros(imsize,1);           %to store the minor axis
ratio = zeros(time,1);         %to store ratio of major and minor axis
                               %of each object

%parameters to control how fast objects grow.
initstep = 2;                  %to control stepsize
stepsize = ones(time,1);       %to store stepsize of each object

%parameters to control orientation of the objects
phimin=125;                    %minimum orientation
phimax=135;                    %maximum orientation
phi = zeros(time,1);           %store orientation of each objects

%arguments of the program
nobjects=0;                    %initial number of object
im=zeros(imsize, imsize);      %initial 2D blank image

%figure, imshow(im);
output = zeros(imsize,imsize,1,time); %4D image to store images.
ex_num = zeros(time,1);     %store iteration number when a new point is added
ex_numc = zeros(time,2);    %matrix to store centre coordinates of objects
greylevel = zeros(time,1);  %store current greylevel of the central section

%body of the programm
for t = 1:time
    output(:,:,:,t+1) = output(:,:,:, t);
    if t==1;
   cx=round(unifrnd(1,imsize,1,2));   %set random location
```

```
   newobjectcode = round(start + 2*(objectcode-start)*unifrnd(0,1));
   im(cx(1),cx(2)) = newobjectcode;
   increment = round(updateinc + unifrnd(0,1)*updateinc);
    for i = max(cx(1)-1,1): min(cx(1)+1,imsize);
        for j =  max(cx(2)-1,1): min(cx(2)+1,imsize);
            im(i, j) = min(im(i,j) + increment, 255);
        end; % loop for j
    end% loop for i
else
  if unifrnd(0,1)  <= pr         %condition of adding a new object
      cx=round(unifrnd(1,imsize,1,2));  % set random location
      %generate the greylevel of the object
      newobjectcode = round(start + 2*(objectcode-start)*unifrnd(0,1));

      if im(cx(1),cx(2))==0 % adding a new point at this time
          disp('Adding a new object at iteration');
                t
         disp('Centre is at location')
                cx
       nobjects= nobjects+1;
       greylevel(nobjects)=newobjectcode;
       ex_num(nobjects)=t;    %stores iteration number when new object added
       ex_numc(nobjects,:)=cx; %store (x,y) location of new object
       nobjects;
      end
  end% end for x <= pr
    %now revisit existing objects for possible update
  for k = 1: nobjects-1
    if unifrnd(0,1) <= updatepr    %updating the object using updatepr
        t                          %at which iteration update is taking place
    disp('Updating object');       %updating existing object
        k                          %display which object is updating
     %make the object size larger than the current size
   if stepsize(k)==1;
     stepsize(k) =  stepsize(k)*initstep*unifrnd(0,1);
   end;
      a(k)= a(k) + stepsize(k);
 %to keep the ratio same for a particular object, that is ratio will
 %be generated once for each object.
    if ratio(k)==0;
  r=  normrnd(mu,sigma);      %generate a random number from N(mu,sigma^2)
     if r>rmax;
         ratio(k)=rmax;
     elseif r<rmin;
        ratio(k)=rmin;
```

```
     else ratio(k) =r;
     end %end for if ratio(k)
    end;%end for if stepsize(k)
     b(k) = a(k)*ratio(k);        %the major axis
      psi=0:5:359;                %vectors of angles in degrees
 %to keep the rotation angle same for a particular object, that is
 %rotation angle will be generated once for each object
    if phi(k) == 0;
        phi(k)=phimin + (phimax - phimin)*unifrnd(0,1);
    end;
%vectors of rotated x and y coordinates using rotated angle phi(k)
if a(k)>1.5
xnew = ex_numc(k,1) + round(a(k)*cosd(psi)*cosd(phi(k))-b(k)*
sind(psi)*sind(phi(k)));
ynew = ex_numc(k,2) + round(a(k)*cosd(psi)*sind(phi(k))+b(k)*
sind(psi)*cosd(phi(k)));
    for i = max(min(xnew),1) : min(max(xnew),imsize);
        for j = max(min(ynew),1) : min(max(ynew),imsize);
      icos = (i-ex_numc(k,1))*cosd(phi(k));
      jcos = (j-ex_numc(k,2))*cosd(phi(k));
      isin = (i-ex_numc(k,1))*sind(phi(k));
      jsin = (j-ex_numc(k,2))*sind(phi(k));
if ((icos+jsin).^2/b(k).^2 + (isin-jcos).^2/a(k).^2<=1),
                                                %filling the ellipse
      im(i, j) = min(im(i,j) + newobjectcode,255); end;
           end;% end for i
         end;% end for j
       end;% end for if
     end%if  unifrnd(0,1) <= updatepr
   end; %end for objects loop
  output(:,:,:,t) = im(:,:);%store im matrix at output matrix
end;%end for time loop
 figure, imshow(im2uint8(output(:,:,:,100)+1, 'indexed'));
```

Code for computing granulometric moments using different SEs. The following code was used to compute the PS moments from the synthetic images of pyramids and ellipses and is also applicable for the real images (corrosion and tea images).

```
%parameters of the program

imsize=256;                    %size of the image
time=100;                      %number of evolution steps
Sp=100;                        %number of simulations

%creates space for storing different quantities
pmg=zeros(Sp*time,imsize); %to store remaining image volume after each opening
spmg=zeros(Sp*time,imsize);  %to store normalised side distribution
moments=zeros(Sp*time,4);    %to store the first four moments

%reading image and embedding it with 0s to avoid edge effect on opening
for p=1:Sp
    filename= strcat('Name of the image file with extension');
                           %creates file name
 I=imread(filename,'frames','all');
                           %read sequence of images and assigned it to I
 s = size(I);              %provides dimension of I
n1=s(1);                   %first dimension of I
n2=s(1)+10;
for t= 1:time
  re0 = I(:,:,:,t);        %assign t$^{th}$ layer of I as re0
   if nnz(re0)==0;
     moments1(time*p-(time-t),:)=0;
                           %if re0 is a matrix of 0s, moments are 0
   else                    %if it is a matrix of positive entries
   I1=zeros(n2);           %embedding with zeros by inserting it into I1
     k=(n2-n1)/2;          %(I1 is larger than I(:,:,:,t))
    for i= (1+k):(n1+k)
       for j=(1+k):(n1+k)
        I1(i,j)=re0(i-k,j-k);
       end;
    end;

%openings by increasing the size of the SE until the image become flat.
  n=sum(I3(:));            %total no. of pixels in the image
  j=1;                     %initial width/radius of the SE
  while(sum(I3(:))~=0)     %checks for positivity of I3
     res = imopen(I3,SE);  %SE is either square, disk, or a line SE.
  pmg(time*p-(time-t),j)=sum(res(:));
                           %stores sum of the remaining pixel
                           % intensities after each opening
```

```
 I3=res;                    %resets the image
 j=j+1;                     %increases the width/radius by 1
 end;

figure,plot(pmg(time*p-(time-t),:), 'g - *'),grid on;
axis([1 j 0 n+50])          %plots the remaining pixel intensities
set(gca, 'xtick',1 : j);    %after each opening
xlabel('Width/Radius of the SE');
ylabel('Volume of the opened image');

%normalised size distribution as the difference in volume between
%two successive openings, n is the original volume of the image.
spmg(time*p-(time-t),:)=1-pmg(time*p-(time-t),:)/n;
figure,stairs(spmg1(time*p-(time-t),:), 'r - *'),grid on;
axis([1 j 0 1])             %plots the normalised size distribution
set(gca, 'xtick',1 : j);
 title('Size distribution')
 xlabel('Width/Radius of the SE)');
ylabel('Area removed by opening');

%computes ps as the derivatives of the normalised size distribution
ppmg = diff(spmg(time*p-(time-t),:));
figure,bar(ppmg,.5);                %plots pattern spectrum
    axis([0 j 0 0.3])
    set(gca, 'xtick',0:j);
   title('Pattern spectrum')
 xlabel('Width/Radius of the SE');
 ylabel('Change in size distribution');

 %computes moments of the pattern spectrum
 y = 1:max(size(ppmg));
 x=2*y-1;                           %define the width/radius of the SEs
 mean = x*ppmg';                    %mean of the PS
 variance = x.^2*ppmg' - (mean).^2; %variance of the PS
 std=sqrt(variance);                %sd of the SD
 if variance == 0;                 %if sd is 0 skewness and kurtosis is 0
     skewness = 0;
     kurtosis = 0;
  else
 a1 = (x-mean).^3*ppmg';              %3rd central moment about mean
 a2 = (x-mean).^4*ppmg';              %4th central moment about mean
 skewness = a1/(variance).^3/2;       %skewness of the PS
 kurtosis = a2/(variance).^2-3;       %kurtosis of the PS
                                %computes 'excess kurtosis' by subtracting
                                %3, which is 0 for normal distribution
```

```
   end;%end of if var
      moments(time*p-(time-t),:) = [mean,std,skewness,kurtosis];
   end;%end of if nnz
  end;% end of time loop
end;%end of simulation loop
```

Appendix II(a): SVM in R

SVMs were implemented in R using the package *e1071*. This provides a training function *svm* with standard and formula interfaces and a predict method using *predict* function. It also provides a method for visualising data, the support vectors, and the decision boundaries. The *svm* function has different parameters which can be tuned for more robust results. Among them 'scale', 'kernel', 'type', 'cross', 'cost', 'fitted' are important to consider.

Explanation of the parameters is given below (from the R help file): *Scale* is a logical vector indicating whether or not to normalise the data, and we chose normalising the data (the default). That means the data are scaled internally to zero mean and unit variance. The centred and scaled values are returned and used for later predictions.

Parameter *type* specifies whether *svm* is used as a classification machine or as a regression machine. Depending on whether the dependent variable is a factor or not, the default setting for type is *C-classification* or *ε-regression*, respectively, but may be overwritten by setting an explicit value. C-classification was used as the dependent variables are factors (time state, class label or tea type).

Cross is a integer value $k > 0$, so that a k-fold cross validation on the training data is performed to assess the quality of the model: the accuracy rate for classification and the Mean Squared Error for regression. We chose not to do cross-validation as we trained the model using a portion of the data, randomly sampled, and tested it on the rest of the data.

Cost is the cost of constraint violation, which greatly affects the classification ability. This is the upper limit of the regularisation term of the Lagrange multipliers, i.e. the value of the largest α_i in equation (3.33). We experimented with different cost values in a grid search approach.

Fitted is a logical argument indicating whether the fitted values should be computed and included in the model or not. We chose computing the fitted values.

The kernel is used in training. Different types of kernel can be used, such as the *linear*, *polynomial*, *radial basis* (also known as the Gaussian kernel), and *sigmoid* (tanh) kernels. By default the kernel parameter γ (except in the linear kernel which has no parameter) is either 1 or 1/(*data dimension*) depending on whether the input data is a vector or not. For the polynomial kernel the default value of η is 0, but we found the optimum value for each set of features used.

Appendix II(b): LDA in R

LDA is employed here by using a generic function *lda* in R library MASS. The function has optional arguments, some of which are:

Prior defines the prior probabilities of class membership. If unspecified, the class proportions for the training set are used. Here we have chosen the default setting. *Tol* specifies the tolerance to decide if a matrix is singular; R will reject variables and linear combinations of unit-variance variables whose variance is less than tol^2.

Cross-validation, *CV*, is a logical argument. If it is true, R returns results (classes and posterior probabilities) for leave-one-out cross-validation. If we set CV as true, the fitted model itself contains the predicted class for the trained data set.

Since we aimed to compare the mean absolute error of the predicted class using different classifiers on the same test set data we set $CV = FALSE$ and used the fitted model to predict class for the test data set using another generic function, *predict*. This returns the prior probabilities used for each class, the group means (the average value of each feature) for each class, the coefficients of the linear discriminants (LDs), and the proportion of the trace of each of the LDs. The coefficients of the LDs are given in a matrix containing the coefficients for each discriminant function, normalised so that the within-groups covariance matrix is spherical and the proportions of the trace are the singular values, which give the ratio of the between- and within-group standard deviations of the linear discriminant variables.

Appendix II(c): Neural network in R

The neural network classifier was applied in R using function *nnet* in the library *nnet*. It fits a feed forward single hidden layer neural network. The architecture of such networks is explained in Section 3.11. Some of the optional arguments of this function are 'weights', 'size', 'Wts', 'mask', 'linout', 'entropy', 'softmax', 'censored', 'skip', 'rang', 'decay', 'maxit', 'Hess', 'trace', 'MaxNWts', 'abstol' and 'reltol'.

Size defines the number of units in the hidden layer, which is set to zero if the hidden layer is to be skipped. The argument size has a great effect so we used a grid search approach to find the optimum number of hidden units for each set of images.

Decay specifies a parameter for weight decay. The default is 0 and we also used grid search to find the optimum value of the parameter.

We set the maximum number of iterations *Maxit* to 5000 (the default is 100) for each set of images. The synthetic images required a higher number of iterations (3000–3500) as they have 100 time points and 100 observations at each time point, but for any of the set of real images the optimisation problem converges quickly.

Rang specifies the range used for the initial random weights, which lie in the interval [-rang, rang] with default value 0.7. For input data with large values it is recommended to choose the value $rang = 1/\max|data|$, but we used a grid search approach to find the best value of the parameter for all sets of images. This provided better classification results in many cases especially when the maximum of the input data is large, e.g. for the synthetic images. For the real images, the maximum value in the data set was not very high, so using either the default value of range or $1/\max|data|$ does not make much difference.

There are many possible activation functions. Options here are the default logistic function or the linear one. We used the logistic function but also tested the linear function and this made little difference to our results.

The function *predict* returns the predicted class for each observation in the test data set. We used argument *type* as 'class' to indicate that *predict* should return class labels and not numeric values, for each observation in the test data set.

Appendix II(d): Summary of PS moments I

Means, sds and coefficients of variation (CVs) for the foreground and background PS moments from the pyramid and ellipse images, as discussed in Section 5.6, are shown in Tables V_1–V_4. M1–M6 and S1–S6 represent the means and sds from the six SEs used.

Table V_1: Means, sds and CVs of the foreground PS moments for pyramid images using 6 SEs at every 5^{th} time point.

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	2.42	0.39	2.43	2.43	1.94	1.24	0.87	0.48	0.88	0.88	1.67	0.17
6	2.54	0.57	2.78	2.78	2.38	2.39	0.99	0.60	1.22	1.22	1.87	1.87
11	3.18	1.31	4.24	4.32	4.10	4.14	1.46	1.00	2.36	2.43	2.43	2.45
16	4.18	2.16	6.13	6.22	6.08	6.11	2.00	1.36	3.42	3.45	2.86	2.89
21	5.39	3.06	8.20	8.25	8.16	8.19	2.59	1.75	4.45	4.47	3.50	3.53
26	6.51	3.93	10.05	10.09	10.02	10.11	3.15	2.20	5.42	5.45	4.27	4.34
31	7.42	4.66	11.57	11.58	11.61	11.69	3.60	2.60	6.27	6.26	5.00	5.09
36	8.29	5.36	13.06	13.04	13.14	13.23	4.05	2.95	7.15	7.08	5.74	5.83
41	9.25	6.13	14.68	14.62	14.80	14.94	4.57	3.33	8.18	8.03	6.57	6.70
46	10.04	6.73	16.05	1598.00	16.25	16.36	4.94	3.56	9.03	8.78	7.31	7.38
51	10.93	7.35	17.61	17.52	17.93	17.88	5.42	3.83	10.02	9.79	8.33	8.16
56	11.84	8.02	19.23	19.16	19.67	19.55	5.88	4.11	11.10	10.84	9.30	9.04
61	12.80	8.67	21.12	20.92	21.60	21.35	6.36	4.39	12.38	12.00	10.54	10.09
66	13.80	9.34	22.91	22.84	23.53	23.25	6.82	4.65	13.44	13.25	11.71	11.23
71	14.94	10.09	25.10	25.14	25.92	25.41	7.44	5.01	14.96	14.95	13.41	12.64
76	15.98	10.74	27.20	27.22	28.06	27.50	7.92	5.25	16.38	16.43	14.78	13.99
81	17.22	11.51	26.60	29.84	30.51	30.06	8.58	5.60	18.00	18.44	16.49	15.96
86	18.56	12.31	32.33	32.67	33.45	32.68	9.34	5.99	19.89	20.57	18.67	17.93
91	19.77	13.01	34.86	35.28	35.95	35.30	9.96	6.31	21.70	22.56	20.47	19.97
96	21.22	13.85	38.06	38.42	39.03	38.47	10.77	6.73	24.20	25.00	22.78	22.52

Mean of the foreground PS moments for pyramid images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.22	0.06	0.17	0.17	0.15	0.15	0.13	0.07	0.10	0.10	0.24	0.24
6	0.45	0.40	0.87	0.84	0.95	0.94	0.29	0.28	0.69	0.69	0.40	0.41
11	0.94	0.79	1.70	1.71	1.78	1.77	0.56	0.42	1.10	1.12	0.58	0.58
16	1.26	0.96	2.10	2.18	2.11	2.12	0.65	0.47	1.14	1.16	0.73	0.75
21	1.48	1.18	2.46	2.44	2.48	2.52	0.75	0.64	1.36	1.33	1.12	1.15
26	1.65	1.34	2.78	2.65	2.80	2.88	0.87	0.74	1.62	1.58	1.39	1.52
31	1.72	1.39	2.87	2.81	2.96	3.12	0.93	0.72	1.73	1.70	1.49	1.76
36	1.68	1.35	2.81	2.84	2.92	3.14	0.95	0.68	1.79	1.82	1.60	1.89
41	1.54	1.21	2.68	2.57	2.67	2.82	0.94	0.64	1.96	1.75	1.51	1.83
46	1.53	1.19	2.63	2.61	2.71	2.71	0.95	0.64	2.07	1.79	1.60	1.89
51	1.57	1.17	2.75	2.91	2.94	2.73	0.97	0.63	2.17	2.09	1.99	1.79
56	1.71	1.28	2.94	3.16	3.28	2.89	1.08	0.73	2.29	2.52	2.33	1.84
61	1.77	1.27	3.36	3.40	3.57	3.12	1.11	0.73	2.73	2.79	2.72	2.24
66	1.83	1.30	3.33	3.64	3.68	3.16	1.18	0.78	2.78	3.20	3.01	2.47
71	1.90	1.32	3.56	3.91	4.08	3.34	1.20	0.83	3.09	3.76	3.67	2.73
76	1.99	1.34	3.84	4.08	4.42	3.62	1.24	0.82	3.54	4.10	4.13	3.28
81	2.22	1.47	4.33	4.53	4.62	4.13	1.39	0.89	3.98	4.67	4.34	4.08
86	2.33	1.50	4.70	5.24	5.26	4.63	1.56	0.93	4.45	5.26	5.27	4.96
91	2.45	1.52	5.16	5.50	5.73	5.18	1.63	0.91	5.00	5.50	5.89	5.56
96	2.49	1.52	5.51	6.00	5.83	6.07	1.74	0.96	5.63	6.08	6.00	6.49

Sd of the foreground PS moments for pyramid images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.09	0.16	0.07	0.07	0.08	0.08	0.15	0.14	0.11	0.11	0.14	0.14
6	0.18	0.70	0.31	0.30	0.40	0.39	0.29	0.46	0.56	0.57	0.22	0.22
11	0.30	0.60	0.40	0.39	0.43	0.43	0.38	0.42	0.47	0.46	0.24	0.24
16	0.30	0.44	0.34	0.35	0.35	0.35	0.32	0.35	0.33	0.34	0.26	0.26
21	0.28	0.39	0.30	0.30	0.30	0.31	0.29	0.37	0.31	0.30	0.32	0.33
26	0.25	0.34	0.28	0.26	0.28	0.29	0.28	0.34	0.30	0.29	0.32	0.35
31	0.23	0.30	0.25	0.24	0.26	0.27	0.26	0.28	0.28	0.27	0.30	0.35
36	0.20	0.25	0.22	0.22	0.22	0.24	0.23	0.23	0.25	0.26	0.28	0.32
41	0.17	0.20	0.18	0.18	0.18	0.19	0.21	0.19	0.24	0.22	0.23	0.27
46	0.15	0.18	0.16	0.16	0.17	0.17	0.19	0.18	0.23	0.20	0.22	0.26
51	0.14	0.16	0.16	0.17	0.16	0.15	0.18	0.16	0.22	0.21	0.24	0.22
56	0.14	0.16	0.15	0.16	0.17	0.15	0.18	0.18	0.21	0.23	0.25	0.20
61	0.14	0.15	0.16	0.16	0.17	0.15	0.17	0.17	0.22	0.23	0.26	0.22
66	0.13	0.14	0.15	0.16	0.16	0.14	0.17	0.17	0.21	0.24	0.26	0.22
71	0.13	0.13	0.14	0.16	0.16	0.13	0.16	0.17	0.21	0.25	0.27	0.22
76	0.12	0.12	0.14	0.15	0.16	0.13	0.16	0.16	0.22	0.25	0.28	0.23
81	0.13	0.13	0.15	0.15	0.15	0.14	0.16	0.16	0.22	0.25	0.26	0.26
86	0.13	0.12	0.15	0.16	0.16	0.14	0.17	0.15	0.22	0.26	0.28	0.28
91	0.12	0.12	0.15	0.16	0.16	0.15	0.16	0.14	0.23	0.24	0.29	0.28
96	0.12	0.11	0.14	0.16	0.15	0.16	0.16	0.14	0.23	0.24	0.26	0.29

CV of the foreground PS moments for pyramid images

Table V_2: Means, sds and CVs of the background PS moments for pyramid images using 6 SEs at every 5^{th} time point.

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	245.27	179.87	254.88	254.89	240.55	240.55	20.82	37.74	3.70	3.63	85.35	85.35
6	244.08	179.03	254.85	254.86	240.52	240.52	23.98	38.17	4.12	4.03	85.35	85.35
11	241.44	177.29	254.71	254.71	140.42	140.42	30.81	39.20	5.69	5.86	85.38	85.38
16	238.42	174.64	254.36	254.36	240.13	240.13	37.03	40.90	8.59	8.64	85.44	85.44
21	234.93	171.50	253.61	253.57	239.65	239.65	43.48	43.12	12.77	13.06	85.63	85.63
26	228.27	167.75	252.53	252.44	238.66	238.66	52.69	45.52	17.33	17.73	85.86	85.86
31	221.51	163.90	250.91	250.79	237.45	237.45	60.07	47.98	22.44	22.83	86.24	86.24
36	214.04	160.33	248.92	248.75	2235.82	2235.82	66.11	50.17	27.53	28.03	86.75	86.75
41	206.26	156.46	246.48	246.24	2234.03	2234.03	71.39	52.50	32.84	33.31	87.30	87.30
46	201.33	152.31	243.70	243.40	232.03	232.03	74.69	54.63	37.87	38.37	88.04	88.04
51	194.44	147.90	240.64	240.31	229.64	229.64	78.17	56.63	42.66	43.17	88.80	88.80
56	187.80	142.75	237.05	236.77	226.72	226.72	80.72	58.23	47.57	48.02	89.72	89.72
61	179.72	137.72	233.11	232.73	223.41	223.41	82.73	59.47	52.30	52.82	90.64	90.64
66	171.51	13.57	228.40	228.05	219.48	219.48	84.35	60.25	57.26	57.72	91.68	91.68
71	162.72	126.92	223.41	223.12	215.46	215.46	84.35	60.73	61.77	62.26	92.78	92.78
76	153.05	121.85	218.04	217.89	211.01	211.01	83.77	60.79	66.09	66.42	93.89	93.89
81	144.90	116.49	212.50	212.31	206.34	206.34	82.56	60.53	69.90	70.33	95.01	95.01
86	133.18	111.00	206.06	206.10	200.43	200.43	78.41	59.75	73.57	73.86	95.86	95.86
91	123.92	105.32	199.93	200.03	194.50	194.50	75.49	58.54	76.52	76.88	96.51	96.51
96	113.37	99.25	193.31	193.58	188.22	188.22	69.87	56.72	79.13	79.53	96.77	96.77

Mean of the background PS moments for pyramid images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	8.81	0.00	0.06	0.06	0.09	0.09	13.99	0.00	1.21	1.22	0.04	0.04
6	8.35	1.47	0.08	0.08	0.10	0.10	12.97	0.79	1.29	1.30	0.04	0.04
11	7.87	2.69	0.19	0.18	0.15	0.15	11.05	1.41	2.02	1.79	0.07	0.07
16	7.7	4.07	0.44	0.41	0.41	0.41	10.06	2.16	2.89	2.77	0.17	0.17
21	7.79	4.43	0.83	0.79	0.69	0.69	9.12	2.51	3.95	3.67	0.33	0.33
26	6.98	5.52	1.22	1.12	1.15	1.15	7.32	3.04	4.32	3.97	0.56	0.56
31	8.46	6.09	1.80	1.72	1.68	1.68	7.25	3.26	5.02	4.87	0.74	0.74
36	10.13	6.37	2.37	2.25	2.39	2.39	6.84	3.32	5.41	5.17	0.94	0.94
41	9.21	6.52	2.72	2.59	2.60	2.60	5.37	3.19	5.18	5.15	1.13	1.13
46	9.73	6.87	3.20	3.15	3.15	3.15	5.42	2.94	5.27	5.30	1.32	1.32
51	10.51	7.55	3.85	3.64	3.83	3.83	5.11	2.91	5.36	5.37	1.73	1.73
56	11.35	8.32	4.56	4.14	4.39	4.39	4.43	2.96	5.47	5.27	1.99	1.99
61	13.88	8.84	5.39	4.75	5.54	5.54	3.75	3.12	5.52	5.33	2.34	2.34
66	12.73	9.81	5.88	5.40	6.09	6.09	3.89	3.83	5.21	5.15	2.46	2.46
71	14.41	10.53	6.56	5.75	6.88	6.88	4.96	4.65	5.05	4.73	2.58	2.58
76	13.97	11.10	7.01	6.08	7.29	7.29	6.09	5.49	4.85	4.60	3.02	3.02
81	15.07	11.41	7.27	6.28	7.58	7.58	8.79	6.39	4.36	4.15	3.25	3.25
86	16.29	12.00	7.81	6.89	8.61	8.61	11.28	7.16	3.88	3.91	3.68	3.68
91	16.05	12.56	8.96	7.68	9.61	9.61	11.52	8.07	3.58	3.65	4.25	4.25
96	16.38	12.73	9.65	8.23	10.33	10.33	11.92	8.64	3.16	3.39	4.57	4.57

Sd of the background PS moments for pyramid images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.04	0.00	0.00	0.00	0.00	0.00	0.67	0.00	0.33	0.34	0.00	0.00
6	0.03	0.01	0.00	0.00	0.00	0.00	0.54	0.02	0.31	0.32	0.00	0.00
11	0.03	0.02	0.00	0.00	0.00	0.00	0.36	0.04	0.35	0.31	0.00	0.00
16	0.03	0.02	0.00	0.00	0.00	0.00	0.27	0.05	0.34	0.32	0.00	0.00
21	0.03	0.03	0.00	0.00	0.00	0.00	0.21	0.06	0.31	0.28	0.00	0.00
26	0.03	0.03	0.00	0.00	0.00	0.00	0.14	0.07	0.25	0.22	0.01	0.01
31	0.04	0.04	0.01	0.01	0.01	0.01	0.12	0.07	0.22	0.21	0.01	0.01
36	0.05	0.04	0.01	0.01	0.01	0.01	0.10	0.07	0.20	0.18	0.01	0.01
41	0.04	0.04	0.01	0.01	0.01	0.01	0.08	0.06	0.16	0.15	0.01	0.01
46	0.05	0.05	0.01	0.01	0.01	0.01	0.07	0.05	0.14	0.14	0.02	0.02
51	0.05	0.05	0.02	0.02	0.02	0.02	0.07	0.05	0.13	0.12	0.02	0.02
56	0.06	0.06	0.02	0.02	0.02	0.02	0.05	0.05	0.12	0.11	0.02	0.02
61	0.08	0.06	0.02	0.02	0.02	0.02	0.05	0.05	0.11	0.10	0.03	0.03
66	0.07	0.07	0.03	0.02	0.03	0.03	0.05	0.06	0.09	0.09	0.03	0.03
71	0.09	0.08	0.03	0.03	0.03	0.03	0.06	0.08	0.08	0.08	0.03	0.03
76	0.09	0.09	0.03	0.03	0.03	0.03	0.07	0.09	0.07	0.07	0.03	0.03
81	0.10	0.10	0.03	0.03	0.04	0.04	0.11	0.11	0.06	0.06	0.03	0.03
86	0.12	0.11	0.04	0.03	0.04	0.04	0.14	0.12	0.05	0.05	0.04	0.04
91	0.13	0.12	0.04	0.04	0.05	0.05	0.15	0.14	0.05	0.05	0.04	0.04
96	0.14	0.13	0.05	0.04	0.05	0.05	0.17	0.15	0.04	0.04	0.05	0.05

CV of the background PS moments for pyramid images

Table V_3: Means, sds and CVs of the foreground PS moments for ellipse images using 6 SEs at every 5^{th} time point.

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	2.62	0.63	2.91	2.96	2.36	2.90	0.76	0.50	0.86	0.88	1.79	1.95
11	3.36	1.44	4.43	4.62	3.86	5.47	1.42	0.87	1.92	2.01	2.17	2.55
16	4.74	2.46	6.51	6.82	5.70	8.10	2.50	1.48	3.09	3.20	2.96	3.68
21	6.10	3.43	8.42	8.85	7.37	10.55	2.94	2.06	4.15	4.33	3.86	5.03
26	7.47	4.40	10.41	10.89	9.12	13.05	3.72	2.63	5.31	5.46	4.89	6.47
31	8.99	5.47	12.59	13.07	11.07	15.63	4.62	3.25	6.65	6.77	6.16	7.97
36	10.29	6.40	14.51	15.05	12.78	18.00	5.35	3.76	7.78	7.89	7.24	9.26
41	11.63	7.30	16.51	17.07	14.57	20.42	6.15	4.27	9.00	9.11	8.44	10.62
46	12.86	8.12	18.50	19.11	16.34	22.81	6.86	4.70	10.34	10.40	9.66	11.93
51	14.43	9.14	20.99	21.73	18.71	25.62	7.79	5.28	11.96	12.15	11.33	13.49
56	16.01	10.17	23.56	24.46	21.04	28.68	8.68	5.28	13.58	13.91	12.80	15.38
61	17.64	11.16	26.31	27.34	23.52	31.97	9.55	6.29	15.30	15.75	14.35	17.52
66	19.32	12.18	29.30	30.29	26-Dec	35.59	10.44	6.78	17.19	17.67	16.08	19.96
71	21.08	13.22	30.61	33.70	29.03	39.54	11.29	7.23	19.43	19.98	18.09	22.52
76	23.14	14.48	36.59	37.72	32.58	44.18	12.24	7.80	22.15	22.59	20.52	25.46
81	25.24	15.76	40.79	42.24	36.36	49.30	13.22	8.42	22.03	25.73	23.26	28.91
86	27.74	17.31	46.02	47.64	41.13	55.47	14.34	9.21	28.56	29.35	26.66	32.97
91	30.36	18.21	51.74	53.48	46.31	62.01	15.48	9.99	32.48	33.26	30.26	37.02
96	33.49	20.83	58.23	60.53	51.32	69.49	16.81	10.88	36.37	37.64	34.00	41.38

Mean of the foreground PS moments for ellipse images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.48	0.29	0.64	0.67	0.76	0.99	0.28	0.13	0.40	0.43	0.20	0.21
11	1.33	0.91	1.80	1.85	1.71	1.87	0.54	0.49	0.79	0.82	0.53	0.69
16	1.98	1.41	2.70	2.70	2.46	2.79	0.86	0.67	1.34	1.25	1.03	1.41
21	2.23	1.62	3.07	3.07	2.81	3.22	1.11	0.81	1.62	1.57	1.46	1.68
26	2.40	1.72	3.31	3.29	3.05	3.43	1.28	0.87	1.79	1.82	1.73	1.86
31	2.58	1.86	3.47	3.50	3.29	3.64	1.49	1.02	2.05	2.07	1.98	2.06
36	2.69	1.91	3.59	3.71	3.42	3.79	1.66	1.13	2.32	2.33	2.17	2.21
41	2.81	1.98	3.82	3.85	3.61	3.81	1.77	1.21	2.64	2.51	2.38	2.32
46	2.86	2.03	4.11	4.04	3.81	3.96	1.85	1.28	3.18	2.81	2.72	2.56
51	3.14	2.31	5.00	4.54	4.49	4.53	2.01	1.53	4.04	3.34	3.45	2.95
56	3.25	2.37	5.83	4.92	4.87	5.08	2.15	1.63	5.04	3.82	3.78	3.60
61	3.48	2.42	6.40	5.55	5.36	5.57	2.33	1.73	5.44	4.67	4.18	4.26
66	3.74	2.59	7.28	6.11	5.91	6.56	2.43	1.74	5.92	5.16	4.56	5.48
71	3.91	2.64	7.95	6.82	6.53	7.37	2.51	1.76	6.71	6.03	5.15	6.20
76	4.06	2.48	8.27	7.30	6.98	7.66	2.62	1.88	7.21	6.41	5.66	6.49
81	4.37	2.88	8.91	8.44	7.59	8.69	2.77	2.08	7.99	7.36	6.26	7.47
86	4.95	3.30	10.08	9.95	8.99	10.48	3.11	2.38	8.99	8.19	7.71	8.64
91	5.31	3.48	10.93	11.21	10.28	11.79	3.29	2.45	10.05	8.81	9.21	9.39
96	5.96	3.88	12.48	12.90	12.02	13.63	3.63	2.60	10.98	10.00	10.41	10.57

Sd of the foreground PS moments for ellipse images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.18	0.47	0.22	0.23	0.32	0.34	0.36	0.26	0.47	0.49	0.11	0.11
11	0.40	0.64	0.41	0.40	0.44	0.34	0.38	0.56	0.41	0.41	0.24	0.27
16	0.42	0.57	0.41	0.40	0.43	0.35	0.40	0.45	0.43	0.39	0.35	0.38
21	0.37	0.47	0.37	0.35	0.38	0.30	0.38	0.39	0.39	0.36	0.38	0.33
26	0.32	0.39	0.32	0.30	0.33	0.26	0.34	0.33	0.34	0.33	0.35	0.29
31	0.29	0.34	0.28	0.27	0.30	0.23	0.32	0.31	0.31	0.31	0.32	0.26
36	0.26	0.30	0.25	0.25	0.27	0.21	0.31	0.30	0.30	0.29	0.30	0.24
41	0.24	0.27	0.23	0.23	0.25	0.19	0.29	0.28	0.29	0.28	0.28	0.22
46	0.22	0.25	0.22	0.21	0.23	0.17	0.27	0.27	0.31	0.27	0.28	0.21
51	0.22	0.25	0.24	0.21	0.24	0.18	0.26	0.29	0.34	0.27	0.30	0.22
56	0.20	0.23	0.25	0.20	0.23	0.18	0.25	0.28	0.37	0.27	0.30	0.23
61	0.20	0.22	0.24	0.20	0.23	0.17	0.24	0.28	0.36	0.30	0.29	0.24
66	0.19	0.21	0.25	0.20	0.23	0.18	0.23	0.26	0.34	0.29	0.28	0.27
71	0.19	0.20	0.24	0.20	0.22	0.19	0.22	0.24	0.35	0.30	0.28	0.28
76	0.18	0.18	0.23	0.19	0.21	0.17	0.21	0.24	0.33	0.28	0.28	0.25
81	0.17	0.18	0.22	0.20	0.21	0.18	0.21	0.25	0.32	0.29	0.27	0.26
86	0.18	0.19	0.22	0.21	0.22	0.19	0.22	0.26	0.31	0.28	0.29	0.26
91	0.17	0.18	0.21	0.21	0.22	0.19	0.21	0.25	0.31	0.27	0.30	0.25
96	0.18	0.19	0.21	0.21	0.23	0.20	0.22	0.24	0.30	0.27	0.31	0.26

CV of the foreground PS moments for ellipse images

Table V_4: Means, sds and CVs of the background PS moments for ellipse images using 6 SEs at every 5^{th} time point.

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	255.00	117.31	133.00	255.00	240.64	240.64	0.00	23.19	0.00	0.00	85.32	85.32
6	247.12	117.24	132.99	254.84	240.48	240.48	24.11	23.35	0.42	4.25	85.36	85.36
11	239.83	117.14	132.98	254.32	239.85	239.85	36.73	23.55	1.08	8.85	85.50	85.50
16	226.94	116.12	132.92	252.94	238.24	238.24	51.12	23.90	2.18	15.79	85.87	85.87
21	211.97	114.20	132.79	250.59	235.45	235.45	61.70	24.98	3.56	23.28	86.58	86.58
26	196.19	108.87	132.56	246.95	231.30	231.30	69.46	28.53	5.27	31.34	87.67	87.67
31	177.30	99.22	132.17	241.70	225.18	225.18	74.11	34.12	7.30	40.00	89.09	89.09
36	158.57	87.34	131.56	235.00	217.70	217.70	74.56	36.21	9.72	48.27	90.71	90.71
41	138.95	77.18	130.67	226.88	208.52	208.52	71.19	34.60	12.41	56.13	92.21	92.21
46	120.34	69.63	129.46	217.34	198.01	198.01	46.13	34.89	15.31	63.11	92.97	92.97
51	101.90	63.05	127.81	205.94	185.96	185.96	53.94	32.74	18.53	69.41	93.36	93.36
56	88.09	53.73	125.83	194.26	173.25	173.25	45.88	25.88	21.68	73.98	92.69	92.69
61	76.46	47.97	123.33	181.25	159.91	159.91	38.40	24.09	24.94	77.14	90.86	90.86
66	67.25	42.32	120.28	167.49	145.75	145.75	32.94	20.06	28.20	78.66	87.80	87.80
71	60.09	35.22	116.80	153.78	132.02	132.02	29.16	13.91	31.24	78.51	83.52	83.52
76	54.21	32.68	112.90	140.34	118.93	118.93	26.10	13.14	34.00	76.90	78.24	78.24
81	49.28	30.47	108.63	127.69	106.57	106.57	23.66	12.33	36.43	74.09	72.04	72.04
86	44.93	28.51	103.92	115.42	94.93	94.93	21.57	11.85	38.49	70.22	65.12	65.12
91	41.29	26.06	99.07	104.49	84.81	84.81	19.96	11.23	40.09	65.93	58.66	58.66
96	38.09	24.23	93.98	94.61	75.86	75.86	18.55	11.06	41.21	61.25	52.73	52.73

Mean of the background PS moments for ellipse images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.00	0.00	0.00	0.00	0.02	0.02	0.00	0.00	0.00	0.00	0.01	0.01
6	4.46	0.00	0.01	0.14	0.15	0.15	10.09	0.00	0.41	1.75	0.08	0.08
11	6.64	0.00	0.02	0.43	0.62	0.62	9.42	0.00	0.64	2.91	0.26	0.26
16	8.89	0.00	0.06	0.92	1.49	1.49	8.13	0.00	0.82	3.65	0.59	0.59
21	10.89	0.00	0.11	1.59	2.55	2.55	6.80	0.00	1.06	4.25	1.07	1.07
26	12.53	0.00	0.18	2.61	3.94	3.94	5.50	0.00	1.24	5.00	1.64	1.64
31	13.59	0.00	0.32	3.83	5.54	5.54	5.38	0.00	1.50	5.36	2.33	2.33
36	15.12	0.00	0.46	5.27	6.62	6.62	7.94	0.00	1.67	5.63	2.98	2.98
41	15.52	0.00	0.63	6.03	7.84	7.84	10.94	0.00	1.75	4.86	3.66	3.66
46	15.07	0.00	0.88	7.50	9.95	9.95	13.84	0.00	1.93	4.54	4.76	4.76
51	12.39	0.00	1.11	8.47	11.09	11.09	13.14	0.00	1.94	3.84	5.93	5.93
56	9.56	0.00	1.37	9.14	11.56	11.56	10.75	0.00	1.98	3.10	6.99	6.99
61	7.94	0.00	1.72	10.13	12.30	12.30	8.20	0.00	2.06	2.53	7.93	7.93
66	6.81	0.00	2.09	11.04	12.23	12.23	6.44	0.00	2.06	2.12	8.77	8.77
71	5.84	0.00	2.53	11.55	12.20	12.20	5.60	0.00	1.95	2.50	9.64	9.64
76	5.70	0.00	3.15	11.96	12.31	12.31	4.73	0.00	1.97	3.55	10.63	10.63
81	4.56	0.00	3.64	12.03	12.16	12.16	4.06	0.00	1.89	4.41	10.96	10.96
86	3.98	0.00	4.27	11.86	12.72	12.72	3.32	0.00	1.73	5.56	11.28	11.28
91	3.55	0.00	4.87	11.05	10.94	10.94	2.97	0.00	1.53	6.05	11.10	11.10
96	3.38	0.00	5.58	10.96	9.84	9.84	2.68	0.00	1.47	7.12	10.19	10.19

Sd of the background PS moments for ellipse images

Time	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	S5	S6
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.02	0.00	0.00	0.00	0.00	0.00	0.42	0.00	0.98	0.41	0.00	0.00
11	0.03	0.00	0.00	0.00	0.00	0.00	0.26	0.00	0.59	0.33	0.00	0.00
16	0.04	0.00	0.00	0.00	0.01	0.01	0.16	0.00	0.38	0.23	0.01	0.01
21	0.05	0.00	0.00	0.01	0.01	0.01	0.11	0.00	0.30	0.18	0.01	0.01
26	0.06	0.00	0.00	0.01	0.02	0.02	0.08	0.00	0.23	0.16	0.02	0.02
31	0.08	0.00	0.00	0.02	0.02	0.02	0.07	0.00	0.21	0.13	0.03	0.03
36	0.10	0.00	0.00	0.02	0.03	0.03	0.11	0.00	0.17	0.12	0.03	0.03
41	0.11	0.00	0.00	0.03	0.04	0.04	0.15	0.00	0.14	0.09	0.04	0.04
46	0.13	0.00	0.01	0.03	0.05	0.05	0.22	0.00	0.13	0.07	0.05	0.05
51	0.12	0.00	0.01	0.04	0.06	0.06	0.24	0.00	0.10	0.06	0.06	0.06
56	0.11	0.00	0.01	0.05	0.07	0.07	0.23	0.00	0.09	0.04	0.08	0.08
61	0.10	0.00	0.01	0.06	0.08	0.08	0.21	0.00	0.08	0.03	0.09	0.09
66	0.10	0.00	0.02	0.07	0.08	0.08	0.20	0.00	0.07	0.03	0.10	0.10
71	0.10	0.00	0.02	0.08	0.09	0.09	0.19	0.00	0.06	0.03	0.12	0.12
76	0.10	0.00	0.03	0.09	0.10	0.10	0.18	0.00	0.06	0.05	0.14	0.14
81	0.09	0.00	0.03	0.09	0.11	0.11	0.17	0.00	0.05	0.06	0.15	0.15
86	0.09	0.00	0.04	0.10	0.12	0.12	0.15	0.00	0.05	0.08	0.17	0.17
91	0.09	0.00	0.05	0.11	0.13	0.13	0.15	0.00	0.04	0.09	0.19	0.19
96	0.09	0.00	0.06	0.11	0.13	0.13	0.14	0.00	0.04	0.12	0.19	0.19

CV of the background PS moments for ellipse images

Appendix III: Summary of PS moments II

Means, sds and CVs of the PS moments using a square and a disk SE from the bottom-hat transformed corrosion images are shown in Table VI, discussed in Section 6.4.

Table VI: Means, sds and CVs of the PS moments for the hat-transformed corrosion images.

Time	M1	M2	SD1	SD2	SK1	SK2	KT1	KT2
1	6.6565	3.2607	4.5828	2.6405	0.0032	0.0227	-2.9977	-2.9817
2	7.5256	3.7773	5.4877	3.1449	0.0022	0.0141	-2.9989	-2.9908
3	8.8921	4.5955	6.2471	3.5949	0.0014	0.0085	-2.9993	-2.9945
4	10.1779	5.3489	7.2285	4.1534	0.0009	0.0054	-2.9996	-2.9969
5	11.2492	5.9891	7.9117	4.5508	0.0007	0.0041	-2.9997	-2.9978
6	12.4619	6.6898	8.9238	5.0891	0.0005	0.0029	-2.9998	-2.9986
7	13.7676	7.4305	10.1043	5.6975	0.0004	0.0020	-2.9999	-2.9991
8	14.8493	7.9708	11.2805	6.1374	0.0003	0.0016	-2.9999	-2.9993
9	17.7661	9.5363	12.6737	6.7399	0.0002	0.0010	-3.0000	-2.9995
10	17.8071	9.5141	13.8678	7.4247	0.0002	0.0008	-3.0000	-2.9997

Mean of the PS moments for the corrosion images using square and disk SEs

Time	M1	M2	SD1	SD2	SK1	SK2	KT1	KT2
1	0.1660	0.0865	0.0863	0.0503	0.0004	0.0020	0.0002	0.0013
2	0.1771	0.1097	0.1258	0.0783	0.0002	0.0014	0.0001	0.0009
3	0.3253	0.2095	0.1975	0.1016	0.0002	0.0012	0.0001	0.0007
4	0.3726	0.2391	0.2313	0.1100	0.0001	0.0007	0.0000	0.0004
5	0.5808	0.3508	0.2325	0.1269	0.0001	0.0008	0.0000	0.0002
6	0.3749	0.2277	0.2556	0.1215	0.0001	0.0002	0.0000	0.0001
7	0.3730	0.2129	0.3341	0.1882	0.0001	0.0003	0.0000	0.0001
8	0.4001	0.2162	0.3762	0.1470	0.0000	0.0002	0.0000	0.0001
9	0.8933	0.5197	0.4820	0.1704	0.0000	0.0001	0.0000	0.0000
10	0.8268	0.4252	0.8622	0.3645	0.0000	0.0002	0.0000	0.0001

Mean of the PS moments for the corrosion images using square and disk SEs

Time	M1	M2	SD1	SD2	SK1	SK2	KT1	KT2
1	0.0249	0.0265	0.0188	0.0191	0.1183	0.0864	-0.0001	-0.0004
2	0.0235	0.0291	0.0229	0.0249	0.1029	0.1021	0.0000	-0.0003
3	0.0366	0.0456	0.0316	0.0283	0.1476	0.1352	0.0000	-0.0002
4	0.0366	0.0447	0.0320	0.0265	0.1245	0.1220	0.0000	-0.0001
5	0.0516	0.0586	0.0294	0.0279	0.1751	0.1836	0.0000	-0.0001
6	0.0301	0.0340	0.0286	0.0239	0.1041	0.0850	0.0000	0.0000
7	0.0271	0.0286	0.0331	0.0330	0.1931	0.1680	0.0000	0.0000
8	0.0269	0.0271	0.0334	0.0240	0.1065	0.1060	0.0000	0.0000
9	0.0505	0.0545	0.0380	0.0253	0.1354	0.1431	0.0000	0.0000
10	0.0640	0.0447	0.0622	0.0491	0.2020	0.1795	0.0000	0.0000

Mean of the PS moments for the corrosion images using square and disk SEs

Note: M1=Mean from a square SE; M2=Mean from a disk SE; SD1=Sd from a square SE; SD2=SD from a disk SE; SK1=Skewness from a square SE; SK2=Skewness from a disk SE; KT1=Kurtosis from a square SE; KT2=Kurtosis from a disk SE.

Appendix IV: Summary of PS moments III

Table VII shows the means, sds and CVs of the PS moments using a square and a disk SE from the top-hat transformed Indian tea images, discussed in Section 7.5.1.

Table VII: Means, sds and CVs of the PS moments from the hat-transformed tea images.

Class	M1	M2	SD1	SD2	SK1	SK2	KT1	KT2
1	8.8704	4.1926	5.3168	2.8186	0.0010	0.0097	-1.0169	-1.4163
2	9.3842	4.7665	5.8495	3.3328	0.0009	0.0070	-0.9573	-1.2406
3	9.9126	5.4027	6.3916	3.8846	0.0008	0.0047	-0.9266	-1.1598
4	10.4323	5.9379	6.8994	4.3667	0.0007	0.0035	-0.8802	-1.0734
5	11.0887	6.5586	7.4201	4.8688	0.0006	0.0027	-0.8570	-0.9884
6	11.6420	7.0889	7.8931	5.3397	0.0005	0.0021	-0.8090	-0.8946
7	12.2653	7.6914	8.4368	5.8734	0.0005	0.0017	-0.7841	-0.8389
8	12.7290	8.2018	8.9882	6.4256	0.0004	0.0013	-0.7519	-0.7791

Mean of the PS moments for the tea images using square and disk SEs

Class	M1	M2	SD1	SD2	SK1	SK2	KT1	KT2
1	0.2161	0.1056	0.0894	0.0442	0.0002	0.0014	0.0658	0.0591
2	0.2125	0.1259	0.0975	0.0622	0.0002	0.0009	0.0728	0.0723
3	0.2495	0.1534	0.1225	0.0711	0.0001	0.0006	0.0867	0.0861
4	0.1932	0.1292	0.1264	0.0905	0.0001	0.0003	0.0739	0.0674
5	0.3364	0.2279	0.1940	0.1346	0.0001	0.0004	0.1210	0.1282
6	0.3185	0.2245	0.1934	0.1346	0.0001	0.0003	0.1031	0.1278
7	0.3809	0.2856	0.2432	0.1856	0.0001	0.0003	0.1161	0.1587
8	0.3375	0.2901	0.2456	0.2087	0.0001	0.0002	0.0976	0.1447

Sd of the PS moments for the tea images using square and disk SEs

Class	M1	M2	SD1	SD2	SK1	SK2	KT1	KT2
1	0.0244	0.0252	0.0168	0.0157	0.2105	0.1406	-0.0647	-0.0418
2	0.0226	0.0264	0.0167	0.0187	0.1679	0.1297	-0.0761	-0.0583
3	0.0252	0.0284	0.0192	0.0183	0.1629	0.1361	-0.0936	-0.0743
4	0.0185	0.0218	0.0183	0.0207	0.1092	0.0952	-0.0840	-0.0628
5	0.0303	0.0348	0.0261	0.0277	0.1888	0.1638	-0.1412	-0.1297
6	0.0274	0.0317	0.0245	0.0252	0.1528	0.1462	-0.1275	-0.1429
7	0.0311	0.0371	0.0288	0.0316	0.1691	0.1680	-0.1481	-0.1892
8	0.0265	0.0354	0.0273	0.0325	0.1314	0.1598	-0.1298	-0.1857

CV of the PS moments for the tea images using square and disk SEs

Note: M1=Mean from a square SE; M2=Mean from a disk SE; SD1=Sd from a square SE; SD2=SD from a disk SE; SK1=Skewness from a square SE; SK2=Skewness from a disk SE; KT1=Kurtosis from a square SE; KT2=Kurtosis from a disk SE.

Appendix V: Scaling image intensities

The effect of scaling image intensities as $I = round((I - ma) * (L - 1)/(MA - ma))$, where $L = 16$ is the number of bins, and ma and MA are the minimum and maximum intensities of image I respectively (Section 9.5). The histograms of the original image and the transformed image have the same shape, as the transformation only scaled the original image intensities down to 0 to 15.

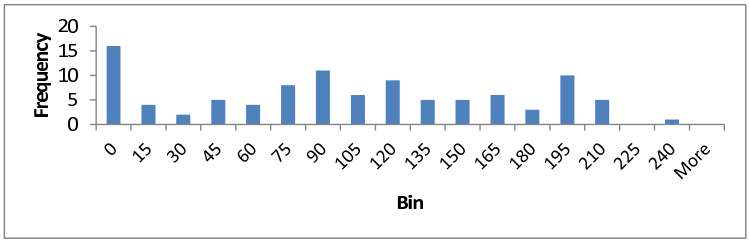
Table VIII: Original image and transformed images used in Matlab function 'hist2' with respective histograms.

0	165	175	188	114	210	141	182	75	155
80	193	203	198	135	65	118	10	0	34
181	165	186	149	117	74	144	70	96	0
31	37	0	0	185	94	150	136	99	81
52	50	186	129	0	0	82	168	153	193
40	6	0	0	91	116	209	63	105	131
0	65	232	193	117	110	157	0	186	87
65	3	0	77	120	123	78	42	87	53
0	0	126	206	167	107	78	17	84	53
154	100	77	90	29	65	114	3	0	0

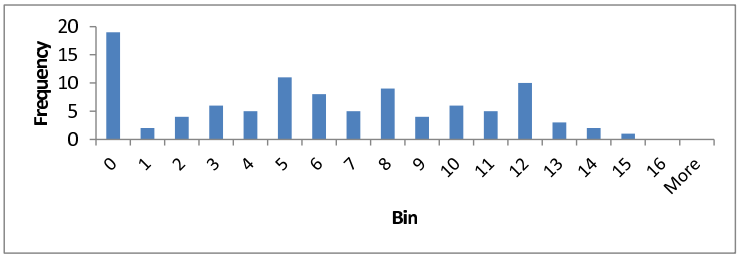
Original image I

0	11	11	12	7	14	9	12	5	10
5	12	13	13	9	4	8	1	0	2
12	11	12	10	8	5	9	5	6	0
2	2	0	0	12	6	10	9	6	5
3	3	12	8	0	0	5	11	10	12
3	0	0	0	6	8	14	4	7	8
0	4	15	12	8	7	10	0	12	6
4	0	0	5	8	8	5	3	6	3
0	0	8	13	11	7	5	1	5	3
10	6	5	6	2	4	7	0	0	0

Scaled image: round((I-min(I(:)))*(nbins-1)/max(I(:))-min(I(:))), nbins=16



Histogram of original image



Histogram of scaled image

Biblography

Alata, O. and C. Ramananjarasoa (2005). Unsupervised textured image segmentation using 2d quarter plane autoregressive model with four prediction supports. Pattern Recognition Letters 26(8), pp. 1069–1081.

Albregtsen, F., B. Nielsen, and H. Danielsen (2000). Adaptive grey level run length features from class distance matrices. In Proceedings of 15th International Conference on Pattern Recognition 3, pp. 738–741.

Arul, P., V. Amin, and D. Carlson (1993). Characterization of beef muscle tissue using texture analysis of ultrasonic images. In Proceedings of 12th Southern Biomedical Engineering Conference, New Orleans, USA, pp. 141–143.

Auran, P. and K. Malvig (1996). Real-time extraction of connected components in 3-D sonar range images. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, pp. 580–585.

Ayala, G. and J. Domingo (2001). Spatial size distributions: applications to shape and texture analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(12), pp. 1430–1442.

Bajcsy, P. and P. Groves (2004). Methodology for hyperspectral band selection. Photogrammetric Engineering and Remote Sensing 70(7), pp. 793–802.

Balas, C., V. Papadakis, N. Papadakis, A. Papadakis, E. Vazgiouraki, and G. Themelis (2003). A novel hyper-spectral imaging apparatus for the nondestructive analysis of objects of artistic and historic value. Journal of Cultural Heritage 4(1), pp. 330–337.

Ballard, D. and C. Brown (1982). Computer Vision. Englewood Cliffs, New Jersey, USA: Prentice Hall.

Baraldi, A. and F. Parmiggiani (1995). An investigation of the textural characteristics associated with gray level co-occurrence matrix statistical parameters. IEEE Transactions on Geoscience and Remote Sensing 33(2), pp. 293–304.

Batman, S. and E. Dougherty (1997). Size distributions for multivariate morphological granulometries: texture classification and statistical properties. Optical Engineering 36(5), pp. 1518–1529.

Batman, S. and E. Dougherty (2001). Morphological granulometric estimation of

random patterns in the context of parameterised random sets. Pattern Recognition 34(6), pp. 1207–1217.

Bazi, Y. and F. Melgani (2006). Toward an optimal SVM classification system for hyperspectral remote sensing images. IEEE Transactions on Geoscience and Remote Sensing 44(11), pp. 3374–3385.

Bekios-Calfa, J., J. Buenaposada, and L. Baumela (2011). Revisiting linear discriminant techniques in gender recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(4), pp. 858–864.

Benediktsson, J., J. Palmason, and J. Sveinsson (2005). Classification of hyperspectral data from urban areas based on extended morphological profiles. IEEE Transactions on Geoscience and Remote Sensing 43(3), pp. 480–491.

Besag, J. (1986). On the statistical analysis of dirty pictures. Journal of Royal Statistical Society 48(3), pp. 259–302.

Bettoli, B. and E. Dougherty (1993). Linear granulometric moments of noisy binary images. Journal of Mathematical Imaging and Vision 3(3), pp. 299–319.

Bhattacharyya, N., S. Seth, B. Tudu, P. Tamuly, A. Jana, D. Ggosh, R. Bandyopadhyay, M. Bhuyan, and S. Sabhapandit (2007). Detection of optimum fermentation time for black tea manufacturing using electronic nose. Sensors and Actuators B 122, pp. 627–634.

Bhuyan, M. and S. Borah (2001). Use of electronic nose in tea industry. In Proceedings of International Conference on Energy, Automation and Information Technology (EAIT), Indian Institute of Technology, Kharagpur, India, pp. 848–853.

Boland, M. and R. Murphy (2001). A neural network classifier capable of organizing the patterns of all major subcellular structures in fluorescence microscope images of HeLa cells. Bioinformatics 17(12), pp. 1213–1223.

Borah, S. and M. Bhuyan (2003). Non-destructive testing of tea fermentation using image processing. Insight 45, pp. 55–58.

Borah, S., M. Bhuyan, and H. Saikia (2002). ANN based colour detection in tea fermentation. In Proceedings of Indian Conference on Computer Vision, Graphics and Image Processing, Ahmadabad, India.

Borah, S., E. Hines, and M. Bhuyan (2007). Wavelet transform based image

texture analysis for size estimation applied to the sorting of tea granules. Journal of Food Engineering 79(2), pp. 629–639.

Bouguila, N. (2011). Count data modeling and classification using finite mixtures of distributions. IEEE Transactions on Neural Networks 22(2), pp. 186–198.

Brodatz, P. (1966). Textures: A Photographic Album for Artists and Designers. New York, USA: Dover.

Burgeth, B., A. Bruhn, S. Didas, J. Weickert, and M. Well (2007). Morphology for matrix data: ordering versus pdf-based approach. Image and Vision Computing 25(4), pp. 496–511.

Burgeth, B., A. Bruhn, N. Papenberg, M. Welk, and J. Weickert (2007). Mathematical morphology for matrix fields induced by the Loewner ordering in higher dimensions. Signal Processing 87(2), pp. 277–290.

Camps-Valls, G., T. Marsheva, and D. Zhou (2007). Semi-supervised graphbased hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing 45(10), pp. 3044–3054.

Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 8(6), pp. 679–698.

Chan, C.-H. and G. Pang (2000). Fabric defect detection by Fourier analysis. IEEE Transactions on Industry Applications 36(5), pp. 1267–276.

Chanda, B. and D. Majumder (1988). A note on the use of the graylevel cooccurrence matrix in threshold selection. Journal of Signal Processing 15(2), pp. 149–167.

Chang, C.-I. (2002). Target signature-constrained mixed pixel classification for hyperspectral imagery. IEEE Transactions on Geoscience and Remote Sensing 40(5), pp. 1065–1081.

Chantler, M. (1995). Why illumination direction is fundamental to texture analysis. IEE Proceedings on Vision, Image and Signal Processing 142(4), pp. 199–206.

Chantler, M., M. Schmidt, M. Petrou, and G. Mc-Gunnigle (2002). The effect of illumination rotation on texture filters: Lissajous's ellipses. In A. Heyden, G. Spar, M. Nielsen and P. Johansen (Eds.), In Proceedings of 7th European Conference on Computer Vision (ECCV 2002), Copenhagen, Denmark, part 3, pp. 289–303.

Chaplot, S., L. Patnaik, and N. Jagannathan (2006). Classification of magnetic resonance brain images using wavelets as input to support vector machine and neural network. Biomedical Signal Processing and Control 1(1), pp. 86–92.

Chaux, C., L. Duval, and J.-C. Pesquet (2006). Image analysis using a dual-tree m-band wavelet transform. IEEE Transactions on Image Processing 15(8), pp. 2397–2404.

Chellappa, R. and S. Chatterjee (1985). Classification of textures using Gaussian Markov random fields. IEEE Transactions on Acoustics, Speech, and Signal Processing 33(4), pp. 959–963.

Chen, P.-F. and T. Tran (1994). Hyperspectral imagery classification using a backpropagation neural network. In Proceedings of IEEE International Conference on Neural Networks, Orlando, Florida, USA, pp. 2942–2947.

Chen, Q., Z. Guo, and J. Zhao (2008a). Identification of green tea's (Camellia Sinensis (l.)) quality level according to measurement of main catechins and caffeine contents by HPLC and support vector classification pattern recognition. Journal of Pharmaceutical and Biomedical Analysis 48(5), pp. 1321–1325.

Chen, Q., J. Zhao, and J. Cai (2008b). Identification of tea varieties using computer vision. Pattern Recognition Transactions of the American Society of Agricultural and Biological Engineers 51(2), pp. 965–976.

Chen, Q., J. Zhao, C. Fang, and D. Wang (2007). Feasibility study on identification of green, black and oolong teas using near-infrared reflectance spectroscopy based on support vector machine (SVM). Spectrochimica Acta 66(3), pp. 568–574.

Chen, Q., J. Zhao, and H. Lin (2009a). Study on discrimination of roast green tea (Camellia Sinensis I.) according to geographical origin by FT-NIR spectroscopy and supervised pattern recognition. Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy 72(4), pp. 845–850.

Chen, Q., J. Zhao, H. Zhang, and X. Wang (2006). Feasibility study on quantitative and quantitative analysis in tea by near infrared spectroscopy with multivariate calibration. Analytica Chimica Acta 572(1), pp. 77–84.

Chen, S. and D. Zhang (2011). Semisupervised dimensionality reduction with pairwise constraints for hyperspectral image classification. IEEE Geoscience and Remote Sensing Letters 8(2), pp. 369–373.

Chen, X.-J., H.-Q. Yang, D. Wu, and Y. He (2008c). A new method to discriminate tea categories. In Proceedings of the 7th World Congress on Intelligent Control and Automation, Chongqing, China, pp. 2236–2240.

Chen, Y. and E. Dougherty (1991). Queueing interpretation of adaptive reconstructive multiparameter τ-opening filters. Nonlinear Image Processing IX, 3304, pp. 59–65.

Chen, Y. and E. Dougherty (1994). Grey scale morphological granulometric texture classification. Optical Engineering 33(8), pp. 2713–2722.

Chen, Y., M. Yu, J. Xu, X. Chen, and J. Shi (2009b). Differentiation of eight tea (Camellia Sinensis) cultivars in China by elemental fingerprint of their leaves. Journal of the Science of Food and Agriculture 89(14), pp. 2350–2355.

Choi, K. and S. Kim (2005). Morphological analysis and classification of types of surface corrosion damage by digital image processing. Journal of Corrosion Science 47, pp. 1–15.

Chu, A., C. Sehgal, and J. Greenleaf (1990). Use of grey value distribution of run lengths for texture analysis. Pattern Recognition Letters 11(6), pp. 415–419.

Clausi, D. (2002). An analysis of co-occurrence texture statistics as a function of grey level quantization. Canadian Journal of Remote Sensing 28(1), pp. 45–62.

Codaro, E., R. Nakazato, A. Horovistiz, L. Ribeiro, R. Ribeiro, and L. Hein (2002). An image processing method for morphology characterization and pitting corrosion evaluation. Materials Science and Engineering A 334(1-2), pp. 298–306.

Cortes, C. and V. Vapnik (1995). Support-vector networks. Machine Learning 20(3), pp. 273–297.

Daubechies, I. (1988). Orthonormal bases of compactly supported wavelets. Communications on Pure and Applied Mathematics 41, pp. 909–996.

Daubechies, I. (1990). The wavelet transform, time-frequency localisation and signal analysis. IEEE Transactions on Information Theory 36, pp. 961–1005.

Daugman, J. (1980). Two dimensional spectral analysis of cortical receptive field profiles. Vision Research 20(10), pp. 847–856.

Daugman, J. (1985). Uncertainty relation for resolution in space, spatial frequency and orientation optimised by two-dimensional visual cortical filters. Journal of Optical Society of America 2(7), pp. 1160–1169.

de Almeida, C., R. de Souza, and A. Candeias (2010). Texture classification based on co-occurrence matrix and self-organizing map. In Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Istanbul, Turkey, pp. 2487–2491.

Debnath, L. (2002). Wavelet Transforms and their Application. Boston, USA: Birkhäuser.

Deguchi, K. (1986). Two-dimensional auto-regressive model for analysis and synthesis of grey-level textures. In Proceedings of the 1st International Symposium for Science on Form, KTK Scientific Publishers, Tokyo, Japan, pp. 441–449.

Demir, B. and S. Ertürk (2007). Hyperspectral image classification using relevance vector machines. IEEE Geoscience and Remote Sensing Letters 4(4), pp. 586–590.

Demir, B. and S. Ertürk (2011). Empirical mode decomposition of hyperspectral images for support vector machine classification. IEEE Transactions on Geoscience and Remote Sensing 48(11), pp. 4071–4084.

Dettori, L. and L. Semler (2007). A comparison of wavelet, ridgelet, and curvlet-based texture classification algorithms in computed tomography. Computers in Biology and Medicine 37(4), pp. 486–498.

Díaz, M., G. Ayala, R. Sebastian, and L. Martínez-Costa (2007). Granulometric analysis of corneal endothelium specular images by using a germ-grain model. Computers in Biology and Medicine 37(3), pp. 364–375.

Dougherty, E. and J. Astola (1994). An Introduction to Nonlinear Image Processing. Washington DC, USA: SPIE Press.

Dougherty, E. and R. Lotufo (2003). Hands-on Morphological Image Processing. Washington DC, USA: SPIE Press.

Dougherty, E., J. Newell, and J. Pelz (1992). Morphological texture-based maximum likelihood pixel classification based on local granulometric moments. Pattern Recognition 25(10), pp. 1181–1198.

Dougherty, E. and J. Pelz (1991). Morphological granulometric analysis of electrophotographic images-size distribution statistics for process control. Optical Engineering 30(4), pp. 438–445.

Drucker, H., W. Donghui, and V. Vapnik (2002). Support vector machines for

spam categorisation. IEEE Transactions on Neural Networks 10(5), pp. 1048–1054.

Du, Q. and C.-I. Chang (2001). Hidden Markov model approaches to hyperspectral image classification. In Proceedings of IEEE International Symposium on Geoscience and Remote Sensing, Sydney, Australia, pp. 2683–2685.

Dutta, R., E. Hines, J. Gardner, K. Kashwan, and M. Bhuyan (1994). Tea quality prediction using a tin oxide-based electronic nose: an artificial intelligence approach. Sensors and Actuators B 94, pp. 228–237.

Egmont-Peterson, M., D. de Ridder, and H. Handels (2002). Image processing with neural networks- a review. Pattern Recognition 35(10), pp. 2279.2301.

Eleyan, A., H. Ozkaramanli, and H. Demirel (2008). Complex wavelet transform-based face recognition. EURASIP Journal on Advances in Signal Processing 2008, pp. 123–151.

Everitt, B. and G. Dunn (2001). Applied Multivariate Data Analysis. London, UK: Hodder.

Fernández, A., O. Ghita, E. González, F. Bianconi, and P. F. Whelan (2011). Evaluation of robustness against rotation of LBP, CCR and ILBP features in granite texture classification. Machine Vision and Applications, 22(6), pp. 913-926.

Fernández-Cáceres, P., M. Martín, F. Pablos, and A. González (2001). Differentiation of tea (Camellia Sinensis) varieties and their geographical origin according to their metal content. Journal of Agriculture Food Chemistry 49(10), pp. 4775–4779.

Fogel, I. and D. Sagi (1989). Gabor filters as texture discriminators. Journal of Biological Cybernetics 61, pp. 103–113.

Fujita, O. (1998). Statistical estimation of the number of hidden units for feed-forward neural networks. Neural Networks 4(1), pp. 851–859.

Galloway, M. (1975). Texture analysis using gray level run lengths. Computer Graphics and Image Processing 4(2), pp. 172–179.

Gardiner, W. (2001). Statistics for the Biosciences. London, UK: Prentice Hall.

Geladi, P., H. Grahn, and J. Burger (2007). Multivariate images, hyperspectral imaging: Background and equipment. In H. Grahn and P. Geladi (Eds.), Tech-

niques and Applications of Hyperspectral Image Analysis. Chichester, UK: John Wiley and Sons.

Giardina, C. and E. Dougherty (1987). Morphological Methods in Image and Signal Processing. New Jersey, USA: Prentice Hall.

Gilchrist, J. and T. Hyvarinen (2006). Hyperspectral imaging spectroscopy: a look at real-life application. Photonics Handbook, H-140-H-144.

Glasbey, C. and G. Horgan (1994). Image Analysis for the Biological Sciences. Chichester, UK: John Wiley and Sons.

Gomez, F., J. Schmidhuber, and R. Miikkulainen (2008). Accelerated neural evolution through cooperatively coevolved synapses. Journal of Machine Learning Research 9, pp. 937-965.

Gong, P., J. Marceau, and P. Howarth (1992). A comparison of spatial feature extraction algorithms for land-use classification with spot HRV data. Remote Sensing of the Environment 40(2), pp. 137–151.

Gonzalez, R. and R. Woods (2008). Digital Image Processing, Third Edition. New Jersey, USA: Prentice Hall.

Goutsias, J. and S. Batman (2001). Morphological methods for boi-medical image analysis. In M. Sonka and J. Fitzpatrick (Eds.), The Handbook of Medical Imaging, pp. 175–263. Washington DC, USA: SPIE Press.

Goutsias, J., H. Heijmans, and K. Sivakumar (1995). Morphological operators for image sequences. Computer Vision and Image Understanding 62(3), pp. 326–346.

Graves, A., M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber (2009). A novel connectionist system for unconstrained handwriting recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(5), pp. 855–868.

Gray, A., S. Marshall, and J. McKenzie (2006). Modeling of evolving textures using granulometries. In S. Marshall and G. Sicuranza (Eds.), Advances in Nonlinear Signal and Image Processing, EURASIP Series on Signal Processing and Communications. New York, USA: Hindawi Publishing Corporation.

Gray, A., J. McKenzie, and S. Marshall (2005). Texture classification of grey scale corrosion images. In Proceedings of IEE International Conference on Visual Information Engineering (VIE 2005), Glasgow, UK, pp. 219–225.

Grossmann, A. and J. Morlet (1984). Decomposition of Hardy functions into square integrable wavelets of constant shape. SIAM Journal of Mathematical Analysis 15(4), pp. 723–736.

Guo, B., S. Gunn, R. Damper, and J. Nelson (2008). Customizing kernel functions for SVM-based hyperspectral image classification. IEEE Transactions on Image Processing 17(4), pp. 622–629.

Haddon, J. and J. Boyce (1993). Co-occurrence matrices for image analysis. Electronics and Communications Engineering Journal 5(2), pp. 71–83.

Haley, G. and B. Manjunath (1995). Rotation invariant texture classification using modified Gabor filters. In Proceedings of IEEE International Conference on Image Processing, Washington, USA, pp. 262–265.

Hand, D. (1981). Discrimination and Classification. Chichester, UK: John Wiley and Sons Ltd.

Haq, Q., L. Shi, T. Tao, and S. Yang (2010). A robust band compression technique for hyperspectral image classification. In Proceedings of IEEE International Conference on Intelligent Computing and Intelligent Systems, Xiamen, Chaina, pp. 196–200.

Haralick, R. (1979). Statistical and structural approaches to texture. Proceedings of the IEEE 67(5), pp. 786–804.

Haralick, R., K. Shanmugan, and I. Dinstein (1973). Textural features for image classification. IEEE Transactions on Systems, Man and Cybernetics 3(6), pp. 610–621.

Harrison, L., P. Dastidar, H. Eskola, R. Jarvenpaa, H. Pertovaara, T. Luukkaala, P. Kellokumpu, and S. Soimakallio (2008). Texture analysis on MRI images of non-Hodgkin lymphoma. Computers in Biology and Medicine 38(4), pp. 519–524.

Harrison, R., F. Bianconi, R. Harvey, and W. Wang (2011). A texture analysis approach to identifying Sabellaria Spinulosa colonies in sidescan sonar imagery. In Proceedings of IMVIP 2011, Irish Machine Vision and Image Processing Conference, Dublin, Ireland, UK, pp. 64–69.

Haykin, S. (1998). Neural Networks - A Comprehensive Foundation. Englewood Cliffs, New Jersey, USA: Prentice Hall.

Heijmans, H. (1979). Mathematical morphology: a geometrical approach to im-

age processing. Nieuw Archief voor Wiskunde 10(3), pp. 237–276.

Herrador, M. and A. González (2001). Pattern recognition procedures for differentiation of green, black and oolong teas according to their metal content from inductively coupled plasma atomic emission spectrometry. Talanta 53(6), pp. 1249–1257.

Ho, Y.-X., M. Landy, and L. Maloney (2006). How direction of illumination affects visually perceived surface roughness. Journal of Vision 6(5), pp. 634–648.

Hollmén, J., V. Tresp, and O. Simula (2000). A learning vector quantisation algorithm for probabilistic model. In Proceedings of 10th European Signal Processing Conference (EUSIPCO 2000), Tampere, Finland, pp. 721–724.

Horn, R. and C. Johnson (1994). Topics in Matrix Analysis. Cambridge, UK: Cambridge University Press.

Hsu, P.-H. and H.-H. Yang (2007). Hyperspectral image classification using wavelet networks. In Proceedings of IEEE International Symposium on Geoscience and Remote Sensing, Barcelona, Spain, pp. 1767–1770.

Huang, X. and L. Zhang (2008). An adaptive mean-shift analysis approach for object extraction and classification from urban hyperspectral imagery. IEEE Transactions on Geoscience and Remote Sensing 46(12), 4173–4185.

Hwang, W., H. Wang, H. Kim, S.-C. Kee, and J. Kim (2011). Face recognition system using multiple face model of hybrid Fourier feature under uncontrolled illumination variation. IEEE Transactions on Image Processing 20(4), pp. 1152–1165.

Izenman, A. (2008). Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning. Philadelphia, USA: Springer.

Jafari-Khouzani, K. and H. Soltanian-Zadeh (2005). Rotation-invariant multiresolution texture analysis using Radon and wavelet transforms. IEEE Transactions on Image Processing 14(6), 783–795.

Jain, A. (1981). Advances in mathematical models for image processing. In Proceedings of the IEEE 69(5), pp. 502–528.

Jain, A. (1989). Fundamentals of Digital Image Processing. Englewood Cliffs, New Jersey, USA: Prentice Hall Information and System Sciences Series.

Jain, A. K., R. P. W. Duin, and J. Mao (2000). Statistical pattern recognition: a

review. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(1), pp. 4–34.

Jain, A. and F. Farrokhnia (1990). Unsupervised texture segmentation using Gabor filters. In Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Los Angeles, CA, USA, pp. 14–19.

Jensen, A., A. Berge, and A. Solberg (2008). Regression approaches to small sample inverse covariance matrix estimation for hyperspectral image classification. In Proceedings of IEEE International Geoscience and Remote Sensing Symposium, Barcelona, Spain, pp. 3781–3784.

Jia, S., Z. Ji, Z. Zhu, and Y. Qian (2010a). Feature selection technique for hyperspectral imagery classification with noise reduction preprocessing. In Proceedings of Chinese Conference on Pattern Recognition, Chongqing, China, pp. 1–5.

Jia, S., Y. Qian, J. Li, W. Liu, and Z. Ji (2010b). Feature extraction and selection hybrid algorithm for hyperspectral imagery classification. In Proceedings of IEEE International Symposium on Geoscience and Remote Sensing, Honolulu, Hawaii, USA, pp. 72–75.

Jing, X., L. Liu, S. Li, Y. Yao, L. Bian, Q. Liu, Y. Dong, and Z. Sui (2009). A novel discriminant analysis approach using angular Fourier transform for face recognition. In Proceedings of the 3rd International Symposium on Intelligent Information Technology Application, Nanchang, China, pp. 117–120.

Karahaliou, A., I. Boniatis, S. Skiadopoulos, F. Sakellaropoulos, N. Arikidis, E. Likaki, G. Panayiotakis, and L. Costaridou (2008). Breast cancer diagnosis: analyzing texture of tissue surrounding microcalcifications. IEEE Transactions on Information Technology in Biomedicine 12(6), pp. 731–738.

Karatzoglou, A., D. Meyer, and K. Hornik (2006). Support vector machines in R. Journal of Statistical Software 15(9), pp. 10–24.

Kelman, T., J. Ren, and S. Marshall (2011). Classification of Chinese tea samples for food quality control using hyperspectral imaging. In Proceedings of the 2nd Hyperspectral Imaging Conference (HSI 2011), May 2011, Glasgow, UK.

Kim, H.-C., S. Pang, H.-M. Je, D. Kim, and Y. B. Sung (2003). Constructing support vector machine ensemble. Pattern Recognition 36(12), pp. 2757–2767.

Kim, I., K. Jung, S. Park, and J. Hang (2002). Support vector machines for texture classification. IEEE Transactions on Pattern Analysis and Machine In-

telligence 24(11), pp. 1542–1550.

Kim, K., K. Jung, S. Park, and H. Kim (2002). Support vector machines for texture classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(11), pp. 1542–1550.

Kim, W., M. Crawford, and J. Ghosh (2008). Spatially adapted manifold learning for classification of hyperspectral imagery with insufficient labeled data. In Proceedings of IEEE International Symposium on Geoscience and Remote Sensing, Boston, MA, USA, pp. I-213–I-216.

Kimmel, R. and A. Bruckstein (2003). Regularized Laplacian zero crossings as optimal edge integrators. International Journal of Computer Vision 53(3), pp. 225–243.

Kingsbury, N. (1999). Image processing with complex wavelets. Philosophical Transactions of the Royal Society of London A 357, pp. 2543–2560.

Kingsbury, N. (2001). Complex wavelets for shift invariant analysis and filtering of signals. Journal of Applied and Computational Harmonic Analysis 10(3), pp. 234–253.

Kingsbury, N. (2005). The dual-tree complex wavelet transform. IEEE Signal Processing Magazine 22(6), pp. 123–151.

Kittler, J. and J. Illingworth (1986). Minimum error thresholding. Pattern Recognition 19(1), pp. 41–47.

Krishnamachari, S. and R. Chellappa (1997). Multiresolution Gauss-Markov random field models for texture segmentation. IEEE Transactions on Image Processing 6(2), pp. 251–267.

Kumar, V. and E. Manolakos (1997). Unsupervised statistical neural networks for model-based objects recognition. IEEE Transactions on Signal Processing 45(11), pp. 2709–2718.

Kuo, B.-C., C.-S. Huang, C.-C. Hung, Y.-L. Liu, and I.-L. Chen (2010). Spatial information based support vector machine for hyperspectral image classification. In Proceedings of IEEE International Symposium on Geoscience and Remote Sensing Symposium, Honolulu, Hawii, USA, pp. 832–835.

Kurita, T. and N. Otsu (1993). Texture classification by higher order local autocorrelation features. In Proceedings of the Asian Conference on Computer Vision,

Umezono, Tsukuba, Japan, pp. 175–178.

Kyriacou, E., M. Pattichis, C. Pattichis, A. Mavrommatis, C. Christodoulou, S. Kakkos, and A. Nicolaides (2009). Classification of atherosclerotic carotid plaques using morphological analysis on ultrasound images. Journal of Applied Intelligence 30(1), pp. 3–23.

Lehmann, F. (2011). Turbo segmentation of textured images. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(1), pp. 16–29.

Levkowitz, H. and Herman, G.T. (1993). GLHS: A generalised lightness, hue and saturation color model. Graphical Models and Image Processing 55(4), pp. 271–285.

Li, X.-L., P. Nie, Z.-J. Qiu, and Y. He (2011). Using wavelet transform and multi-class least square support vector machine in multi-spectral imaging classification of Chinese famous tea. Journal of Expert Systems with Applications 38(9), pp. 11149–11159.

Li, J., L. Chen, and Y. Cai (2009). Dynamic texture segmentation using 3-D Fourier transform. In Proceedings of 5th IEEE International Conference on Image and Graphics, Shanxi, China, pp. 293–298.

Li, T.-S. (2009). Applying wavelets transform and support vector machine for copper clad laminate defects classification. Journal of Computers and Industrial Engineering 56(3), pp. 1154–1168.

Li, X. and Y. He (2008a). Discriminating varieties of tea plant based on VIS/NIR spectral characteristics and using artificial neural networks. Biosystems Engineering 99(3), pp. 313–321.

Li, X.-L., Y. He, and Z.-J. Qiu (2008b). Textural feature extraction and optimization in wavelet sub-bands for discrimination of green tea brands. In Proceedings of the 7th International Conference on Machine Learning and Cybernetics, Kunming, China, pp. 1461–1466.

Li, W.W. Gong, Y. and W. Chen (2005). Feature selection based on KPCA, SVM and GSFS for face recognition. In S. Singh, M. Singh, C. Apte, and P. Perner (Eds.), Pattern Recognition and Image Analysis, Heidelberg, Berlin, Springer-Verlag, pp. 344–350.

Li, S., J. Kwok, H. Zhu, and Y. Wang (2003). Texture classification using the support vector machines. Pattern Recognition 36(12), pp. 2883–2893.

Liao, S. and A. Chung (2010). A new subspace learning method in Fourier domain for texture classification. In Proceedings of 17th IEEE International Conference on Image Processing (ICIP 2010), Hong Kong, China, pp. 4589–4592.

Liguo, W., D. Luqun, and L. Ming (2009). Hyperspectral imagery classification aiming at protecting classes of interest. In Proceedings of the World Congress on Computer Science and Information Engineering, Washington DC, USA, pp. 144–147.

Liu, S., S. Yung, and S.-M. Andi (2010). Classifying the variety production area and season of Taiwan partially fermented tea by near infrared spectroscopy. Journal of Food and Drug Analysis 18(1), pp. 34–43.

Livens, S., P. Scheunders, G. van de Wouwer, and D. Van Dyck (1997). Wavelets for texture analysis, an overview. In Proceedings of 6th IEE Conference on Image Processing and its Applications (IPA 1997), Dublin, Ireland, UK, pp. 581–585.

Ma, L., C. Cheng, X. Liu, Y. Zhao, A. Wang, and P. Herdewijn (2004). A neural network for predicting the stability of RNA/DNA hybrid duplexes. Chemometrics and Intelligent Laboratory System 70(2), pp. 123–128.

Macleod, M. (1992). Non-linear recursive smoothing filters and their use for noise floor estimation. Electronics Letters 28(12), pp. 1952–1953.

Mahalanobis, P. C. (1936). On the generalised distance in statistics. In Proceedings of the National Institute of Sciences of India 2(1), pp. 49–55.

Mai, M.-Y. and L.-L. Wang (2008). Automatic shoe-pattern boundary extraction by image-processing techniques. Journal of Robotics and Computer Integrated Manufacturing 24(2), pp. 217–227.

Mallat, S. (1999). A Wavelet Tour of Signal Processing, 2nd Edition. San Diego, USA: Academic Press.

Maragos, P. (1987). Pattern spectrum of images and morphological shape-size complexity. In Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1987), Dallas, Texas, USA, pp. 241–244.

Masotti, M. and R. Campanini (2008). Texture classification using invariant ranklet features. Pattern Recognition Letters 29(14), pp. 1980–1986.

Matheron, G. (1975). Random Sets and Integral Geometry. New York, USA:

John Wiley and Sons Ltd.

Mavilio, A., M. Fernández, M. Trivi, H. Rabal, and R. Arizaga (2010). Characterization of a paint drying process through granulometric analysis of speckle dynamic patterns. Signal Processing 90(5), pp. 1623–1630.

Mazanec, J., M. Melišek, M. Oravec, and J. Pavlovičová. (2008). Support vector machines, PCA and LDA in face recognition. Journal of Electrical Engineering 59(4), pp. 203–209.

McKenzie, J. (2004). Classification of dynamically evolving textures using evolution functions. PhD. Thesis, Department of Electronic and Electrical Engineering, University of Strathclyde.

McKenzie, J., J. Jurado, and F. de Pablos (2010). Characterisation of tea leaves according to their total mineral content by means of probabilistic neural networks. Journal of Food Chemistry 123(3), pp. 859–864.

McKenzie, J., S. Marshall, A. Gray, and E. Dougherty (2003). Morphological texture analysis using the texture evolution function. International Journal of Pattern Recognition and Artificial Intelligence, special issue on Quantitative Image Morphology 17(2), pp. 167–185.

Mengko, T. and J. Pramudito (2002). Implementation of Gabor filter to texture analysis of radiographs in the assessment of osteoporosis. In Proceedings of Asia-Pacific Conference on Circuits and Systems, Bali, Indonesia, pp. 251–254.

Mital, D. and G. Leng (1992). Autoregressive approach to surface texture analysis. In Proceedings of International Conference on Industrial Electronics, Control, Instrumentation and Automation, Volume 3, San Diego, USA, pp. 1309–1312.

Moreda-Piñeiro, A., A. Fisher, and S. Hill (2003). The classification of tea according to region of origin using pattern recognition techniques and trace metal data. Journal of Food Composition and Analysis 16(2), pp. 195–211.

Moshou, D., C. Bravo, R. Oberti, J. West, L. Bodria, A. McCartney, and H. Ramon (2005). Plant disease detection based on data fusion of hyperspectral and multi-spectral fluorescence imaging using Kohonen maps. Real-Time Imaging 11(2), pp. 75–83.

Nidamanuri, R. and B. Zbell (2011). Normalized spectral similarity score (NS^3) as an efficient spectral library searching method for hyperspectral image classification. IEEE Journal of Selected Topics in Applied Earth Observations and

Remote Sensing 4(1), pp. 226–240.

Nixon, M. and A. Aguado (2002). Feature Extraction and Image Processing. Oxford, UK: Newnes Press.

Ojala, T., M. Pietikäinen, and D. Harwood (1996). A comparative study of texture measures with classification based on features distribution. Pattern Recognition 29(1), pp. 51–59.

Ojala, T., M. Pietikäinen, and T. Maenpaa (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), pp. 971–987.

Otsu, N. (1979). A threshold selection method from gray level histograms. IEEE Transactions on Systems, Man and Cybernetics 9(1), pp. 62–66.

Paget, R. and I. Longstaff (1998). Texture synthesis via a noncausal nonparametric multiscale Markov random field. IEEE Transactions on Image Processing 7(6), pp. 925–931.

Palaniappan, R. and N.-J. Huan (2005). Improving the performance of two state mental task brain-computer interface design using linear discriminant classifier. In Proceedings of the International Conference on Computer as a Tool (EUROCON 2005), Belgrade, Serbia, pp. 409–412.

Palm, C. (2004). Color texture classification by integrative co-occurrence matrices. Pattern Recognition 37(5), pp. 965–976.

Pasternack, R. M., Z. Qian, J-Y. Zheng, D. N. Metaxas, E. White, and N. N. Boustany (2009). Measurement of subcellular texture by optical Gabor-like filtering with a digital micromirror device. Optics Letters 33(13), pp. 1939–1939.

Petrou, M. and P. García-Sevilla (2006). Image Processing Dealing with Texture. Chichester, UK: John Wiley and Sons.

Pidaparti, R., B. Aghazadeh, A. Whitfield, A. Rao, and G. Mercier (2010). Classification of corrosion defects in NiAl bronze through image analysis. Corrosion Science 52(10), pp. 3661–3666.

Plaza, A., P. Martínez, J. Plaza, and R. Pérez (2005). Dimensionality reduction and classification of hyperspectral image data using sequences of extended morphological transformation. IEEE Transactions on Geoscience and Remote Sensing 43(3), pp. 466–479.

Pontil, M. and A. Verri (1998). Support vector machines for 3D object recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(6), pp. 637–646.

Prasad, S., H. Kalluri, L. Bruce, and S. Samiappan (2010). Data dependent adaptation for improved classification of hyper-spectral imagery. In Proceedings of IEEE International Geoscience and Remote Sensing Symposium, Honolulu, Hawaii, USA, pp. 68–71.

Rabe, A., S. van der Linden, and P. Hostert (2010). Simplifying support vector machines for classification of hyperspectral imagery and selection of relevant features. In Proceedings of 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, Reykjavik, Iceland, pp. 1–4.

Ratle, F., G. Camps-Valls, and J. Weston (2010). Semisupervised neural networks for efficient hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing 48(5), pp. 2271–2282.

Rawlings, J. (1932). Applied Regression Analysis : a Research Tool. California, USA: Pacific Grove.

Rellier, G., X. Descombes, F. Falzon, and J. Zerubia (2004). Texture feature analysis using a Gauss-Markov model in hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing 42(7), pp. 1543–1551.

Ren, J., T. Kelman, and S. Marshall (2011). Adaptive clustering of spectral components for band selection in hyperspectral imagery. In Proceedings of the 2nd Hyperspectral Imaging Conference (HSI 2011), May 2011, Glasgow, UK.

Reulke, R. and A. Lippok (2008). Markov random fields-based texture segmentation for road detection. Proceedungs of the 21st Congress, Beijing, China, Volume XXXVII, PartB3b, Commission III, International Society for Photogrammetry and Remote Sensing, 2008, pp. 615–620.

Ridler, T. and S. Calvard (1978). Picture thresholding using an iterative selection method. IEEE Transactions on Systems, Man and Cybernetics 8(8), pp. 630–632.

Rumelhart, D., G. Hinton, and R. Williams (1986). Learning internal representations by error propagation. Parallel Distributed Processing: Explorations in the Microstructure of Cognition 1, pp. 318–362.

Salari, E., and Ling, Z. (1995). Texture segmentation using hierarchical wavelet decomposition. Pattern Recognition, 28(12), pp. 1819–1824.

Schmidhuber, J., D. Wierstra, M. Gagliolo, and F. Gomez (2007). Training recurrent networks by EVOLINO. Neural Computation 19(3), pp. 757–779.

Serpico, S., M. Dínca, and G. Moser (2004). Design of spectral channels for hyperspectral image classification. In Proceedings of IEEE International Symposium on Geoscience and Remote Sensing, New Jersey, USA, pp. 956–959.

Serpico, S. and G. Moser (2007). Extraction of spectral channels from hyperspectral images for classification purposes. IEEE Transactions on Geoscience and Remote Sensing 45(2), pp. 484–495.

Serra, J. (1983). Image Analysis and Mathemetical Morphology. London, UK: Academic Press.

Shapiro, L. G. and G. Stockman (2002). Computer Vision. New Jersey, USA: Prentice Hall.

Shannon, C. (1948). A mathematical theory of communication. The Bell System Technical Journal 27, pp. 379–423.

Sharma, M. and S. Singh (2001). Evaluation of texture methods for image analysis. In Proceedings of the 7th Australian and New Zealand Intelligent Information Systems Conference, Perth, Western Australia, pp. 117–121.

Shigeo, A. (2010). Support Vector Machines for Pattern Classification. London, UK: Springer.

Shuttleworth, D., A. Todman, R. Naguib, B. Newman, and M. Bennett (2002). Colour texture analysis using co-occurrence matrices for classification of colon cancer images. In Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering, Winnipeg, Canada, pp. 1134–1139.

Soh, L.-K. and C. Tsatsoulis (1999). Texture analysis of SAR sea ice imagery using gray level co-occurrence matrices. IEEE Transactions on Geoscience and Remote Sensing 37(2), pp. 780.795.

Soh, L.-K. and C. Tsatsoulis, D. Gineris, and C. Bertoia (2004). An intelligent system for SAR sea ice image classification. IEEE Transactions on Geoscience and Remote Sensing 42(1), pp. 229-248.

Subramanian, S., N. Gat, M. Sheffield, J. Barhen, and N. Toomarian (1997). Methodology for hyperspectral image classification using novel neural network. In Proceedings of SPIE Conference on Algorithms for Multispectral and Hyper-

spectral Imagery III, Orlando, Florida, USA, SPIE vol. 3071, pp. 128–137.

Sulehria, H., Y. Zhang, and D. Irfan (2007). Mathematical morphology methodology for extraction of vehicle number plates. International Journal of Computers 1(3), pp. 364–375.

Suykens, J. and J. Vandewalle (1999). Least squares support vector machine classifiers. Neural Processing Letters 9(3), pp. 293–300.

Swain, M. and D. Ballard (1991). Color indexing. International Journal of Computer Vision 7(1), pp. 11–32.

Tan, K. and P. Du (2010). Classification of hyperspectral image based on morphological profiles and multi-kernel SVM. In Proceedings of 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, Reykjavik, Iceland, pp. 1–4.

Tang, X. (1998). Texture information in run-length matrices. IEEE Transactions on Image Processing 7(11), pp. 1602–1609.

Thakoor, N., J. Gao, and S. Jung (2007). Hidden Markov model-based weighted likelihood discriminant for 2-D shape classification. IEEE Transactions on Image Processing 16(11), pp. 2707–2719.

Theera-Umpon, N. and S. Dhompongsa (2007). Morphological granulometric features of nucleus in automatic bone marrow white blood cell classification. IEEE Transactions on Information Technology in Biomedicine 11(3), pp. 353–359.

Trussell, H. (1979). Comments on 'picture thresholding using an iterative selection method'. IEEE Transactions on Systems, Man and Cybernetics 9(5), pp. 311.

Tsiaparas, N., S. Golemati, I. Andreadis, J. Stoitsis, I. Valavanis, and K. Nikita (2011). Comparison of multiresolution features for texture classification of carotid atherosclerosis from b-mode ultrasound. IEEE Transactions on Information Technology in Biomedicine 15(1), pp. 130–137.

Tuceryan, M. (1994). Moment based texture segmentation. Pattern Recognition Letters 15, pp. 659–668.

Tuceryan, M. and A. Jain (1998). Texture analysis. In C. Chen, L. Pau, and P. Wang (Eds.), The Handbook of Pattern Recognition and Computer Vision (2nd Edition), New Jersey, USA: World Scientific Publishing Co. Ltd., pp. 207–248.

Tuma, J. and R. Walsh (1998). Engineering Mathematics Handbook, 4th Edition. New York, USA: McGraw-Hill Publishing Company.

Unser, M. (1995). Texture classification and segmentation using wavelet frames. IEEE Transactions on Image Processing 4(11), pp. 1549–1560.

Valera, P., F. Pablos, and A. González (1996). Classification of tea samples by their chemical composition using discriminant analysis. Talanta 43(3), pp. 415–419.

Varma, M. and A. Zisserman (2005). A statistical approach to texture classification from single images. International Journal of Computer Vision 62(1-2), 61–81.

Velten, K. (2009). Mathematical Modeling and Simulation: Introduction for Scientists and Engineers. Geisenheim, Germany: Wiley-VCS.

Vincent, L. (1993). Morphological greyscale reconstruction in image analysis: application and efficient algorithm. IEEE Transactions on Image Processing 2(2), pp. 176–201.

Wang, L. and D.-C. He (1990). Texture classification using texture spectrum. Pattern Recognition 23(8), pp. 905–910.

Wang, Z.-Z. and J.-H. Yong (2008). Texture analysis and classification with linear regression model based on wavelet transform. IEEE Transactions on Image Processing 17(8), pp. 1421–1430.

Webb, A. (2002). Statistical Pattern Recognition. Chichester, UK: John Wiley and Sons.

Wells, W. (1986). Efficient synthesis of Gaussian filters by cascaded uniform filters. IEEE Transactions on Pattern Analysis and Machine Intelligence 8(2), pp. 234–239.

Weston, J. and C. Watkins (1998). Multi-class support vector machines. Technical report, Dept. of Computer Science, Royal Holloway, University of London, UK, Technical Report CSD-TR-98-04.

Wu, D., X. Chen, and Y. He (2007). Application of image texture for discrimination of tea categories using multi-spectral imaging technique and support vector machine. In Proceedings of the 2007 International Conference on Computational Intelligence and Security Workshops, Washington, DC, USA, pp. 291–294.

Wu, D., H. Yang, X. Chen, Y. He, and X. Li (2008). Application of image texture for the sorting of tea categories using multi-spectral imaging technique and support vector machine. Journal of Food Engineering 88(4), pp. 474–483.

Wu, J. and A. Chung (2007). A segmentation model using compound Markov random fields based on a boundary model. IEEE Transactions on Image Processing 16(1), pp. 241–252.

Xiao, S., R. Zhai, and Y. Wu (2007). Rotation-invariant texture characterization using Hough and Fourier transforms. In Proceedings of 1st International Conference on Bioinformatics and Biomedical Engineering, Wuhan, China, pp. 841–844.

Xiao, S.-S. and Y.-X. Wu (2007). Rotation-invariant texture analysis using Radon and Fourier transforms. Journal of Physics Conference Series 48(9), pp. 1459–1464.

Yan, Y., Y. Zhao, H.-F. Xue, X.-D. Kou, and Y. Liu (2010). Integration of spatial-spectral information for hyperspectral image classification. In Proceedings of 2nd IITA International Conference on Geoscience and Remote Sensing, Qingdao, China, pp. 242–245.

Yang, S., W. Cui, and H. Wang (2009). Gabor transform application in feature description of underwater echo signal. In Proceedings of 1st International Conference on Information Science and Engineering, Nanjing, China, pp. 566–569.

Yang, S. and R. Lunetta (2008). The use of MODIS-NDVI data for mapping cropland across the Great Lakes Basin, USA. In Proceedings of IEEE International Geoscience and Remote Sensing Symposium, Boston, USA, Volume 5, pp. 196–199.

Yin, J., C. Gao, Y. Wang, and Y. Wang (2010). Hyperspectral image classification using wavelet packet analysis and gray prediction model. In Proceedings of International Conference on Image Analysis and Signal Processing, Zhejiang, China, pp. 322–326.

Zenoozian, M. and S. Devahastin (2009). Application of wavelet transform coupled with artificial neural network for predicting physicochemical properties of osmotically dehydrated pumpkin. Journal of Food Engineering 90(2), pp. 219–227.

Zhang, L., Q. Liu, C. Zhao, H. Lin, and H. Sun (2010). The detailed vegetation

classification for airborne hyperspectral remote sensing imagery by combining PCA and PP. In Proceedings of the 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, Reykjavik, Iceland, pp. 1–4.

Zhao, J., Q. Chen, J. Cai, and Q. Quyang (2009). Automated tea quality classification by hyper-spectral imaging. Journal of Applied Optics 48(19), pp. 3557–3564.

Zhao, J., Q. Chen, X. Huang, and C. Fang (2006). Qualitative identification of tea categories by near infrared spectroscopy and support vector machine. Journal of Pharmaceutical and Biomedical Analysis 41(4), pp. 1198–1204.

Zhao, Y.-Q., L. Zhang, and S. Kong (2011). Band-subset-based clustering and fusion for hyperspectral imagery classification. IEEE Transactions on Geoscience and Remote Sensing 49(2), pp. 747–756.

Zheng, W. (1997). Estimation of autoregressive signals from noisy measurements. IEE Proceedings on Vision, Image and Signal Processing 144(1), pp. 39–45.

Zheng, Y., Y. Meng, and Y. Jin (2010). Fusing bottom-up and top-down pathways in neural networks for visual object recognition. In Proceedings of the International Joint Conference on Neural Networks, Barcelona, Spain, pp. 1–8.

Zhong, P. and R. Wang (2008). Learning sparse CRFS for feature selection and classification of hyperspectral imagery. IEEE Transactions on Geoscience and Remote Sensing 46(12), pp. 4186–4197.

Zhu, Z., Z. Ji, and S. Jia (2010). Memetic ant colony optimization for band selection of hyperspectral imagery classification. In Proceedings of the Chinese Conference on Pattern Recognition, Chongqing, China, pp. 1–6.

Zingman, I., R. Meir, and R. El-Yaniv (2007). Size-density spectra and their application to image classification. Pattern Recognition 40(12), pp. 3336–3348.