Wavelet Based Image Texture Segmentation using a Modified K-means Algorithm

by

Brian W. Ng

B.E. (First Class Honours), B.Sc. (Ma. & Comp. Sc.)

Thesis submitted for the degree of

Doctor of Philosophy

in

Department of Electrical and Electronic Engineering,
Faculty of Engineering,
Computer and Mathematical Sciences

University of Adelaide, Australia

August, 2003

Typeset in LaTeX 2_ε
Brian W. Ng

Contents

Abstract

In the field of computer vision, texture analysis has historically been an important topic of study. Texture is one of the major primary visual cues to image understanding. The human visual system relies on texture, among other types of cues, to effectively interpret the information contained in an image. Therefore, it is imperative that a fully functional artificial vision system must perform texture analysis. However, effective texture analysis has proved to be a difficult problem. This is mainly due to fundamental difficulties with texture - it is more of a concept than a well-defined object or property. Without an unambiguous definition of texture, researchers have to resort to various models and techniques, all of which have their own advantages and disadvantages. Traditionally, there have been three main approaches to texture analysis: statistical, structural and filter-based. While statistical and structural approaches have been favoured by early researchers, more modern techniques are typically filter-based. This thesis presents a type of filter-based technique. There are two main areas within texture analysis: classification and segmentation, but this thesis only covers the segmentation problem.

Wavelet transforms are a relatively new analytical tool in the scientific community. It is commonly accepted that wavelet theory grew out of classic Fourier analysis. Despite its widespread usefulness, Fourier analysis has several deficiencies; wavelet theory is among many others that attempt to address such problems. Primarily, wavelet transforms decompose signals into joint time-frequency bases, instead of harmonic ones. The inherent multiresolution property associated with joint time-frequency representations makes wavelet transforms far more suitable for analysing non-stationary signals. As a result,

wavelets have been used in many different applications in the relatively short period of time since its introduction.

In this thesis, wavelet transforms are chosen as the primary analytical tool for texture analysis. In particular, a recent advancement in wavelet transforms, called the Dual-Tree Complex Wavelet Transform, is applied to the texture segmentation problem. While the wavelet transform is believed to be a suitable analytical tool, there are other problems to be overcome before a texture segmentation system can be built. Specifically, the feature extraction and feature clustering methods need to be investigated. This thesis examines several possibilities for feature extraction and clustering steps. In particular, novel feature extraction and clustering schemes are introduced and compared to other known techniques. An extensive range of experiments are performed on the texture mosaics, to verify the effectiveness of the proposed feature extraction and clustering methods. The set of inputs to the experiments are carefully chosen as to allow direct comparison with existing methods, so a meaningful indication of the quality of the proposed segmentation system can be obtained. It has been found that the proposed system is capable of producing accurate, highly consistent segmentations on the test mosaics. With the success on the test mosaics, the segmentation system is then applied to several real-world applications. In the real examples, it is found that the system produces more erratic results, often as a consequence of the lack of any problem-specific input to assist the segmentation process. Overall, the proposed system, using wavelet based features, compares well with existing schemes for basic texture segmentation tasks.

Statement of Originality

This work contains no material that has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I give consent to this copy of the thesis, when deposited in the University Library, being available for loan and photocopying.

Signed

Date

This page is blank

Acknowledgements

I would like to take this opportunity to express my sincerest gratitude to all those who have assisted me throughout the last few years. It is the many mentors, friends and colleagues who have not only made my Ph.D. research possible, but also tremendously rewarding and enjoyable.

Firstly, I would like to thank my principal supervisor and respected teacher, Associate Professor Abdesselam (Salim) Bouzerdoum (Edith Cowan University, Australia). Completion of this thesis would not have been possible without his guidance and he has proved to be a continual source of inspiration and encouragement. I owe a lot to his dedication, which has made the difficult task of remote supervision seem easy. The time I spent visiting the VIP research centre at Edith Cowan University has provided me with many fond memories.

I would also like to thank Professor Neil Burgess (Cardiff University, Wales), who co-supervised my research during the early years of my Ph.D. research. His friendly encouragement eased me into the challenges of postgraduate study. He was first rate company at "beer o'clock" on many Friday afternoons. Many thanks also go to Dr. Julian Magarey (Canon Australia) and Dr. David Crisp (DSTO) for providing many ideas for my research work. I was also fortunate to have had useful communications with Professor Bob Bogner (University of Adelaide), whose enthusiasm and energy beyond the call of duty have been inspirational.

During my time at the Department of Electrical and Electronic Engineering, I have made many friends, from fellow postgraduates to staff and undergraduates. Their friendship and support greatly enriched my time as a postgraduate student. I have enjoyed tremendously the many activities in which we took part together, from the regular coffee breaks to those sneaky summer rounds of golf.

Finally, I would like to thank my family and personal friends. The most thanks must go to my parents, for being exceptional role models and an endless fountain of support. Thanks also go to my sister, and a supportive network of extended family in Hong Kong. Further kudos go to many friends for their care and concern. Last of all, thank you, Fiona, for being a constant source of love and encouragement.

Brian

List of Figures

This page is blank

List of Tables

Chapter 1

Introduction

1.1 Texture Analysis and Computer Vision

Texture analysis has been an important field in computer vision over the years. Primarily, texture is one of the major visual cues to understanding the information content in an image. Textural processing, along with other techniques such as edge localisation, are classified as low-level vision processing. Such processing techniques focus on extracting information from the raw input to the visual system. They serve as a basis for higher levels of vision processing, which focus on interpretation and abstraction on objects and their attributes. Texture analysis is therefore a fundamental problem in machine vision.

1.1.1 Texture Analysis

Texture in an image generally refers to a region which not only has smoothly varying intensity in a local sense, but also exhibits some 'pattern', or regularity, on a global scale. To successfully characterise textures, it is equally important to describe both their local (loosely referred to as *micro-textures*) and global (*macro-textures*) properties. In many instances, the distinction between micro- and macro-textures is fuzzy, and often differs from one texture to another. In the context of this thesis, the term texture will refer to digitised, gray-tone images containing predominantly visual textural information as perceived by a human observer.

The topic of texture analysis can be loosely divided into two main problems: classification and segmentation. Texture classification problems deal with the ability of an artificial system to effectively discriminate between different textures. Texture segmentation studies algorithms which divide a given image into distinct regions based on textural information. The primary scope of this thesis is texture segmentation, and classification will not be discussed further in this thesis. However, it should be pointed out that these two problems share many attributes in common, and techniques developed for segmentation may also be suitable for classification, and vice versa.

1.1.2 Texture Segmentation

While texture segmentation has been researched for decades, it is still an important area for further investigation. There are several reasons behind this. Despite the grand collective efforts of researchers from multiple disciplines, texture is still a relatively poorly understood phenomenon. The inherent ambiguity of its very concept is the chief reason for this difficulty. While it may be very easy, even natural, for humans to understand the concept of texture, it is extremely difficult to unambiguously define texture in a logical description. Many researchers have attempted to define texture, but none of these efforts produced a fully satisfactory result. It may very well be possible that such a universal definition will forever elude researchers. Despite this fundamental difficulty, researchers have made significant progress in texture segmentation by focusing on certain attributes of texture. The progress in texture segmentation is reflected in their wide-ranging application in real world problems. Examples of texture segmentation in use are common, such as in surveillance activities [49] and biomedical image processing for diagnosis [75].

Historically, there have been three main approaches to texture segmentation: structural, statistical and filter-based. Structural approaches attempt to characterise textures using texture primitives and arrangement rules of these. This approach is based on the notion that textures are composed of regular tilings of a set of basic texture elements. However, the success of such approaches is also hamstrung by the fact that real world textures often have very complex arrangement rules, as well as texture primitive distortion over the entirety of a textured region. A powerful grammar that can encompass these intricacies is, first of all, very difficult to construct, and secondly, very clumsy to apply

to practical problems. Statistical approaches attempt to describe textures from the statistical data obtained from the textured image. These approaches are more fundamental in the sense that they deal with the raw images directly, but higher-level concepts like texture primitive arrangement are more difficult to incorporate. Haralick [27] cited eight statistical approaches to the measurement and characterisation of textures. They often focus on the spatial frequency properties or edge densities of images. Such measures are taken directly from the gray levels of the image pixels. Statistical approaches have found some success in early texture analysis problems, mainly due to their fundamental nature.

Neither the statistical nor the structural approach is satisfactory for a general analysis of textures, where the relative importance of texture primitives and their arrangements vary. What is required is a method to effectively isolate the defining qualities for a particular texture, whether it be a local property or a global arrangement rule. More recent techniques involve the use of a variable-scale analysis of textured images. The main advantage of this approach is that it is capable of zooming to arbitrary scales in the analysis, thus allowing examination of textures at their appropriate scales. Even within a homogeneous texture, there may be important, defining characteristics at more than one scale. Most of the research efforts on multiscaled approaches use spatial filters with different frequency characteristics as the primary analysis tool. Hence, these approaches are often called filter-based or multi-channel filtering approaches. Early incarnations of these methods use Gabor filters at different scales to characterise textures. More recently, these approaches have evolved to encompass the use of wavelets. Wavelets are, arguably, the more natural mathematical tool to use for this purpose, due to their inherent multiresolution properties. Another potential advantage for wavelets is the non-redundant nature of some wavelet representations. Redundant representations place great computational demands on the algorithms, which reduce their usefulness in certain latency-sensitive applications.

The motivations for investigating wavelet-based texture segmentation in this work is two-folds. Firstly, there have been many exciting new developments in the field of wavelets, and it is interesting to apply these to texture analysis. Secondly, the vast majority of texture segmentation algorithms are very computationally intensive. In many applications, it is highly desirable to have fast texture segmentation algorithms. This would enable texture analysis to be incorporated into more powerful computer vision systems.

1.2 Historical Perspective of Wavelets: Fourier Analysis

The Fourier transform has been the standard analytical tool for engineers and scientists over the past two centuries. Fourier discovered that every periodic function can be expressed as a superposition of simple sinusoids at different harmonic frequencies. This discovery was later extended to cover aperiodic functions, which gave rise to the Fourier transform. The definitions for the Fourier transform and its inverse are

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} \mathrm{d}t \tag{1.1}$$

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{j\omega t} \mathrm{d}\omega \tag{1.2}$$

where the pair of functions $\{f(t), F(\omega)\}$ is known as a Fourier transform pair. The Fourier transform can be viewed as an expansion in the orthonormal basis $\{e^{-j\omega t} | \quad \omega \in \mathbf{R}\}$. These basis functions are the eigenfunctions for linear, time-invariant operators. In studying such systems, the Fourier transform is the perfectly natural tool to use; it essentially performs an eigenfunction expansion on the signals of interest. As such, these techniques form the theoretical foundations for frequency domain analysis that is very important in the mathematical sciences. A disadvantage associated with the Fourier basis is that its basis functions are pure sinusoids, and therefore have infinite support in time. In other words, they have zero localisation in space. This fact renders Fourier techniques unsuitable for describing signals containing discontinuities and sharp spikes. Unfortunately, many signals of practical interests exhibit such characteristics. An illustration of the poor time localisation of the Fourier transform is the famous Gibbs' phenomenon. Another perspective of the localisation problem of the Fourier transform representation can be obtained by examining the inverse transform expression in equation (1.2). The convolution kernel in this expression, $e^{j\omega t}$, has unit magnitude, indicating equal contributions to the inverse transform from every frequency in the transform. The Fourier representation's lack of compactness for non-stationary signals is a major disadvantage in applications such as signal compression.

A significant milestone in the development of Fourier analysis was the discovery of a fast algorithm for computing Fourier transforms, the *Fast Fourier Transform (FFT)*. Along with rapid advances in digital computers, the FFT provided a practically feasible

means for analysing very large data sets. Therefore, Fourier analysis remains, to this day, an important technique for scientists and engineers.

1.3 Time-frequency Representations

The shortcomings of traditional Fourier analysis for non-stationary, aperiodic signals were becoming evident by the early parts of this century. In particular, scientists and engineers needed methods of describing, or representations of, signals that are localised in both time and frequency. These methods are known as *joint time-frequency representations*.

The time and frequency uncertainty (i.e. localisation) of a time-varying function $\psi(t)$, with Fourier transform $\Psi(\omega)$, is defined as

$$\Delta t(\psi) = \frac{\int_{-\infty}^{+\infty}(t-\overline{t})^2|\psi(t)|^2\mathrm{d}t}{E} \tag{1.3}$$

$$\Delta\omega(\psi) = \frac{\int_{-\infty}^{+\infty}(\omega-\overline{\omega})^2|\Psi(\omega)|^2\mathrm{d}\omega}{E} \tag{1.4}$$

where

$$\overline{t} = \frac{\int_{-\infty}^{+\infty} t|\psi(t)|^2\mathrm{d}t}{E} \tag{1.5}$$

$$\overline{\omega} = \frac{\int_{-\infty}^{+\infty} \omega|\Psi(\omega)|^2\mathrm{d}\omega}{E}$$

$$\text{and} \quad E = \int_{-\infty}^{+\infty}|\Psi(\omega)|^2\mathrm{d}\omega = \int_{-\infty}^{+\infty}|\psi(t)|^2\mathrm{d}t \tag{1.6}$$

Qualitatively, the quantity Δt measures the degree of spread of the function $\psi(t)$ about its central value $\overline{t}$. Similarly, $\Delta\omega$ measures the spread of its spectrum around the central value ω. Heisenberg's uncertainty principle places a fundamental lower bound on the product of time and frequency uncertainties:

$$\Delta t(\psi)\Delta\omega(\psi) \geq \frac{1}{4\pi} \tag{1.7}$$

A function is said to be well-localised in time and frequency if the Heisenberg product is as close to the fundamental limit as possible. The Heisenberg product for the traditional Fourier basis is undefined, since the complex exponential functions have zero frequency uncertainty and infinite time uncertainty. More generally, one can visualise the time and frequency resolution tradeoff as a rectangular "atom" in the time-frequency plane, as

shown in figure 1.1. Such an atom is free to extend or contract in both dimensions as long as its area does not shrink below the lower bound mandated by equation (1.7). All time-frequency representations divide the time-frequency plane in their own manner.

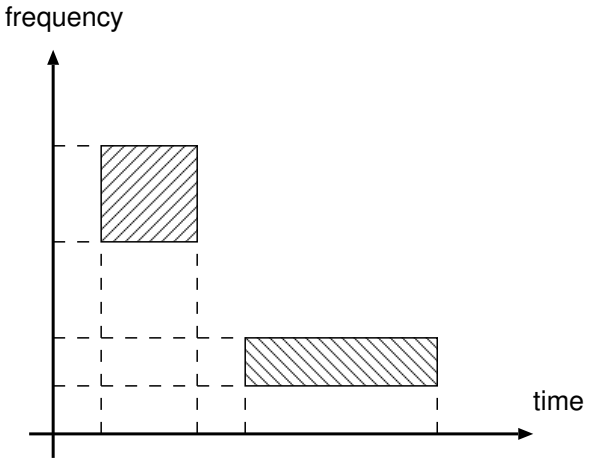


Figure 1.1. The time-frequency plane, illustrating the time-frequency uncertainties for two different atoms.

In 1946, Gabor introduced the *Short-Time Fourier Transform (STFT)*, or the *Gabor transform*. This was an attempt to solve the time localisation problems associated with the traditional Fourier Transform. The Gabor transform is defined as

$$F_{\text{Gabor}}(\omega, \tau) = \int_{-\infty}^{+\infty} w^*(t-\tau) f(t) e^{-j\omega t} \mathrm{d}t \tag{1.8}$$

where $w(t)$ is a window function centred on $t = 0$. In Gabor's original formulation, the window function was chosen as a Gaussian, $w(t) = e^{-(\frac{t}{2\sigma})^2}$. The Fourier transform is thus modified to contain two parameters, one for describing the frequency localisation, the other for time localisation. In general, arbitrary windows with finite time support can also be used, and such transforms are collectively called *windowed Fourier transforms (WFT)*. When viewed as a signal expansion operation, the WFT corresponds to the use of windowed sinusoids for the basis functions. Real-valued families of WFT are sometimes called local trigonometric bases. All these bases overcome some of the shortcomings of the Fourier basis by offering some degree of time-frequency localisation. However, the main problem with these bases is their use of fixed-sized time windows. In other words, the time uncertainty is fixed for all frequencies, and this may be inappropriate for some applications. For example, consider a signal containing a long, sinusoidal steady-state component along with short, spiky transients. A single-sized window is not capable

of capturing both types of variations adequately within the same description. A long window would suit the steady-state response, while short windows are needed for the transient bursts. This problem can be addressed by using variable window widths, which corresponds to a multi-scaled approach. This concept of a multiresolution analysis is an integral part of wavelet theory.

1.4 Wavelet Theory

The theory of wavelets was built on the foundations of joint time-frequency representations, and has evolved rapidly over the last decade to become a powerful tool for scientists and engineers today. A continuous wavelet transform (CWT) is fundamentally an inner-product expansion of a signal over a basis consisting of translated and dilated versions of a single function. Mathematically, this is

$$\Psi(a,b) = \int_{-\infty}^{+\infty} \psi_{a,b}^*(t) f(t) \mathrm{d}t \tag{1.9}$$

where $\psi_{a,b}^*(t)$ is the complex conjugate of the function $\psi_{a,b}(t)$ and $\psi_{a,b}(t)$ is defined as

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}} \psi(\frac{t-b}{a}) \tag{1.10}$$

$\psi(t)$ is called the mother wavelet, which must satisfy the *admissibility criterion*

$$C_\psi = 2\pi \int_{-\infty}^{+\infty} \frac{|\Psi(\omega)|^2}{|\omega|} \mathrm{d}\omega < \infty \tag{1.11}$$

where $\Psi(\omega)$ is the Fourier transform of $\psi(t)$. This condition ensures that the set of all translation and dilation of $\psi(t)$ forms a basis for the L_2-space, i.e. the space of all finite energy functions. This is an important mathematical property, because it shows that we can decompose any arbitrary real-world signal into a series of wavelet functions, thereby validating the CWT as a general purpose analysis tool. Moreover, one is free to place additional constraints to produce bases with special properties such as orthogonality. The CWT is a linear, invertible, shift-invariant transform which also preserves energy, similar to Parseval's theorem for Fourier transforms. Particular wavelets can be constructed to give desirable mathematical properties such as orthogonality and arbitrary smoothness.

It is evident that the wavelet transform uses analysing functions that are characterised by a dilation factor, a, and a translation factor, b. This allows the analysis of functions

at arbitrary time and frequency resolutions. In practice, such freedom is unnecessary, since most interesting information are contained in a few time-frequency atoms. This implies that the values of a and b may be discretised without reducing the power of the wavelet transform. In addition, with the increasing involvement of computers in scientific work, it is desirable to work with a discretised form of the wavelet transform. It is standard practice to employ only dyadic values of a and integer shifts relative to each scale ($a = 2^{-j}, b = 2^{-j}k$, where $j, k \in \mathbb{Z}$). More general values for a and b are entirely possible, but they have not been heavily applied to practical problems. The various types of discrete wavelet transforms, and their associated properties, will be covered in more detail in Chapter 2. An attractive property of discrete wavelet transforms is that they form a multiresolution analysis on the signal space. Effectively, the transforms automatically zoom to several different scales and produce a multi-scale representation of the input data. This multi-scale approach is especially useful in the context of texture analysis, since texture is an inherently multi-scale phenomenon. Another advantage of discrete wavelets is the existence of fast algorithms for computing the transforms. A computationally efficient means for calculating wavelet transforms enables its use in many more applications. These are the strong justifications for employing wavelet transforms as the mathematical tool for texture analysis.

1.5 Aims of this Thesis

While texture segmentation has been a topic with a long history, it can still be regarded as a relatively unsolved problem. Various texture segmentation schemes have their own advantages and disadvantages, and not a single one can claim overall superiority in all circumstances. The author believes that such a solution may never be found. It is with this philosophy that this thesis approaches texture segmentation; it aims to achieve an efficient, robust method for texture segmentation. The framework of this research is based on a simple system architecture, which is strictly adhered to throughout this work. The proposed system consists of a feature extractor and a clustering algorithm. This system takes an image with multiple textured regions as input, and produces a segmented image as its output. It is desirable to minimise the number of auxiliary inputs, in the form of external parameters, needed by the system. Conceptually, the system should ideally be a black box device. This thesis examines the system performance with different

algorithms for both texture feature extraction and feature clustering, with the aim of identifying an efficient combination of techniques. The judgement will solely be based on the segmentation performance achieved for various input test images. Performance will be measured using directly calculated, quantitative metrics. Human visual interpretation is only used to augment the quantitative results, whenever it is appropriate to do so.

1.6 Contributions of Thesis

Throughout the undertaking of this research work, contributions have been made to several areas in texture segmentation. The key contributions are outlined below:

- The Dual-tree Complex Wavelet Transform [36] is used as the primary analysis tool for texture feature extraction. Previous work in wavelet methods for texture segmentation have not made use of this technique. This transform has some fundamental advantages over traditional techniques such as Gabor and Discrete Wavelet Transforms. [Chapters 2 and 3]

- A novel texture feature extraction scheme is introduced. The scheme has two main parts:

  - a feature extraction stage which is designed specifically to extract texture features directly from Discrete Wavelet and Dual-tree Complex Wavelet Transform coefficients. The processing in this stage is an innovative adaptation of previous efforts.

  - feature conditioning is an important phase after feature extraction. The contribution is in the focus on attempting to find an optimal smoothing window shape for a texture, through the use of Kaiser windows with variable sidelobe levels. [Chapter 3]

- Extracted features are examined for their separability. A number of different techniques are used to gauge the degree of separability for a given feature set, before clustering. [Chapter 5]

- A novel modified K-means clustering algorithm is introduced. The mixing of the additional spatial proximity measure with the feature space distance is proposed. This

new combined measure is believed to be very beneficial for texture segmentation. [Chapter 5]

- The segmentation performance of the novel feature extraction method, and the proposed modified K-means algorithm are studied through a large number of segmentation experiments. More than 100 commonly-used artificial texture mosaics are used in the experiments, making this one of the more comprehensive studies. Commonly-used images are chosen to facilitate meaningful comparisons with results already in the literature. [Chapter 5]

- The texture segmentation techniques developed in this thesis are applied to several real-world applications. The feature extraction and modified K-means algorithms are useful in producing decent segmentations of real images, ranging from aerial photographs of terrain to scene object segmentation. [Chapter 6]

1.7 Thesis Overview

Chapter 2 reviews wavelet theory in some detail. While a full treatment of wavelets is far beyond the scope of a single thesis, a detailed description of the particular wavelets used for texture segmentation is important in the context of this thesis. This chapter lays the theoretical groundwork on which the entire system is based.

Chapter 3 describes the feature extraction process in texture segmentation problems. Historical approaches to texture feature extraction are examined. A novel feature extraction method based on wavelet transforms is presented in this chapter, including a discussion of the important feature conditioning process. This extraction method computes features directly from wavelet transform coefficients of textured images.

In Chapter 4, methods for clustering features to produce a full segmentation are examined. The focus is on the K-means clustering algorithm, but several other alternative algorithms are also presented.

The main results of this thesis appear in Chapter 5, where the texture segmentation experiments are described in full detail. Visual and quantitative results are presented, as they are both important measures for the merits of our process. Discussion of the results

and their significance follows, and a proposed method for improving the quality of the segmentations is examined.

With the bulk of the experiments being performed on test data sets, it is interesting to apply the techniques developed in this thesis to some real application examples. Chapter 6 considers three real-world examples, to which the developed segmentation algorithms are applied.

Finally, Chapter 7 draws the conclusions of this thesis, and discusses possible future directions for continuation of this research.

This page is blank

Chapter 2

Wavelet Transforms

2.1 Multiresolution Analysis and Wavelet Transforms

The history of the wavelet transform traces back to Fourier theory. In a Fourier decomposition, a signal is expanded as an integral of sinusoidal oscillations over a range of frequencies. This results in a linear, invertible and orthonormal transform which has been widely applied to many areas in science and engineering. However, a major weakness of Fourier theory is its poor time-frequency characteristics. The lack of any time information in a Fourier representation makes this technique unsuitable for many real world signals. Thus joint time-frequency representations were born. A common element among these representations is some mechanism that allows a trade-off between time and frequency resolutions. The first of such representations is the *Windowed Fourier Transform*, which uses an additional time window in the transform. In particular, the Gabor transform is one such transform with a Gaussian window. Time-frequency representations later evolved to other forms which do not use windowed sinusoids as the analytical basis. Wavelet theory is one such example. This chapter attempts to provide a suitable treatment of wavelets, which would serve as the background theory for the rest of the work in this thesis.

2.1.1 Continuous Wavelet Transforms

In his work on the analysis of geophysical data, Morlet used a variation of the Gabor transform, where the window function had a variable width. He called the resulting

analysing functions "wavelets of constant shape". Working with Grossmann, Morlet later extended his theory, and formulated the *Continuous Wavelet Transform (CWT)*. The CWT is a linear transform whose basis functions are simply dilations and translations of a single function, called the *mother wavelet*, $\psi(t)$. A mother wavelet must satisfy the admissibility criterion [17, 18, 81] given in equation (1.11). For most common function spaces of practical interests, $\Psi(\omega)$ will always have sufficient decay at high frequencies, so this condition reduces to $\Psi(\omega) = 0$ at the origin. In other words, an admissible mother wavelet must have a vanishing integral over the real line. It is usual practice to normalise the mother wavelet so that it has unit energy:

$$\| \psi(t) \|^2 = \int_{-\infty}^{\infty} |\psi(t)|^2 \, \mathrm{d}t = \frac{1}{2\pi} \int_{-\infty}^{\infty} \| \Psi(\omega) \|^2 \, \mathrm{d}\omega = 1 \tag{2.1}$$

The CWT of a function $f(t)$, with respect to mother wavelet $\psi(t)$, is defined as

$$\begin{aligned} W_f(a,b) &= \langle f, \psi_{a,b} \rangle = \int_{-\infty}^{\infty} f(t) \psi^*_{a,b}(t) \, \mathrm{d}t \\ \text{where} \qquad \psi_{a,b}(t) &= \tfrac{1}{\sqrt{a}} \psi(\tfrac{t-b}{a}), \quad a > 0, b \in \mathbf{R} \end{aligned} \tag{2.2}$$

Equation (2.2) reveals that the CWT is a highly redundant transform, since it maps a function of one variable to a function of two variables. The two variables correspond to the scale (a) and the position (b) of the analysis function. Notice that $\psi_{a,b}(t)$ is normalised so that $\| \psi_{a,b}(t) \| = 1$ as well. The CWT is a linear transform, since the inner-product operator is linear. It satisfies shift invariance, an important property that is valuable in data feature extraction. The CWT is also energy preserving, similar to Parseval's result for Fourier transforms. This property is expressed as

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |W_f(a,b)|^2 \, \frac{\mathrm{d}a\mathrm{d}b}{a^2} \tag{2.3}$$

where C_ψ is as defined in equation (1.11). The CWT is invertible, and the reconstruction formula is

$$f(t) = \frac{1}{C_\psi} \int_0^{\infty} \int_{-\infty}^{\infty} W_f(a,b) \psi_{a,b}(t) \, \frac{\mathrm{d}a\mathrm{d}b}{a^2} \tag{2.4}$$

This expression is identical to Calderón's identity discovered in 1960 [48]. A proof of the inversion formula can be found in [81].

In many applications, it is often desirable to partition the time-frequency plane into disjoint regions. This can be done by discretising the scaling and translation factors in equation (2.2). This leads to the various discretised wavelet transforms. Another

reason for discretising the CWT is to reduce its tremendous redundancy. In real-world problems, redundancy incurs a great computational penalty on the system, thus reducing its efficiency.

2.1.2 Discrete Wavelet Transforms

The basic discretisation of the CWT can be characterised by $a = a_0^m, b = nb_0a_0^m$, where $a_0 > 1, b_0 > 0$ are fixed, and $n, m \in \mathbf{Z}$. The discretised wavelet transform is expressed as

$$W_f(m,n) = a_0^{-m/2} \int_{-\infty}^{\infty} f(t)\psi^*(a_0^{-m}t - nb_0)\,\mathrm{d}t \tag{2.5}$$

The translation step size, b is dependent on the chosen dilation factor a. This choice allows the wavelets $\psi_{a,b}$ to adapt to different resolutions, with large step sizes at low resolution and vice versa. In the vast majority of cases, $a_0 = 2$ and $b_0 = 1$ are used, resulting in dyadic wavelets which partitions the time-frequency plane into binary regions. Other values of a_0, b_0 are entirely possible, and their theory is very similar to that of dyadic wavelets. Non-dyadic have been applied to several applications, but these instances are very rare. The remainder of this thesis will only deal with dyadic wavelets.

The theory for the discretised wavelet transform is based on the mathematical framework for *frames*. A frame is a set of vectors, not necessarily independent, that spans a given vector space. That is, frames are not bases, and they decompose signals into a non-unique, redundant representation. In the current context, the abstract concept of vectors is taken to refer to real-valued functions (e.g. $\psi(t)$). To be precise, a frame is defined as a set of functions $g_i, i \in J$, in a Hilbert space $\mathbf{H}$, for which there exist $0 < A \leq B < \infty$, such that, for all $f \in \mathbf{H}$,

$$A \parallel f \parallel^2 \leq \sum_{i \in J} |\langle g_i, f \rangle|^2 \leq B \parallel f \parallel^2 \tag{2.6}$$

where $\parallel f \parallel^2$ is the L_2-norm of f, and the constants, A and B, are called frame bounds. If $A = B$, then the frame is called a "tight" frame. In this case, the constant A gives the redundancy ratio of the representation. If the frame is linearly independent, then it is called a *Riesz basis*. For the special case of a Riesz basis with $A = B = 1$, the result is an *orthonormal* basis; this case will be discussed in more detail later. In the current context, the set of discretised wavelet functions $\psi_{m,n}(t) = 2^{-m/2}\psi(2^{-m}t - n)$ is a frame in L^2, the space of all finite-energy signals. For simplicity, only Riesz bases are

considered in subsequent discussions of wavelet frames. Since frames are complete, the functions $\psi_{m,n}(t)$ are capable of representing every signal. Moreover, it should be possible to construct sets of $\psi_{m,n}(t)$ to form tight frames, or even orthonormal bases.

Unlike the CWT, frame representations are not immediately invertible. To reconstruct a signal from its frames decomposition, one must first construct a *dual frame*. Consider a frame of wavelets, $\{\psi_{m,n}(t)\}$; its dual frame, $\{\tilde{\psi}_{m,n}(t)\}$, is defined as

$$\tilde{\Psi}_{m,n}(\omega) = \frac{\Psi_{m,n}(\omega)}{\sum_{k=-\infty}^{\infty} |\Psi_{m,n}(\omega + 2\pi k)|^2} \tag{2.7}$$

where $\tilde{\Psi}(\omega)$ is the Fourier transform of $\tilde{\psi}(t)$. The dual frame is orthogonal with respect to the original frame:

$$\langle \psi_{m,n}, \tilde{\psi}_{j,k} \rangle = \delta_{j,m}\delta_{k,n}, \quad j,k,m,n \in \mathbf{Z} \tag{2.8}$$

where $\tilde{\psi}_{j,k} = 2^{-j/2}\tilde{\psi}(2^{-j}t - k)$ and $\delta_{j,k}$ is the Kronecker delta. It is this property which gives biorthogonal bases their name. Obviously, orthonormal bases correspond to the special case where the set is self-dual. The reconstruction formula for a wavelet frames basis is

$$f(t) = \sum_{m,n\in\mathbf{Z}} \langle \psi_{m,n}, f \rangle \tilde{\psi}_{m,n} = \sum_{m,n\in\mathbf{Z}} \langle \tilde{\psi}_{m,n}, f \rangle \psi_{m,n} \tag{2.9}$$

Equation (2.9) illustrates the interchangeable nature of dual bases, even though the properties of these functions can be quite different. This interchangeability has been exploited in applications such as image compression, to achieve better image quality at fixed compression ratios.

2.1.3 Multiresolution Analysis and Orthonormal Wavelets

There are several different approaches to the theory of wavelets. The previous section arrived at wavelets from a real analysis' perspective. Alternatively, there are Mallat's multiresolution and an engineering-motivated filter banks approaches. All of these approaches are fundamentally equivalent, but Mallat's approach is the most intuitively appealing one. We will now take the multiresolution approach, which will be followed by a discussion of its relationships with filter banks.

For simplicity, let us consider only the space of finite energy functions, $L^2(\mathbf{R})$. In no way does this limit our discussion, for all real-world signals belonging to this space. A Multiresolution Analysis (MRA) on $L^2(\mathbf{R})$ consists of [17],[18],[47]:

1. A sequence of nested subspaces, $V_j, j \in \mathbf{Z}$,

$$\ldots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \ldots \tag{2.10}$$

2. Upward completeness of the subspaces

$$\overline{\bigcup_{j \in \mathbf{Z}} V_j} = L^2(\mathbf{R}) \tag{2.11}$$

3. Downward completeness

$$\bigcap_{j \in \mathbf{Z}} V_j = 0 \tag{2.12}$$

4. Scale Invariance

$$f(t) \in V_j \Rightarrow f(2t) \in V_{j-1} \tag{2.13}$$

5. Shift Invariance

$$f(t) \in V_j \Rightarrow f(t-k) \in V_j \quad \forall k \in \mathbf{Z} \tag{2.14}$$

6. Existence of basis

$$\begin{gathered} \exists \phi(t) \in V_0 \text{ such that } \{\phi_{0,n} | n \in \mathbf{Z}\} \\ \text{forms a Riesz basis for } V_0 \text{ where } \phi_{0,n} = \phi(t-n) \end{gathered} \tag{2.15}$$

Notice that conditions (2.13) and (2.15) imply that $\{\phi_{j,k}; j, k \in \mathbf{Z}\}$ is a Riesz basis for V_j, where $\phi_{j,k}(t) = 2^{-j/2}\phi(2^{-j}x - k)$. An interpretation of an MRA is that it is a mechanism for approximating functions at different scales. The hierarchical spaces V_j are used to approximate any given function to a certain accuracy. Decreasing j yield a finer and finer approximations to a function $f \in L^2(\mathbf{R})$. In a more specific setting, the last condition given in equation (2.15) can be replaced by specifying the existence of a $\phi(t)$ such that $\{\phi_{0,n}(t)\}$ forms an orthonormal basis for V_0. This would lead to orthonormal wavelet representations of $L^2(\mathbf{R})$, as described in the following paragraphs.

A consequence of the above structure of an orthonormal MRA is that it is possible to explicitly construct an orthonormal wavelet basis for $L^2(\mathbf{R})$. To make the arguments clear, let us define the orthogonal complement of V_j in V_{j-1} by

$$V_{j-1} = V_j \oplus W_j, \qquad W_j \perp V_j \tag{2.16}$$

Referring back to the approximation interpretation of an MRA, the complement subspaces W_j are the errors spaces at different resolutions. The multiresolution structure of the subspaces V_j means that the spaces $W_j, j \in \mathbf{Z}$ satisfy the following,

- Completeness

$$\bigoplus_{j\in\mathbf{Z}} W_j = L^2(\mathbf{R}) \tag{2.17}$$

- Scale Invariance

$$f(t) \in W_j \Rightarrow f(2t) \in W_{j-1} \tag{2.18}$$

- Shift Invariance

$$f(t) \in W_j \Rightarrow f(t-k) \in W_j \quad \forall k \in \mathbf{Z} \tag{2.19}$$

The above properties imply that there exists a function $\psi(t)$ such that $\{\psi(t-k); k \in \mathbf{Z}\}$ and $\langle \psi(t), \phi(t) \rangle = 0$, forms an orthonormal basis for W_0. Hence, by properties (2.17) and (2.18), $\{\psi_{j,k}(t),\ j,k \in \mathbf{Z}\}$ is an orthonormal basis for $L^2(\mathbf{R})$. Thus, finding a wavelet basis for $L^2(\mathbf{R})$ is equivalent to finding an orthonormal basis for the subspace W_0 in the multiresolution analysis structure.

The algorithm for finding an orthonormal wavelet basis usually begins with finding a suitable scaling function $\phi(t)$. From the MRA structure, one has

$$\phi(t) = \sum_{n\in\mathbf{Z}} p(n)\phi_{-1,n}(t) \tag{2.20}$$

where $p(n) = \langle \phi(t), \phi_{-1,n}(t) \rangle$ is a sequence of real numbers, since $\phi(t) \in V_0 \subset V_{-1}$, and $\{\phi_{-1,n}, n \in \mathbf{Z}\}$ is a basis for V_{-1}. The sequence $p(n)$ is referred to as a filter. Obviously, this equation is valid for all pairs of successive scales. This is the *dilation equation* that is fundamental in wavelet theory. In the Fourier domain, the dilation equation becomes

$$\Phi(\omega) = P(\frac{\omega}{2})\Phi(\frac{\omega}{2}) \tag{2.21}$$

where $P(\omega)$ is the discrete Fourier transform of $p(n)$

$$P(\omega) = \sum_{n \in \mathbf{Z}} p(n) e^{-j\omega n} \tag{2.22}$$

This condition merely corresponds to the ladder structure of the MRA. To construct orthonormal bases, one imposes the extra constraint of orthonormality:

$$\langle \phi(t), \phi(t-k) \rangle = \int_{-\infty}^{\infty} \phi(t)\phi^*(t-k)\,\mathrm{d}t = \delta_{k,0}, \quad \forall k \in \mathbf{Z} \tag{2.23}$$

or equivalently, in the Fourier domain,

$$\sum_{l \in \mathbf{Z}} |\Phi(\omega + 2\pi l)|^2 = \frac{1}{2\pi} \tag{2.24}$$

where $\Phi(\omega)$ is the Fourier transform of $\phi(t)$. Combining equations (2.21) and (2.24) yields a condition for the filter, $p(n)$,

$$|P(\omega)|^2 + |P(\omega + \pi)|^2 = 1 \tag{2.25}$$

Once having found a filter that satisfies equation (2.25), the scaling function $\phi(t)$ can be computed by iterating equation (2.21),

$$\Phi(\omega) = \prod_{j=1}^{\infty} P(2^{-j}\omega) \tag{2.26}$$

provided that the infinite product converges. In practice, the infinite product is guaranteed to converge if $P(0) = 1$, which, together with equation (2.25), means $p(n)$ must be a low-pass filter. Finally, the wavelet can be derived from the scaling function. By carrying out arguments similar to the construction of ϕ, one can arrive at an appropriate choice for ψ,

$$\Psi(\omega) = e^{j\omega/2} \cdot \overline{P(\frac{\omega}{2} + \pi)} \cdot \Phi(\frac{\omega}{2}) \tag{2.27}$$

or in time domain,

$$\psi(t) = \sum_n q(n)\phi_{-1,n}(t), \quad \text{where} \quad q(n) = (-1)^n p(1-n) \tag{2.28}$$

Since $p(n)$ is a low-pass filter, the filter $q(n)$ must be high-pass. It is now clear that constructing an orthonormal wavelet is equivalent to finding a low-pass filter $p(n)$ that satisfies equation (2.25). There are many solutions to equation (2.25), and additional conditions are usually applied to yield particular solutions. For example, the famous

Daubechies family of wavelets are constructed with filters that are smooth and of finite length (compact support). These conditions restrict the filters to assume the form $P(\omega) = [\frac{1}{2}(1 + e^{j\omega})]^N Q(e^{j\omega})$, where Q is a polynomial, and N is an integer that measures the smoothness, or *regularity* [17], of the filter. Another example would be the Meyer orthonormal wavelet, which has infinite smoothness, but lacks compact support. In practical applications, it is highly desirable to have FIR $p(n)$, which produces wavelets with compact support. However, it is well known that orthonormal wavelets with compact support cannot have symmetry properties. In other words, the filters $p(n)$ cannot be linear phase.

The MRA structure presented in equations (2.10)-(2.15) can be generalised to construct biorthogonal wavelets. Recall that a biorthogonal basis is composed from two mother wavelets, $\psi(t), \tilde{\psi}(t)$ which are duals of each other. The biorthogonality refers to the fact that dyadic dilations and translations of one mother wavelet is orthogonal to the other. This behaviour can be incorporated into the modified MRA by specifying *two* hierarchical approximation subspaces

$$\begin{aligned} &\ldots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \ldots \\ &\ldots \tilde{V}_2 \subset \tilde{V}_1 \subset \tilde{V}_0 \subset \tilde{V}_{-1} \subset \tilde{V}_{-2} \ldots \end{aligned} \tag{2.29}$$

where each sequence satisfies the usual MRA properties. Let's define the complementary sequences of subspaces $W_j, \tilde{W}_j$, where

$$\begin{aligned} V_{j-1} &= V_j \oplus W_j, \qquad W_j \perp \tilde{V}_j \\ \tilde{V}_{j-1} &= \tilde{V}_j \oplus \tilde{W}_j, \qquad \tilde{W}_j \perp V_j \end{aligned} \tag{2.30}$$

Notice how the complement of V_j in V_{j-1}, W_j, is orthogonal to the *dual* approximation space at the same scale, $\tilde{V}_j$. Thus, the biorthogonal multiresolution analysis boot-straps itself as it progresses to coarser scales. The dilation equations and the filter conditions can be derived in a similar fashion as for the orthonormal case.

$$\begin{aligned} \phi(t) &= \sum_{n \in \mathbf{Z}} p(n) \phi_{-1,n}(t) \\ \tilde{\phi}(t) &= \sum_{n \in \mathbf{Z}} \tilde{p}(n) \tilde{\phi}_{-1,n}(t) \end{aligned} \tag{2.31}$$

where $p(n), \tilde{p}(n)$ are real sequences. Define $P(\omega)$ and $\tilde{P}(\omega)$ as the discrete Fourier transforms of $p(n)$ and $\tilde{p}(n)$, respectively; the two scaling functions can be obtained from

$$\begin{aligned} \Phi(\omega) &= \prod_{j=1}^{\infty} P(2^{-j}\omega) \\ \tilde{\Phi}(\omega) &= \prod_{j=1}^{\infty} \tilde{P}(2^{-j}\omega) \end{aligned} \tag{2.32}$$

The wavelet construction equations become

$$\begin{aligned} \Psi(\omega) &= e^{j\omega/2} \cdot \tilde{P}^*(\tfrac{\omega}{2} + \pi) \cdot \Phi(\tfrac{\omega}{2}) \\ \tilde{\Psi}(\omega) &= e^{j\omega/2} \cdot P^*(\tfrac{\omega}{2} + \pi) \cdot \tilde{\Phi}(\tfrac{\omega}{2}) \end{aligned} \tag{2.33}$$

or in time domain,

$$\begin{aligned} \psi(t) &= \sum_n \tilde{q}(n)\phi_{-1,n}(t), \quad \text{where} \quad \tilde{q}(n) = (-1)^n \tilde{p}(1-n) \\ \tilde{\psi}(t) &= \sum_n q(n)\tilde{\phi}_{-1,n}(t), \quad \text{where} \quad q(n) = (-1)^n p(1-n) \end{aligned} \tag{2.34}$$

As in the orthonormal case, we are free to apply further conditions on the filters $p(n), \tilde{p}(n)$. Since the wavelets no longer need to be orthonormal, it allows an extra degree of freedom. In particular, it is usual to require that $p(n)$ and $\tilde{p}(n)$ to be symmetric or anti-symmetric, meaning that these filters will have linear phase. The linear phase property is vital in applications such as image compression, because phase distortions produce highly undesirable visual artifacts. The filter relations for the orthonormal and biorthogonal wavelets as derived in this section forms the basis for the development of the Fast Wavelet Transform.

2.2 Fast Wavelet Transform and Perfect Reconstruction

A milestone in the development and popularisation of Fourier analysis was the discovery of the Fast Fourier Transform (FFT) algorithms. The FFT, along with the rapid advancements in computing technologies, were the major enabling factors behind the application of Fourier techniques to a wide range of problems. In essence, the FFT algorithms utilised factorisations of the unitary transform matrix into the product of sparse matrices. Successive multiplications by sparse matrices can be implemented extremely efficiently on computers, and the net gain of the FFT is huge: an $O(N \log N)$ algorithm versus an

$O(N^2)$ by direct computations. In many ways, the Fast Wavelet Transform (FWT) plays a similar role in the application of wavelet theory to practical applications. Historically, Mallat [45] first developed an efficient, discrete algorithm for computing decompositions and reconstructions of sampled signals in 1986. In her landmark paper in 1988, Daubechies [17] pointed out the equivalence of Mallat's work to multiresolution analysis and wavelet transforms. The nature of Mallat's algorithm is closely related to sampled filter-banks in engineering, which have been used for decades before their relationship with wavelets were recognised.

2.2.1 The Fast Wavelet Transform algorithm

The multiresolution analysis framework on which orthonormal and biorthogonal wavelet bases are constructed naturally leads to an efficient, hierarchical algorithm for computing wavelet coefficients. Let's begin with the more general biorthogonal case. Assume that we start with an expansion of a function in V_j, i.e.

$$f(t) = \sum_n c_j(n)\phi_{j,n}(t), \quad \text{where} \quad c_j(n) = \langle f(t), \phi_{j,n}(t) \rangle \tag{2.35}$$

It is then straight-forward to compute all the coefficients $c_{j+1}(n)$ and $d_{j+1}(n)$ from a knowledge of $c_j(n)$. Using equations (2.20) and (2.28), we have

$$d_{j+1}(n) = \langle f, \psi_{j+1,n} \rangle = \sum_k q(k-2n)c_j(k) \tag{2.36}$$

and

$$c_{j+1}(n) = \langle f, \phi_{j+1,n} \rangle = \sum_k p(k-2n)c_j(k) \tag{2.37}$$

where $p(n)$ and $q(n)$ are the same filters as defined in the dilation equations (2.20) & (2.28). Equations (2.36) and (2.37), can be viewed as a filtering operation with the time-reversed versions of the filters $p(n)$ and $q(n)$, followed by discarding every second sample (downsampling by 2). The downsampling preserves the total amount of coefficients. At each level, the data, $c_j(n)$, is separated into two halves, or *subbands*: one contains the low frequency, coarse information ($c_{j+1}(n)$), and the other half contains the high-frequency, detailed differences between successive resolutions ($d_j(n)$). This is commonly known as the *analysis* stage, and is illustrated in figure 2.1. A full k-level FWT re-iterates equations (2.36) and (2.37) k times to yield a successive approximation to the original data $c_0(n)$.

The higher level coefficients approximate the data at lower resolutions, and vice versa. The mathematical analysis of the FWT is usually carried out in the z-domain. The z-transform of a discrete sequence $\{a(n), n \in \mathbf{Z}\}$ is defined as

$$A(z) = \sum_{n \in \mathbf{Z}} a(n) z^{-n} \tag{2.38}$$

Let $\tilde{h}$ and $\tilde{g}$ to be the time-reversed versions of $p(n)$ and $q(n)$, respectively. The analysis stage equations are

$$\begin{aligned} C_{j+1}(z) &= \tilde{H}(z) C_j(z) \\ D_{j+1}(z) &= \tilde{G}(z) C_j(z) \end{aligned} \tag{2.39}$$

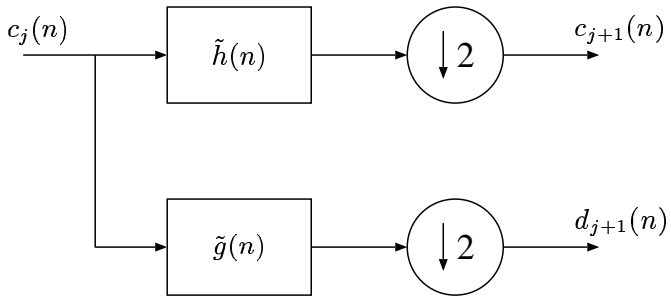


Figure 2.1. Block diagram for the FWT analysis stage.

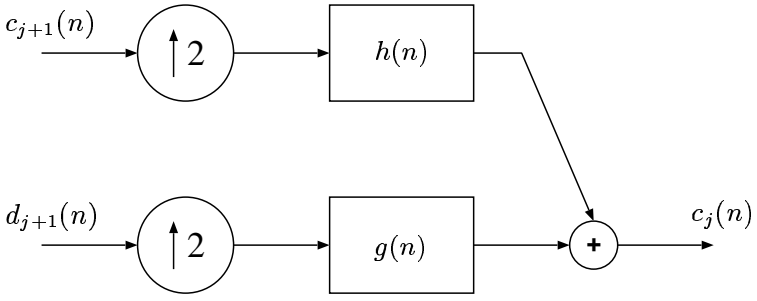


Figure 2.2. Block diagram for the FWT synthesis stage.

The inverse transform, can also be derived from the MRA structure. From equations (2.29) & (2.30), it is evident that each reconstruction stage needs to add the detail subbands ($d_{j+1}(n)$) back to the coarse approximation subband ($c_{j+1}(n)$). Each *synthesis*

stage thus requires upsampling each subband by 2, followed by filtering with the decomposed subbands with the synthesis filters, $h(n)$ and $g(n)$, and adding the results. This is illustrated in figure 2.2, and the equation is

$$C_j(z) = H(z)C_{j+1}(z) + G(z)D_{j+1}(z) \tag{2.40}$$

The synthesis stages are repeated over all levels to recover the original signal $c_0(n)$. The entire analysis and synthesis process in a k-level FWT is illustrated in figure 2.3. The number of computations is halved with an increase in the level of the transform, as a direct consequence of the reduced data from downsampling. This reduction in complexity with levels leads to the high efficiency of the algorithm. It is not difficult to show that the Fast Wavelet Transform has an overall computational complexity of $O(N \log N)$. Since the analysis and synthesis stages employ two filters (low- and high-pass), the FWT system is commonly called a 2-channel filter bank in engineering literature.

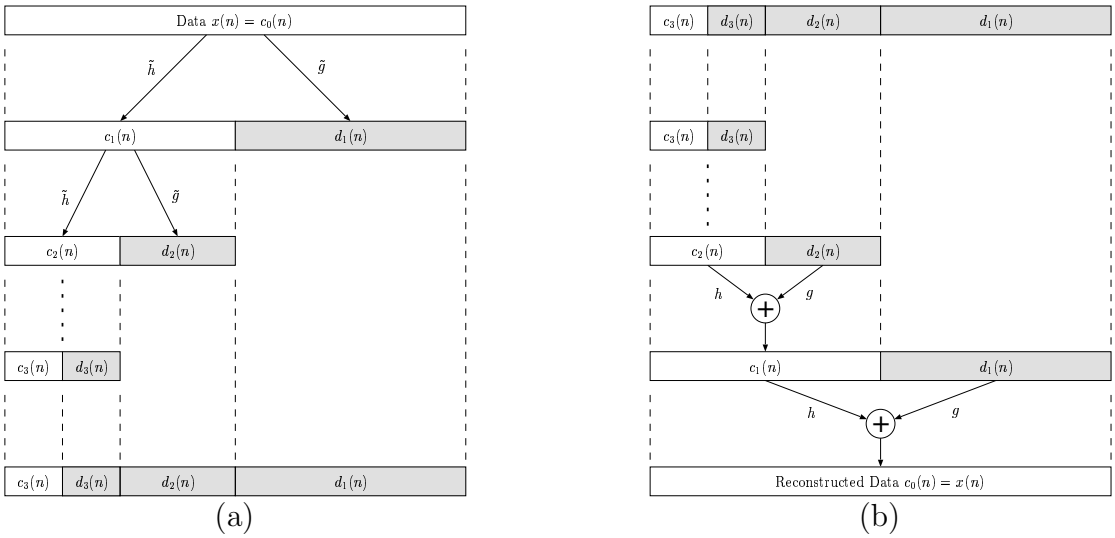


Figure 2.3. The (a) forward and (b) inverse Fast Wavelet Transforms for a 1-D signal $c_0(n)$. The filters $\tilde{h}(n), \tilde{g}(n)$ are known as the analysis filters, and $h(n), g(n)$ are the synthesis filters. In the orthonormal case, $h(n), \tilde{h}(n)$ and $g(n), \tilde{g}(n)$ are time-reversed versions of each other.

2.2.2 Perfect Reconstruction Filters

Due to the downsampling and upsampling operations in the transform process, the synthesis and analysis filters must satisfy certain conditions if the reconstructed signal is to

be the same as the original signal. Essentially, these filters must be constructed such that the information discarded by the downsampling can be recovered in the synthesis process. The conditions for this *perfect reconstruction* are

$$\tilde{H}(z)H(-z) + \tilde{G}(z)G(-z) = 0 \tag{2.41}$$

$$\tilde{H}(z)H(z) + \tilde{G}(z)G(z) = 2z^{-l}, \quad l \in \mathbf{Z} \tag{2.42}$$

The integer l measures the total latency of the system combined from the analysis and synthesis processes. The first condition removes aliasing in the reconstructed signal, while the second condition ensures signal integrity is preserved. For simplicity, it is usual practice to choose the filters in such a way that the anti-aliasing condition is automatically satisfied. This can be achieved by:

$$H(z) = \tilde{G}(-z), G(z) = -\tilde{H}(-z) \tag{2.43}$$

This implies that $h(n)$ is the reversed, alternate-signed version of $\tilde{g}(n)$, and similarly for the pair $g(n), \tilde{h}(n)$. In the orthonormal case, $h(n) = \tilde{h}(-n)$ and $g(n) = \tilde{g}(-n)$. If we define $P(z) = z^l H(z)\tilde{H}(z)$, then the perfect reconstruction requirement becomes

$$P(z) + P(-z) = 2 \tag{2.44}$$

This implies that $p(n)$ is a *half-band* filter; all its even coefficients are zero, with the exception of $p(0)$, which must be 1. The odd coefficients are free design variables. In practice, the wavelet filters are usually derived by first designing a particular half-band filter $p(n)$. The analysis and synthesis low-pass filters are then obtained by applying spectral factorisation to $p(n)$.

Despite their central roles in the development of wavelet theory, the scaling and wavelet functions do *not* appear anywhere in the FWT algorithm. In other words, it is unnecessary to explicitly construct these functions in order to compute the wavelet transform coefficients; all that is needed are the analysis filters, $\tilde{h}(n)$ and $\tilde{g}(n)$. The same applies for the inverse transform; only the synthesis filters $h(n)$ and $g(n)$ are needed.

A key observation of the FWT is that the decomposition is *not* shift invariant like the CWT. That is, an odd sample shift in the input signal results in a drastically different set of transform coefficients. This is purely a consequence of the downsampling operations, since filtering with the analysis filters is a shift invariant process. In many practical situations, this shift variance is an undesirable aspect of the FWT. Researchers have spent significant efforts in overcoming this problem, leading to a variety of different approaches.

2.2.3 Implementation Issues

There is an important practical consideration in implementing the FWT algorithm. To start off the pyramidal decomposition, it is necessary to first obtain the lowest level coefficients, $c_{o,j}$. In almost all instances, these are assumed to be the data on hand. In effect, this is an approximation to the actual coefficients, which should be obtained from inner product computations, $c_{0,n} = \langle f(t), \psi_{0,n}(t) \rangle$. The explicit computation of these inner products are expensive, and the errors from the simple approximations are acceptable for many applications.

Direct implementations of the FWT are inefficient. To discard half of the filtered samples, as suggested by figure 2.1, is a complete waste of computational time. In the synthesis process, upsampling prior to filtering implies half of the samples will be zero, and hence will not contribute to the inverse transform. Instead, *polyphase filter* [69] implementations are much better, because they eliminate the need to perform unnecessary computations. In these algorithms, the filters are separated into two halves, one containing the odd samples, and the other contains the even ones. In the z-domain, this is represented by

$$H(z) = H_e(z^2) + z^{-1} H_o(z^2) \tag{2.45}$$

where $H_e(z)$ and $H_o(z)$ are the polyphase filters which contain the even and odd components, respectively. For analysis, the level-j coarse coefficients, $C_j(z)$ are decomposed into its polyphase components, $C_{j,e}(z)$ and $C_{j,o}(z)$, before filtering with the corresponding polyphase filters. The filtered sequences are then summed to yield the level-$j+1$ coarse and detail coefficients:

$$\begin{aligned} C_{j+1}(z) &= \tilde{H}_e(z) C_{j,e}(z) + \tilde{H}_o(z) C_{j,o}(z) \\ D_{j+1}(z) &= \tilde{G}_e(z) C_{j,e}(z) + \tilde{G}_o(z) C_{j,o}(z) \end{aligned} \tag{2.46}$$

For synthesis, the level-j coarse and detail coefficients, $C_j(z), D_j(z)$, are multiplied by their corresponding polyphase filters before being summed to produce the level-$j-1$ coarse coefficients:

$$\begin{aligned} C_{j-1,e}(z) &= H_e(z) C_j(z) + G_e(z) D_j(z) \\ C_{j-1,o}(z) &= H_o(z) C_j(z) + G_o(z) D_j(z) \end{aligned} \tag{2.47}$$

If we define the analysis and synthesis polyphase matrices to be

$$Analysis: \quad \widetilde{\mathbf{H}}_{\mathbf{p}}(z) = \begin{bmatrix} \tilde{H}_e(z) & \tilde{H}_o(z) \\ \tilde{G}_e(z) & \tilde{G}_o(z) \end{bmatrix} \tag{2.48}$$

$$Synthesis: \quad \mathbf{H}_p(z) = \begin{bmatrix} H_e(z) & H_o(z) \\ G_e(z) & G_o(z) \end{bmatrix} \tag{2.49}$$

then the analysis and synthesis steps in equations (2.46) and (2.47) can be written as

$$\begin{bmatrix} C_{j+1}(z) \\ D_{j+1}(z) \end{bmatrix} = \widetilde{\mathbf{H}}_{\mathbf{p}}(z) \begin{bmatrix} C_{j,e}(z) \\ C_{j,o}(z) \end{bmatrix} \tag{2.50}$$

$$\begin{bmatrix} C_{j-1,e}(z) \\ C_{j-1,o}(z) \end{bmatrix} = \mathbf{H}_p(z) \begin{bmatrix} C_j(z) \\ D_j(z) \end{bmatrix} \tag{2.51}$$

The block diagrams of a polyphase implementation are shown in figures 2.4 and 2.5.

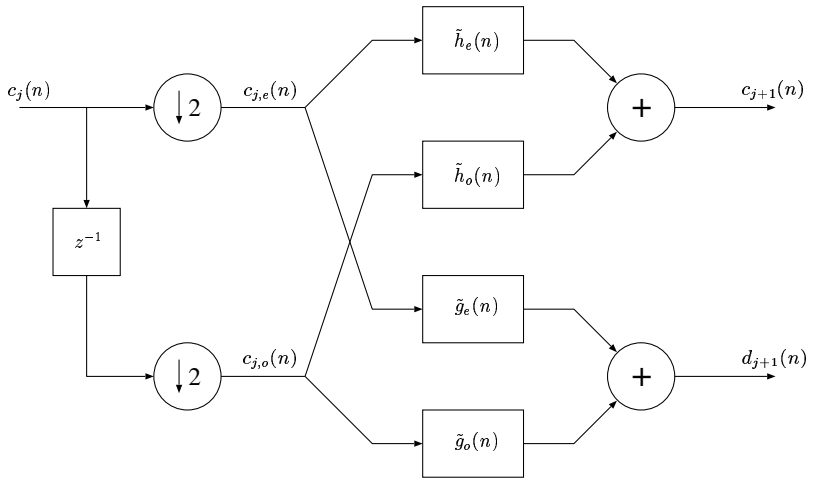


Figure 2.4. Analysis polyphase filter bank.

Another issue with the FWT algorithm deals with boundary handling. Ideally, with signals of infinite length, this issue does not arise. In practice, an application often has a data set of finite size, $x(n)$, which leads to problems with coefficients at the beginning and end of the data set. In effect, it runs out of data for computing the convolution sum for the filters. Historically, there has been several different approaches toward this problem. Cohen *et al.* [13] proposed to construct separate wavelet bases for the boundary data, effectively splitting the wavelet transform into three pieces with different analysis and synthesis filters. This technique falls in the more general category of time-varying filter

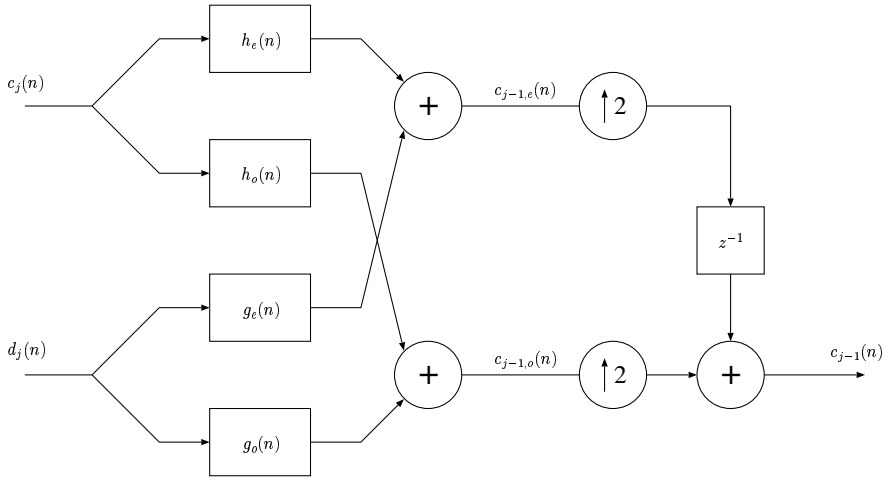


Figure 2.5. Synthesis polyphase filter bank.

banks. The problem with such an approach is that the boundary handling requires great care and computational expense. Simpler methods are preferred for this reason. The most straight-forward method would be to extend the input data beyond their finite extent when computing the filter convolutions. In the literature, there has been three main extension strategies proposed: zero, periodic and symmetric. Theoretically, symmetric is by far the most superior, because it does not lead to artificial discontinuities at the boundaries. The presence of these discontinuities produces large magnitude wavelet coefficients near the data boundaries. Periodic extension is akin to using circular convolution in the filter banks; perfect reconstruction can be guaranteed if the original data has an even number of samples for dyadic wavelet transforms, which use a downsampling rate of 2. However, symmetric extension must only be used with symmetric filters, which means biorthogonal wavelets. Asymmetric filters cannot produce perfect reconstruction with the symmetric extension technique; the same problem does not apply to zero or periodic extensions. In fact, there are further conditions that the analysis and synthesis filters must satisfy if the symmetrically-extended wavelet transform can be reconstructed. Several researchers [1],[38],[3],[4] have done a lot of work on symmetric extensions, and most of these are very general in scope. These include the analysis of the boundary conditions for a vast range of symmetries and downsampling rates. Within the scope of this thesis, only the dyadic wavelets case is relevant. The requirements for implementing symmetric extensions for such a system will be briefly described below.

2.2.4 Symmetric Extended FWT

Let's first consider the different symmetries that can exist for finite-length sequences. One distinction is about the position of the point of symmetry. A sequence can either be symmetric about a sample or about a point midway between successive samples. These are referred to as whole-sample or half-sample symmetries, respectively. The other division between symmetries is the sign; basically, a symmetry can be either symmetric or anti-symmetric. The former refers to the case where samples are repeated on either side of the centre of symmetry, while the latter has their samples' signs inverted. The two distinctions naturally lead to the four different symmetries: whole-sample symmetric (WS), whole-sample anti-symmetric (WA), half-sample symmetric (HS) and half-sample anti-symmetric (HA). These are illustrated in figure 2.6. Obviously, these descriptions apply equally to the data and filters alike in our subsequent discussions.

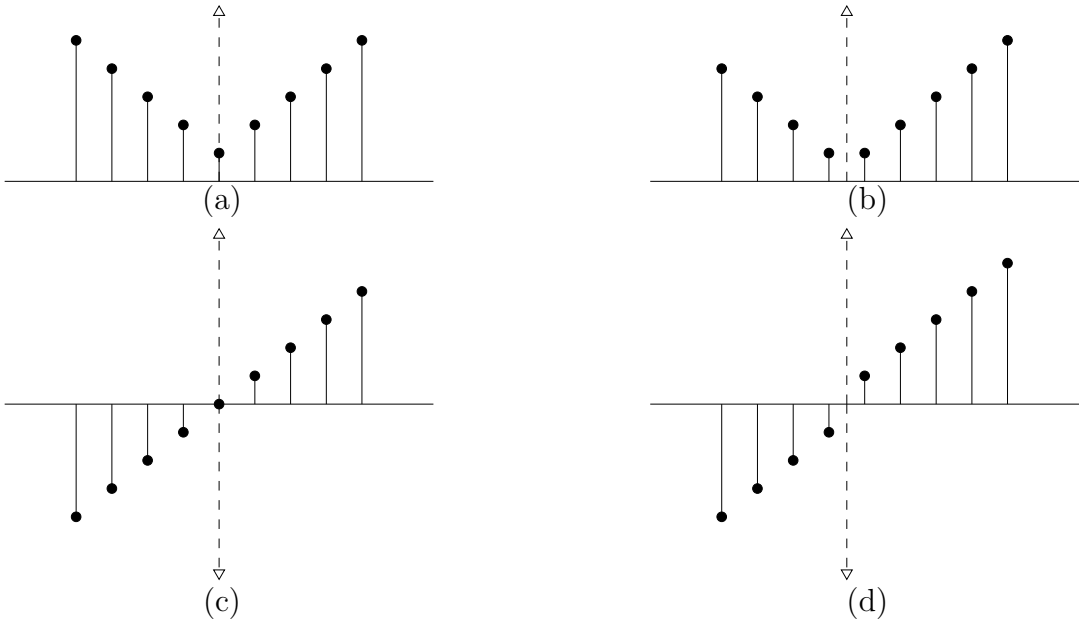


Figure 2.6. Four different types of symmetric extensions: (a) WS, (b) HS, (c) WA, (d) HA

Once having classified all the different possible symmetries, we can turn our attention to the extension strategies for a given data boundary. A symmetric extension is completely described the by the sign of the extension, and the number of repeated samples at the boundary. For a finite-length sequence, extensions are required at both ends. Adopting Brislawn's notation [4], this can be presented by $E_c^{(m,n)}$, where c can either be symmetric (s) or anti-symmetric (a), and m, n are positive integers denoting the number of repeated

samples at the left and right boundaries, respectively. In practice, m and n are usually either 1 or 2. A few examples of different extensions are illustrated in figure 2.7. Once a finite-length signal has been extended, it can be subjected to the filtering in a standard analysis filter bank (figure 2.1).

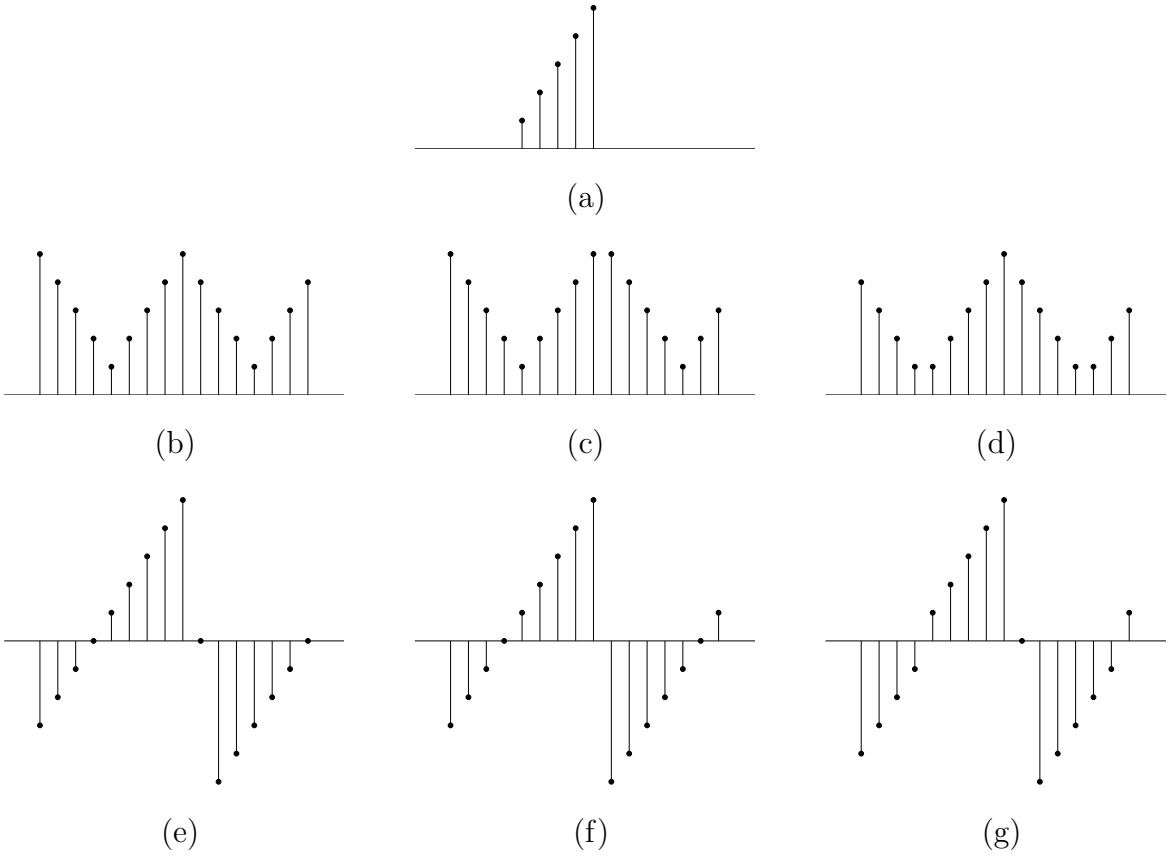


Figure 2.7. Examples of symmetrically extended samples. (a) Original data sequence; (b) $E_s^{(1,1)}$; (c) $E_s^{(1,2)}$; (d) $E_s^{(2,1)}$; (e) $E_a^{(1,1)}$; (f) $E_a^{(1,2)}$; (g) $E_a^{(2,1)}$

In determining the appropriate criteria for a perfectly reconstructing filter bank, it is first necessary to decide the *expansiveness* of the decomposition. A transform is said to be *non-expansive* if the total number of coefficients in all the subbands is equal to the original number of data samples. The difficulty of a perfectly reconstructing, symmetrically extended wavelet transform is really with the desire to have non-expansive decompositions. With expansive transforms, there is more freedom to choose due to the redundancy in the subbands. The shift variant nature of a FWT mandates great care in selecting which symmetrically extended transform coefficients must be retained. This task is simple for periodic extensions on a N-sample signal, because the data is periodically

repeated, therefore any $\frac{N}{2}$ consecutive samples in each subband must contain sufficient information for perfect reconstruction. However, with a symmetrically extended transform, the situation is much more complicated, especially if higher level transforms are needed. Due to the recursive nature of the FWT algorithm, it is necessary that all the subbands are individually symmetric about their boundaries. Only certain combinations of input extension strategies and filter symmetries and phases can produce symmetric subbands. It is then a matter of systematically working through all the different combinations to determine which combinations are allowable. Since our attention is restricted to 2-channel filter banks only (dyadic wavelets), there are not too many cases to consider. Further complications arise among the allowable combinations; filters with non-zero group delays require a non-zero shift in the filtered signal before downsampling, or else the subband coefficients will not align properly. Depending on the symmetry of the analysis filters, the coarse and detail subbands may have different lengths; this can lead to many complications depending on the particular application. For synthesis, the extension strategy must be matched to the analysis filter and extension combination in order to achieve perfect reconstruction. In [4], Brislawn has tabulated all the different possible combinations and these will not be repeated here. In practice, restrictions are often placed on the filters and signals, which then limits the allowable extension strategies. In our work, we only consider dyadic wavelet transforms on even-length signals. It is also much more convenient if the subbands produced have the same size; that is, a signal of length N should decompose into two subbands of $\frac{N}{2}$ coefficients each. The extension strategies then solely depend on the symmetry of the analysis filters in use. The relevant cases are:

- if filter $\tilde{h}(n)$ or $\tilde{g}(n)$ is WS, then an $E_s^{(1,1)}$ extension must be used; reconstruction extension is $E_s^{(1,2)}$ or $E_s^{(2,1)}$, for filter group delay (k) being even or odd, respectively. Subband shift after analysis filtering is either $\frac{k}{2}$ or $\frac{k+1}{2}$ for even or odd k, respectively.
- if filter $\tilde{h}(n)$ or $\tilde{g}(n)$ is HS or HA, then an $E_s^{(2,2)}$ extension must be used; reconstruction extension is $E_s^{(2,2)}$ or $E_a^{(2,2)}$, for HS and HA filters, respectively. Filter group delay, k, must have the form $k = 2m + \frac{3}{2}$. Subband shift after analysis filtering is $\frac{k+1/2}{2}$.

Note that the only possible analysis filter pairs, $\{\tilde{h}(n), \tilde{g}(n)\}$ must either have the form WS/WS or HS/HA. Equipped with these rules, it is then a matter of applying the appropriate delays and extensions in the filter banks to yield a perfectly reconstructing,

symmetric extended wavelet transform. The modified analysis and synthesis filter banks are illustrated in figures 2.8 and 2.9 below.

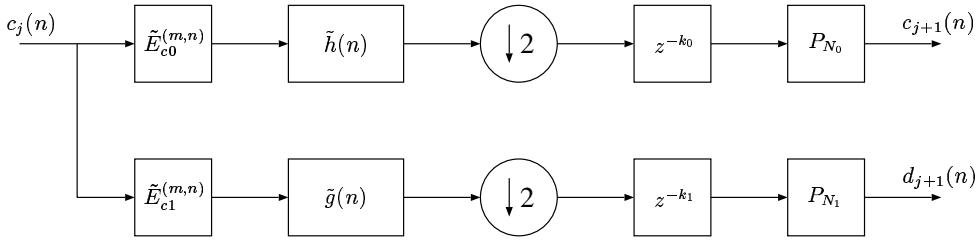


Figure 2.8. The modified FWT analysis bank for symmetric extensions.

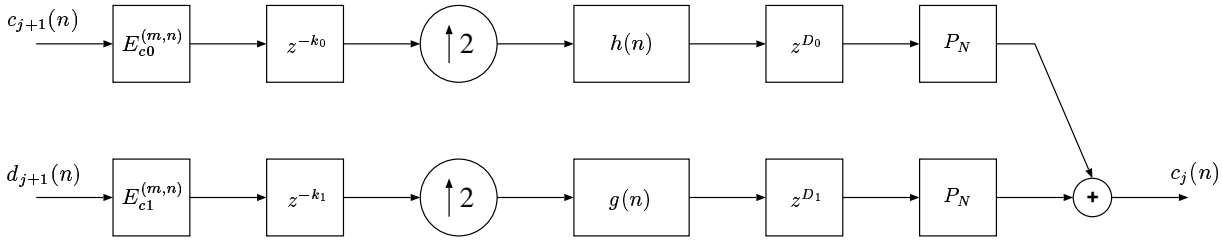


Figure 2.9. The modified FWT synthesis bank for symmetric extensions.

2.3 Wavelet Transforms by Lifting

Section 2.2 discussed the Fast Wavelet Transform and the common polyphase filter-bank implementation. An alternative approach to wavelet transforms has surfaced in more recent literature. It abandons the traditional focus on frequency analysis of filters, and instead focuses on the filtering as computed in the time domain. This new technique is called *lifting*, and will be discussed briefly below.

2.3.1 The Lifting Theorem

The lifting scheme for constructing biorthogonal wavelets has been well-studied [7], [19], [72], [73], [70], [71]. The idea behind the lifting scheme is based on the *lifting theorem*:

Theorem 1 (Lifting) *Assume that the set of filters* $\{H_0(z), \tilde{H}_0(z), G_0(z), \tilde{G}_0(z)\}$ *is biorthogonal. That is, they satisfy the following:*

$$\tilde{H}_0(z)H_0^*(z) = \tilde{G}_0(z)G_0^*(z) = 1 \tag{2.52}$$

$$\tilde{G}_0(z)H_0^*(z) = \tilde{H}_0(z)G_0^*(z) = 0 \tag{2.53}$$

$$\text{and} \quad H_0^*(z)\tilde{H}_0(z) + G_0^*(z)\tilde{G}_0(z) = 1 \tag{2.54}$$

A new set of biorthogonal filters $\{H_1(z), \tilde{H}_1(z), G_1(z), \tilde{G}_1(z)\}$ *can be found as*

$$\begin{aligned} H_1(z) &= H_0(z) \\ \tilde{H}_1(z) &= \tilde{H}_0(z) + S(z^2)\tilde{G}_0(z) \\ G_1(z) &= G_0(z) - S^*(z^2)G_0(z) \\ \tilde{G}_1(z) &= \tilde{G}_0(z) \end{aligned} \tag{2.55}$$

where $S(z)$ *is a trigonometric polynomial.*

In polyphase notation, this is equivalent to the following:

Corollary 1 *A biorthogonal polyphase matrix pair,* $\mathbf{H}_p(z), \widetilde{\mathbf{H}}_\mathbf{p}(z)$*, can be written in the form*

$$\begin{aligned} \mathbf{H_p}(z) &= \mathbf{H_p}^0(z) \begin{bmatrix} 1 & s(z) \\ 0 & 1 \end{bmatrix} \\ \widetilde{\mathbf{H}}_\mathbf{p}(z) &= \widetilde{\mathbf{H}}^0_\mathbf{p}(z) \begin{bmatrix} 1 & 0 \\ -s(z^{-1}) & 1 \end{bmatrix} \end{aligned} \tag{2.56}$$

where the matrices $\mathbf{H_p}^0(z), \widetilde{\mathbf{H}}^0_\mathbf{p}(z)$ *are another biorthogonal pair.*

Equation (2.56) remains true if the filters are replaced by their duals. Such a step is known as *dual lifting*. By combining lifting with dual lifting - a process known as *cake-walking* - one can obtain any set of finite length biorthogonal filters from a set of shorter ones. In particular, one can begin with a trivial biorthogonal set, and apply cake-walking to arrive at a more useful set of filters. An example of a commonly used trivial orthogonal set is the Lazy set:

$$\tilde{H}(z) = H(z) = 1 \tag{2.57}$$

$$\tilde{G}(z) = G(z) = z^{-1} \tag{2.58}$$

These filters simply separate the input stream into odd and even components, i.e. its polyphase representation. It is shown in [19] that the Euclidean algorithm can be used to decompose any biorthogonal filter into a product of a finite number of lifting and dual lifting factors. The division algorithm is repeatedly applied on the analysis low-pass polyphase filters until a trivial common divisor is arrived at. The analysis high-pass polyphase filters are obtained from this factorisation by further lifting steps. Another way to interpret lifting is from the perspective of matrix factorisation. From equations (2.48) and (2.50), it is evident that the FWT can be regarded as a matrix multiplication of polynomials. The lifting scheme simply factors $\widetilde{\mathbf{H}}_{\mathbf{p}}(z)$ and $\mathbf{H}_{\mathbf{p}}(z)$ into a product of simpler polyphase matrices. More specifically, these simple matrices have the form:

$$\mathbf{H}_i(z) = \begin{bmatrix} 1 & 0 \\ s(z) & 1 \end{bmatrix} \quad \text{or} \quad \mathbf{H}_i(z) = \begin{bmatrix} 1 & s(z) \\ 0 & 1 \end{bmatrix} \tag{2.59}$$

where $s(z) = z^{-k}(a + bz^{-1}), a, b \in \mathbf{R}$ and $k \in \mathbf{Z}$. In the time domain, these lifting steps are equivalent to the following simple operations

$$\begin{cases} C_{j+1,e}(n) &= C_{j,e}(n) \\ C_{j+1,o}(n) &= C_{j,o}(n) + a_j C_{j,e}(n-k) + b_j C_{j,e}(n-k-1) \end{cases} \tag{2.60}$$

or

$$\begin{cases} C_{j+1,e}(n) &= C_{j,e}(n) + a_j C_{j,o}(n-k) + b_j C_{j,o}(n-k-1) \\ C_{j+1,o}(n) &= C_{j,o}(n) \end{cases} \tag{2.61}$$

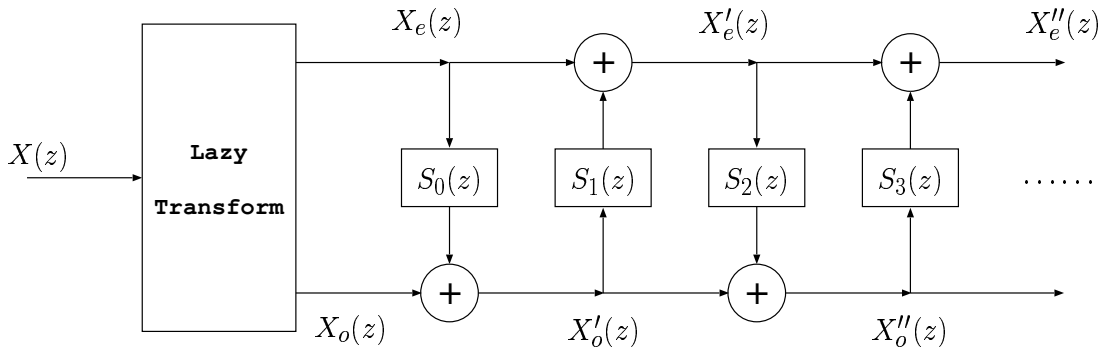


Figure 2.10. One stage of the fast lifted wavelet transform (FLWT). The two-tap filters are $S_i(z) = z^{-k}(a_i + b_i z^{-1})$

Figure 2.10 illustrates these operations. The fast lifted wavelet transform (FLWT) involves three steps

1. Compute the lazy wavelet
2. Repeat the operations in equation (2.60) or (2.61) for all the lifting steps in the filter factorisation
3. Re-iterate steps 1. & 2. on subbands to yield final transform

It should be noted that the lifting scheme does not produce any wavelets that cannot be somehow obtained from more orthodox constructions (e.g. factorisation of half-band filter $p(n)$). What it provides is an alternative insight into the understanding of wavelets. In addition, it provides a foundation for the construction of what is known as second generation wavelets. These are generalisations of conventional wavelets, where the basis functions are no longer translations and dilations of a single function. However, these second generation wavelets will preserve some of the important properties of first generation wavelets; (bi-)orthogonality, time-frequency localisation and a multiresolution nature. Second generation wavelets are an unknown field, and hence they are not discussed in this thesis.

2.3.2 Computational Advantages

The main advantage of the fast lifted wavelet transform is that it is efficient to implement in digital hardware. The form of equation (2.60) and (2.61) shows that the calculations in each step of the transform can be done completely *in situ*. This lowers the memory requirements of the transform. In fact, the only memory overhead in the algorithm is for the lifting steps and the temporary memory during the polyphase decomposition (Lazy wavelet transform). For long filters, the FLWT algorithm needs approximately half the number of operations when compared to the traditional filter bank algorithm [19]. In practice, the simple, lattice-like computations in equations (2.60) and (2.61) in each lifting step is very easy to implement and requires no programming overheads such as loops. The resultant code is very fast to execute. It is believed that the simple structure of the FLWT makes it highly suitable for high performance implementations.

2.4 Wavelet Packets and Best Basis

The wavelet packet representation is a generalisation of the basic wavelet transform. Instead of performing decomposition on the coarse subband at each scale in the FWT, wavelet packet analysis performs decomposition on either or both of the subbands. Mathematically, this is the expansion of the signal in terms of the translations of the wavelets at the lowest resolution. In contrast, conventional wavelet transforms express the signal as a superposition of the scaling function at the lowest resolution and wavelet functions at all resolutions. This can be diagrammatically represented by a binary decomposition tree; the conventional FWT would result in a tree with only one side. Examples of such trees is shown in figure 2.11.

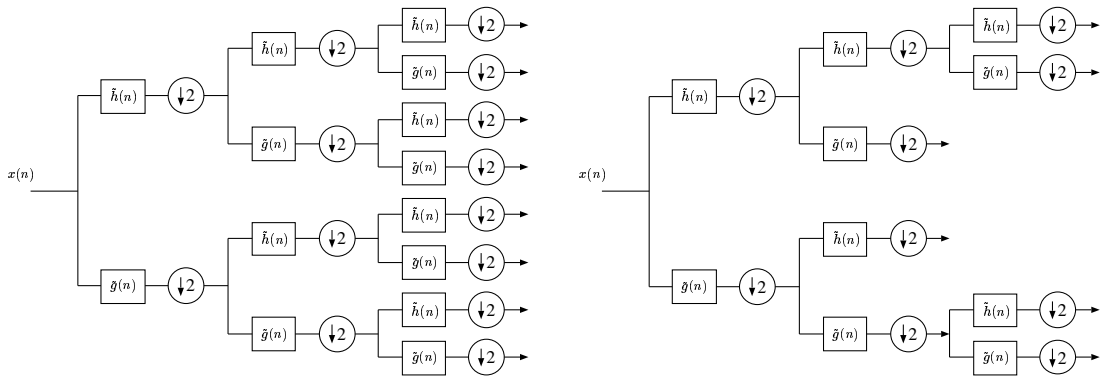


Figure 2.11. Examples of the two wavelet packet decompositions.

A particular advantage of wavelet packets is its flexibility. For each subband in the decomposition (i.e. at each node in the binary tree), it is an entirely arbitrary choice whether to further decompose the subband or not. In practical applications, this decision may rest on whatever additional condition imposed by the problem. This idea has led to the Best Basis selection algorithm, first proposed by Wickerhauser [83]. In an algorithm, the decomposition decision at each node is determined by the difference in some cost function, such as entropy, between a parent and its children nodes. Ultimately, the aim of the algorithm is to find the most compact representation of the data. The adaptive, best basis procedure has been applied to several problems with good results [83, 58, 21]. Variations of Best Basis are numerous in the literature, and it is beyond the scope of this thesis to provide a detailed description of these all. The original work contained in thesis does not use wavelet packets, but it is important as a potential future research direction.

2.5 Multidimensional Wavelets

Thus far, only one-dimensional wavelets have been discussed. In image processing applications, where the data are inherently two dimensional, the theory of wavelets must be generalised accordingly. Traditionally, Gabor filters have been popular in image analysis applications; a basic 2-D Gabor filter can be written as

$$h_0(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp(-\frac{1}{2}(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}))\exp(j2\pi f_0 x) \tag{2.62}$$

where σ_x and σ_y are the filter bandwidths along the x and y directions, and f_0 is the centre frequency. A complete basis of 2-D Gabor wavelets can be generated by applying dilation and rotation operations on the basic Gabor function:

$$\begin{aligned} h(x,y) &= Ah_0(x',y') \\ x' &= A(x\cos\theta + y\sin\theta) \\ y' &= A(-x\sin\theta + y\cos\theta) \end{aligned} \tag{2.63}$$

where θ is an arbitrary rotation, and A is a scale factor. Researchers have found that dyadic dilations ($A = 2^m$) with 4 or 6 evenly-spaced rotations to perform well. Usually, filter bandwidths are designed such that the set of filters' 3dB contours in their frequency responses touch each other.

Traditionally, the approach in 2-D wavelets is to construct them as direct tensor products of 1-D wavelets. Mathematically, this means that the 2-D scaling function is formed by

$$\phi(x,y) = \phi(x)\phi(y) \tag{2.64}$$

where $\phi(x)$ is a valid 1-D scaling function. From this formulation, there are *three* 2-D wavelet functions, resulting from the different tensor product combinations.

$$\psi_{lh}(x,y) = \phi(x)\psi(y) \tag{2.65}$$

$$\psi_{hl}(x,y) = \psi(x)\phi(y) \tag{2.66}$$

$$\psi_{hh}(x,y) = \psi(x)\psi(y) \tag{2.67}$$

The subscripts *hl, lh, hh* correspond to the action of the high- and low-pass filters in the horizontal and vertical directions. One consequence of the tensor product is that these particular wavelet functions have three preferential directions. The wavelet $\psi_{lh}(x,y)$

responds strongly to vertical edges, while $\psi_{hl}(x, y)$ and $\psi_{hh}(x, y)$ respond to horizontal and diagonal edges, respectively. An example is shown in figure 2.12. The wavelet transform of the "Lenna" image is separated into four distinct regions, or subbands. These subbands are critically downsampled, by a factor of 4 in this case, so as to preserve the total number of pixels in the image. From now on, the term 'separable 2-D wavelets' will refer to wavelets obtained from tensor products of 1-D wavelets.

Figure 2.12. The original Lenna image, 256×256, 8-bit gray level, and the image after a 2-level fast wavelet transform. The filter used was the tensor product from a pair of Daubechies 9-7 biorthogonal pair in both the horizontal and vertical dimensions. The LL_2 subband coefficients have been reduced to illustrate the other subbands.

An alternative to the tensor product approach is to start from a true 2-D multiresolution analysis. This approach gives true, non-separable 2-D wavelets [12]. All of the properties of a 1-D MRA (equations (2.10)-(2.15)) can be directly generalised to 2-D, with the dilation factor (2 in the above definition) replaced by a dilation matrix, $\mathbf{D}$. A dilation matrix is simply a 2×2 matrix with integer entries. The determinant of this matrix gives the downsampling factor. An additional requirement on this dilation matrix is that its eigenvalues must have magnitude strictly greater than 1, to ensure a true dilation in all directions. Most of the efforts on developing non-separable wavelets have concentrated on using dilation matrices with determinant 2. A good example of such a downsampling scheme is the *quincunx* grid, which offers some theoretical advantages over a separable grid, such as isotropy.

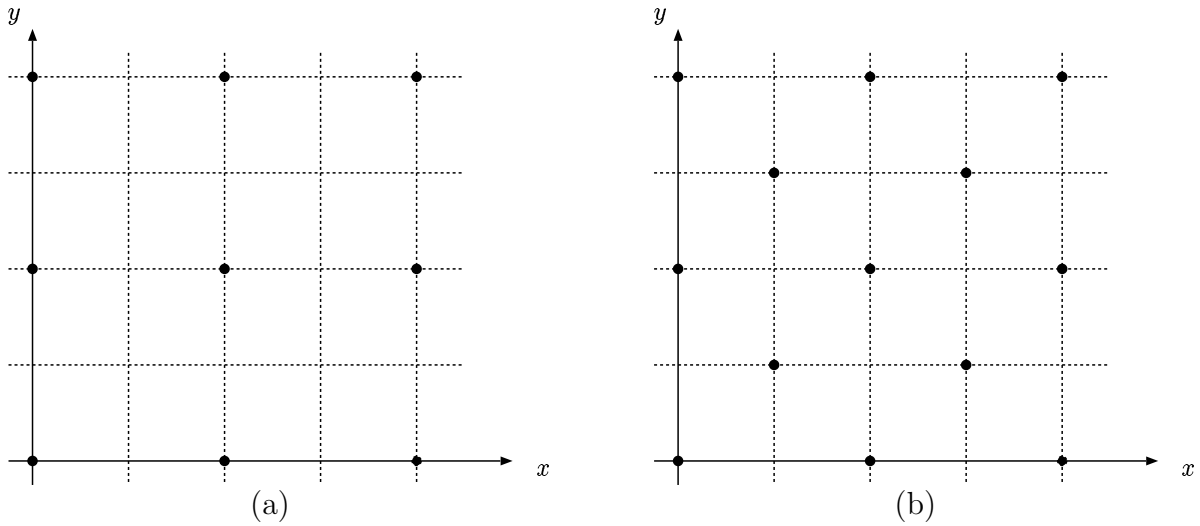


Figure 2.13. (a) A separable grid (downsampling factor 4); (b) the quincunx grid (downsampling factor 2)

Almost all image processing applications to date use separable 2-D wavelets. The main area of application has been in image compression, where outstanding quality with high compression ratios has already been demonstrated. There are many benefits to using separable 2-D wavelets. For instance, they inherit properties such as finite support, orthogonality and smoothness from its root 1-D wavelets, therefore eliminating the need to repeat complicated and difficult analysis on the 2-D wavelets themselves. The use of tensor products is equivalent to the use of separable 2-D filters in a filter bank implementation. This implies that the 2-D wavelet transforms can be synthesised as a cascade of 1-D transforms, which is computationally efficient. On the other hand, fully two-dimensional filter banks are computationally expensive to implement.

However, there are several disadvantages associated with separable 2-D wavelets. First of all, they exhibit strong preferences to horizontal, vertical and diagonal frequencies in an image. For many image processing problems, a more isotropic analysis is preferred. In addition, the use of separable filters gives the designer far fewer degrees of freedom than non-separable ones for filters of the same size. This has important consequences in the properties of the resulting wavelets. One of the most notable consequence is the fact that orthogonal, symmetric wavelets with compact support is possible for non-separable wavelets, but not for the separable case. Therefore, there are good reasons to study true, non-separable 2-D wavelets for image processing applications. However, the mathematics

for non-separable 2-D wavelets are very difficult, and there are also unfavourable implementation issues. These are important obstacles to overcome if non-separable wavelets are to be in wide-spread use. In this thesis, separable 2-D wavelets are used throughout, due to the abovementioned difficulties of non-separable 2-D wavelets. The complex wavelet transform discussed in the next section overcomes some of the limitations of basic separable 2-D wavelets, without the analytical and mathematical complexity of non-separable 2-D wavelets.

2.6 Dual-Tree Complex Wavelet Transform

The major disadvantage of a conventional FWT, with its critical sub-sampling, is its lack of *shift invariance*. The FWT is highly sensitive to the precise positioning of the input signal, in the sense that a slight shift in the input typically results in a drastic change in the distribution of energy among the subbands. This is a highly undesirable characteristic for image analysis and image understanding tasks. A robust image analyser must be able to correctly identify the same features irrespective of their position in the image. This is particularly true in texture analysis, where the very notion of textures implies an arrangement of shifted patterns and hence mandates shift invariant techniques in their analysis. A brute force solution to the shift variance of the FWT would be to eliminate all down-sampling from the decomposition, resulting in the wavelet frame representation. This full redundancy greatly increases the computational burden on the system. In many image analysis applications, the massive increase in the number of coefficients and computational complexity propagates through all subsequent processing steps, thus raising the system cost significantly. A similar problem exists for Gabor transforms, which are also highly redundant representations. However, in pattern recognition problems, a certain degree of redundancy is often useful. Generally, redundancy can mask noise in the extracted features which could otherwise interfere with the decision making process. It is well known that some redundancy usually leads to better classification performance. Thus, the degree of redundancy in a representation is a tradeoff between the desire to minimise computational costs and to increase recognition reliability.

The separability of 2-D FWT greatly restricts the directional selectivity of this representation. Directional information is very useful in many image analysis applications,

where important features are present in several spatial orientations. The FWT can only provide three preferred directions, oriented at 0°, 45° and 90° to the horizontal. This is even worse when one considers that the diagonal (HH) subbands contain information along both the $+45°$ and $-45°$ directions. In other words, it is impossible to isolate information along $+45°$ or $-45°$ directions separately, thus effectively restricting the FWT preferred directions to just the horizontal and vertical. This limitation is illustrated in figure 2.14. In comparison, 2-D Gabor decompositions produce subbands that can be oriented in any number of arbitrary directions, since θ is a free parameter in a Gabor function. Typical applications of Gabor filtering [32], [15] use basis functions aligned to four different orientations, even though any number of directions are possible, since each additional direction increases the redundancy in the decomposition.

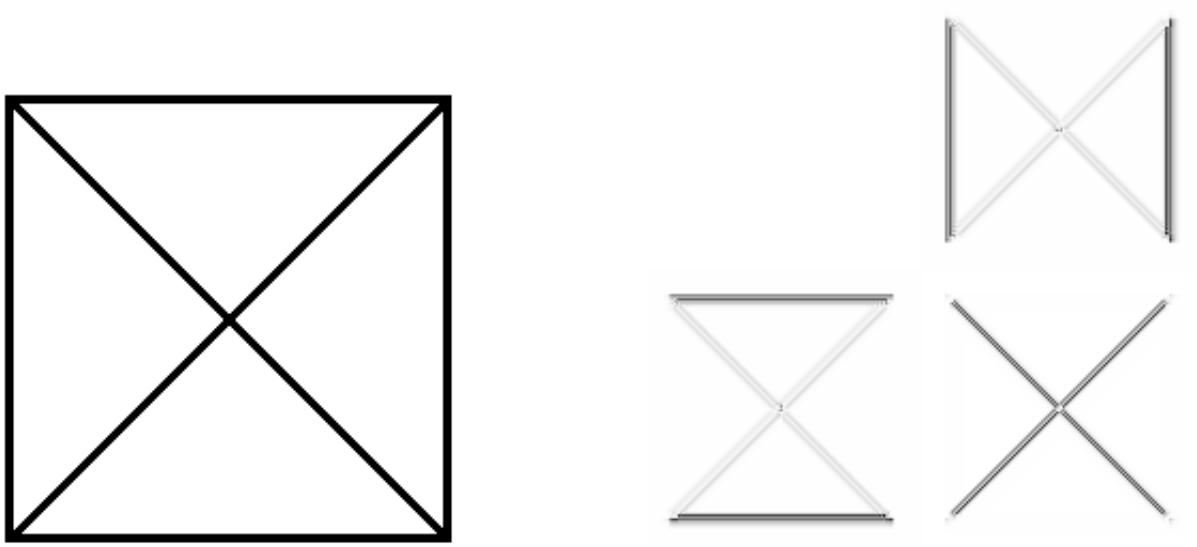


Figure 2.14. The directional limitations of the FWT. A 1-level, Daubechies 16 orthonormal wavelet transform is applied to the figure on the left. Its transform is shown on the right; the LL subband is removed to better highlight the high-frequency subbands. The preferred directions at $0°, 90°$ and $\pm 45°$ are clearly visible in the transform image.

The shift invariance and poor directional selectivity of the FWT is due to the use of real-valued filters. It has been discovered that complex-valued filters can provide approximate shift invariance [44]. The key to this technique is the presence of real and imaginary parts in the wavelet coefficients. With the extra dimension, it is possible to ensure that the magnitude of the complex response vary slowly with respect to input shift, therefore producing approximate shift invariance in the downsampled signals. On the other hand,

it is the phase of these coefficients that varies rapidly with respect to input shift. These conditions can be achieved if the real and imaginary parts of the scaling and wavelet functions have opposing symmetries and their centres are shifted with respect to each other. In essence, the peaks of one part (real or imaginary) compensates for the troughs of the other, so that rapid variations in the magnitude are eliminated. The resulting magnitudes have Gaussian-like shape, which offer good time-frequency characteristics. This is illustrated in figure 2.16.

Due to the presence of real and imaginary parts, a complex wavelet transform produces twice the amount of data as a normal FWT. In other words, the complex wavelet transform is a 2-times redundant representation. This moderate degree of redundancy is a small price to pay for shift invariance, especially since the alternative, wavelet frames representation, has much higher redundancy. In addition, it is impossible to obtain perfect reconstruction for the complex wavelets. The phase and symmetry relationships between the real and imaginary parts of the filters prohibit them from also satisfying the perfect reconstruction conditions. The complication really arises from the fact that complex filters do not have symmetric responses for positive and negative frequencies. For many image processing applications, the perfect reconstruction property is mandatory; image compression is a classic example. The complex wavelet transform is therefore ill-suited to such tasks. Note that the popular Gabor filter decomposition also does not have the perfect reconstruction property.

Kingsbury [36] discovered that a complex wavelet transform can be computed using *two real DWT trees* in parallel, with separate trees being responsible for the real and imaginary parts of the coefficients. This transform is hence known as a Dual-Tree Complex Wavelet Transform (DT-CxWT). The trick to make this decomposition provide shift invariance is to ensure that the samples in the two trees, for the 1-D case, are always half-sample offset (at the output subband's sampling rate) from each other. This is achieved if the first level tree filters are shifted copies of each other and higher level filters are designed to have opposing symmetries. In addition, the sidelobes of these filters will need to be strictly controlled [36] to minimise cross-band interference, which lead to shift variance. Ideally, the low-pass magnitude response in the two trees should match to yield good shift-invariance, but this can only be approximately achieved in reality. Since the individual trees are plain ordinary FWTs, they are invertible, and therefore the DT-CxWT, as a whole, is perfectly reconstructing. The 1-D DT-CxWT is illustrated in

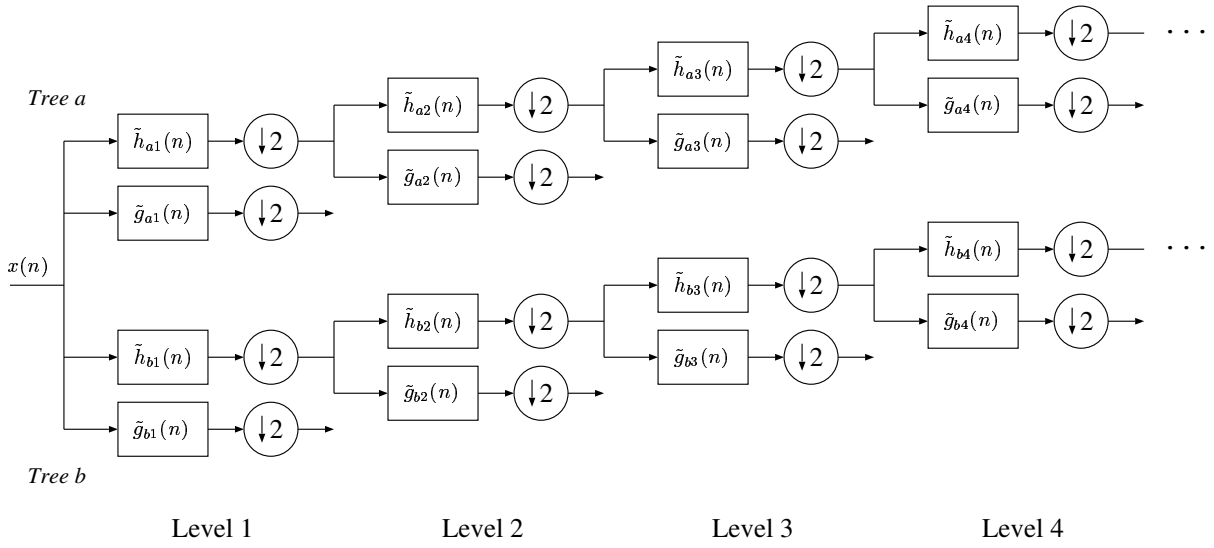


Figure 2.15. The 1-dimensional DT-CxWT, clearly showing the use of two ordinary DWT trees to compute the real and imaginary parts of the CWT

figure 2.15. Higher dimensional wavelets are obtained by taking tensor products of the basic 1-D filters as usual. The 1-D DT-CxWT algorithm is applied to each dimension separately. This structure naturally leads to a $2^m : 1$ redundant transform for an m-dimensional signal. For 2-D, an additional sum and difference operator is necessary to obtain the proper complex subbands from the four filter trees, aa, ab, ba, bb. This is a direct consequence of the rules of complex multiplication. The 2D DT-CxWT algorithm is shown in figure 2.17.

In light of the above discussion, it is apparent that two different filter pairs (low- and high-pass) are needed to construct a DT-CxWT, as opposed to the single filter pair used in a conventional FWT. It is normal practice to use linear phase filters for image processing applications, where phase distortion can drastically alter the contents of an image. Kingsbury has designed a pair of biorthogonal FIR filters that satisfies all of the above requirements; this pair is used in all subsequent experiments. To our knowledge, there are no other readily available filter pairs or families (cf. Daubechies family for FWT) in the literature to date. In designing filters suitable for use in a DT-CxWT, they must satisfy the conditions mentioned above. Kingsbury has detailed the necessary conditions in his publications, so they are not repeated here.

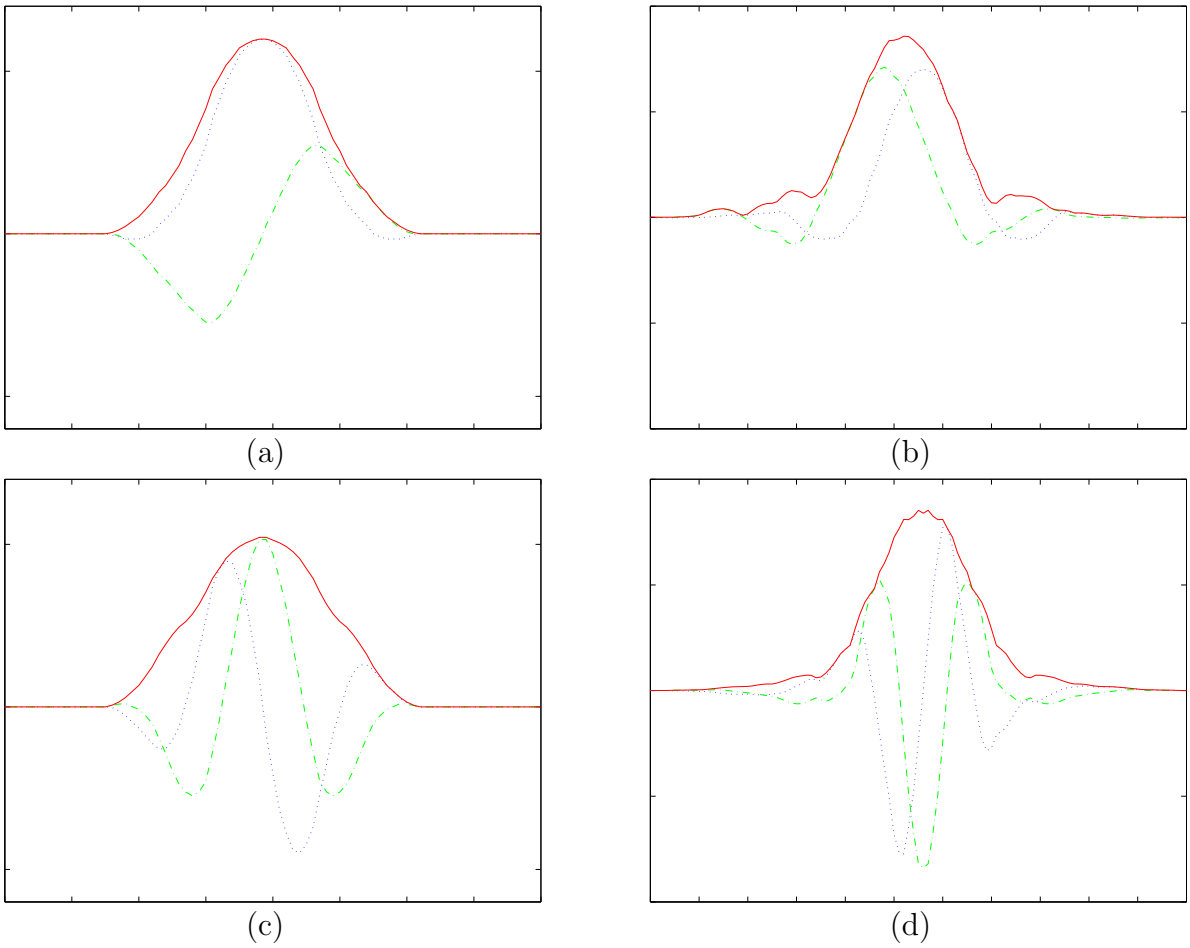


Figure 2.16. Level 4 impulse response for a complex wavelet. (a) scaling function $\phi(x)$ from a complex wavelet; (b) scaling function from a DT-CxWT; (c) wavelet function $\psi(x)$ from a complex wavelet; (b) wavelet function from a DT-CxWT. The DT-CxWT gives a reasonable approximation to the actual complex transform, albeit with longer filter lengths.

Since the DT-CxWT filters have different delays designed into them, the sum and difference operations on the DT-CxWT components yield strongly-directed transform subbands. The separable 2-D DT-CxWT filters are capable of selectively passing signals in pairs of quadrants in the 2-D frequency space due to the phase differences in the constituent subbands. This property is fundamental to achieving directional responses. While conventional FWT subbands are aligned with three directions (0°, 45° and 90°), a 2-D DT-CxWT has its subbands aligned to six different directions, at ±15°, ±45° and ±75°. Figure 2.19 illustrates the strong directionality of the DT-CxWT. The transform is

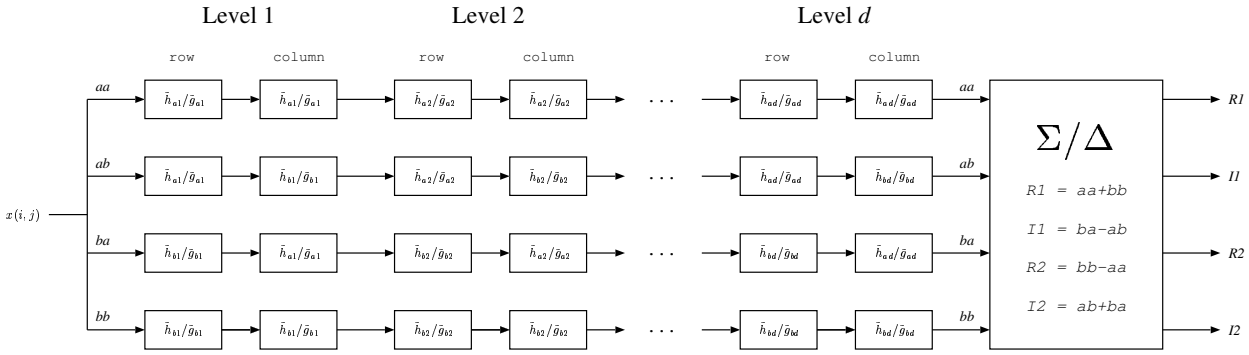
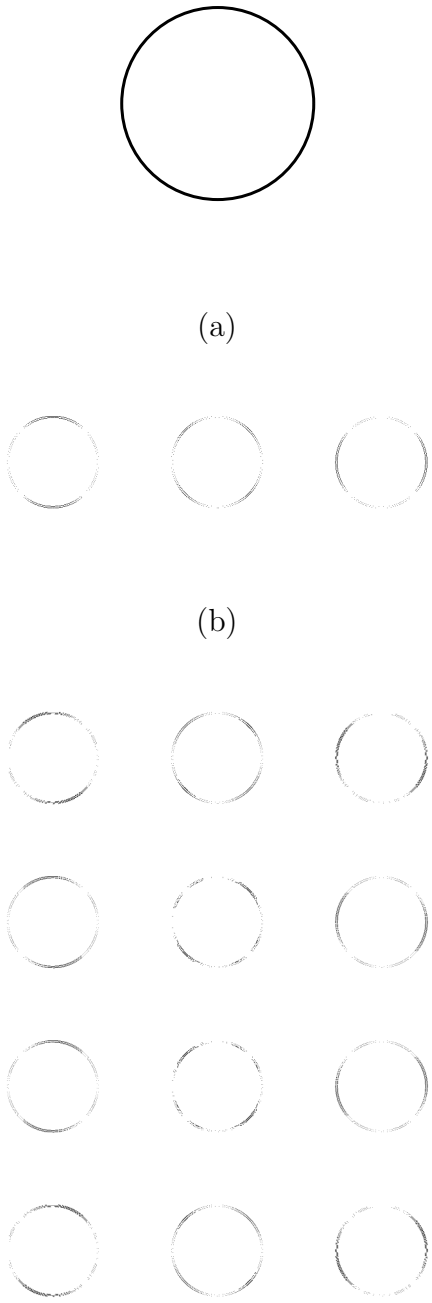


Figure 2.17. The 2-dimensional DT-CxWT. The rectangular boxes indicate the one level 1-D DWTs applied along rows and columns of the image. The sum-difference operator is applied at the end to yield the final DT-CxWT decompositions.

performed with symmetrical extension techniques (Section 2.2) at the image boundaries to limit artifacts at the edges. While the periodic extension technique has been widely used for their simplicity, the artificial discontinuities introduced in the transform coefficients are undesirable for texture segmentation.

A recent advance in DT-CxWT has been a new design in the filters used in the algorithm. In the original DT-CxWT, pairs of odd-length and even-length filters are used alternately in the transform trees. While this provided the necessary phase and symmetry conditions for shift invariance, there are a few undesirable consequences. Firstly, the effective sampling points in the two trees are placed asymmetrically. This implies that the higher-level coefficients do not interpolate the coefficients in the level immediately below. Whatsmore, with different filters in the transform tree, there must exist some mismatch in the frequency responses of the two trees, which affects the degree of the transform's shift invariance. Great care was taken to minimise the difference between the responses of the Kingsbury pairs in their design. These difficulties are addressed with the introduction of the *Q-shift* dual tree [37]. The new architecture uses the same sample-shifted odd-length pair for the first level transform, but all subsequent levels use even-length filters. The higher level transform filters in the Q-shift DT-CxWT are obtained from a single prototype, designed with a group delay of $q = \frac{1}{4}$. The required half-sample offset between trees can then be obtained by simply using the time-reversed version of the same filter. With a group delay of $\frac{1}{4}$, linear phase filters are no longer possible. This raises the possibility of using orthonormal filter sets (cf. Daubechies wavelets), further



(a)

(b)

(c)

Figure 2.18. Comparison between the directionality of the DWT and DT-CxWT. Level 1 high-pass subbands are shown here. (a) a circle is used as the input image; (b) the DWT subbands clearly exhibits 3 preferred directions, while (c) the 12 DT-CxWT subbands has 6 preferred directions (for both real and imaginary parts).

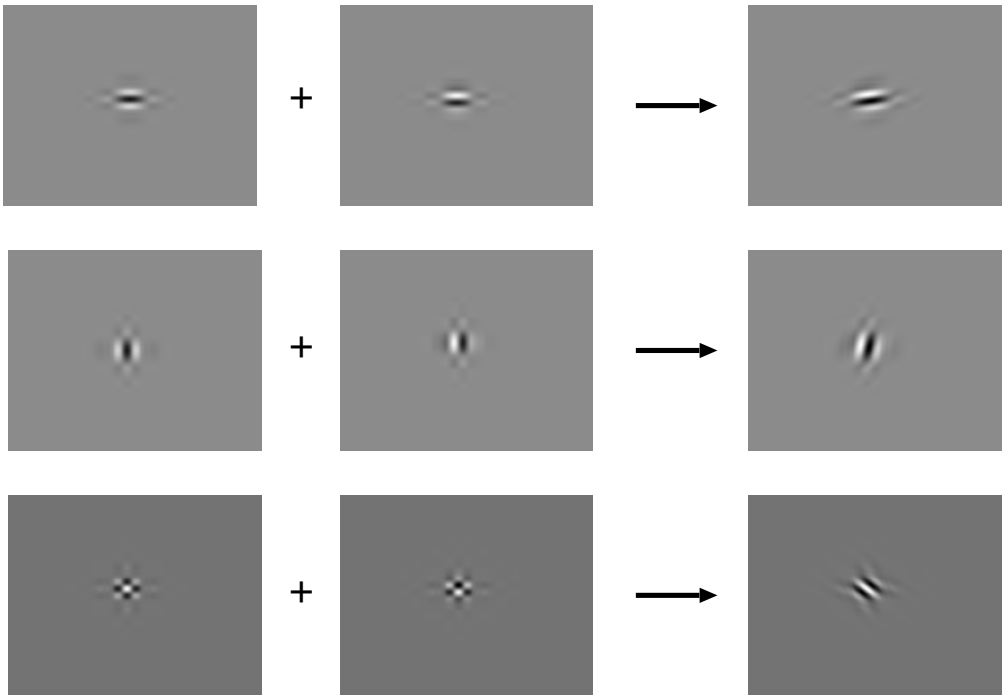


Figure 2.19. The effect of the sum-difference operator on the directionality of the 2-D DT-CxWT. The impulse responses for the *LH* and *HL* subbands in the aa and bb components are shown on top; their difference, $R1$ is on the bottom. The effect of different filter delays is clearly shown. The two separable impulse responses are combined to give a response with strong orientation in the -75° direction.

simplifying the transform by producing the reconstruction filters from the same prototype. In fact, the orthonormal nature of this filter set means that one only has to swap trees a and b for the inverse transform, which leads to implementation efficiency. The Q-shift dual tree architecture is much more elegant, as it leads to matching frequency responses and symmetric sampling grids across the two trees. It is appealing in its simplicity as well: one only needs to design a single filter instead of matching biorthogonal pairs. The restrictions on the possible filters are less complicated, also as a result of having only one filter.

2.7 Summary

The wavelet transform is a relatively recent mathematical development that has quickly found a place in many engineering applications. The mathematical basis of wavelet theory was presented in this chapter, as well as several implementation issues. Many different types of wavelets have been applied to texture analysis problems, the most popular being the Gabor wavelets, due to their superior time-frequency trade-off in theory. This thesis abandons Gabor wavelets in favour of critically or partially subsampled transforms due to computational concerns. In particular, the DT-CxWT discussed in 2.6 is heavily favoured for their unique mix of efficient computational characteristics and vastly improved directional properties over the critically subsampled DWT. The remainder of the thesis will focus on good texture segmentation performance with features derived from both of these transforms.

Chapter 3

Texture Features

The wavelet transform is the main analysis tool investigated in this thesis. While the desirable characteristics of the wavelet transform have been discussed in the previous chapter, the raw wavelet coefficients are not useful by themselves. A good feature extraction process is crucial to the overall segmentation performance of the system. This chapter discusses the feature extraction methodology used in this work.

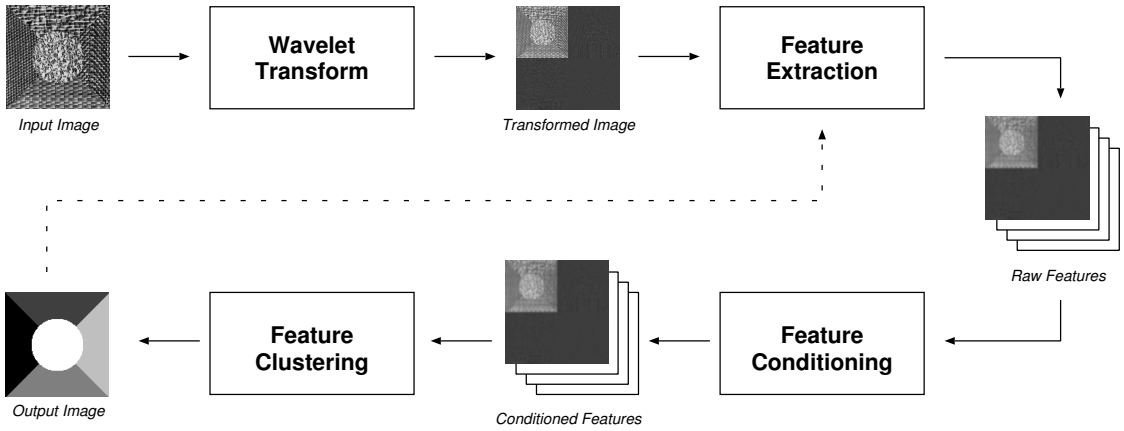


Figure 3.1. Wavelet-based texture segmentation system architecture

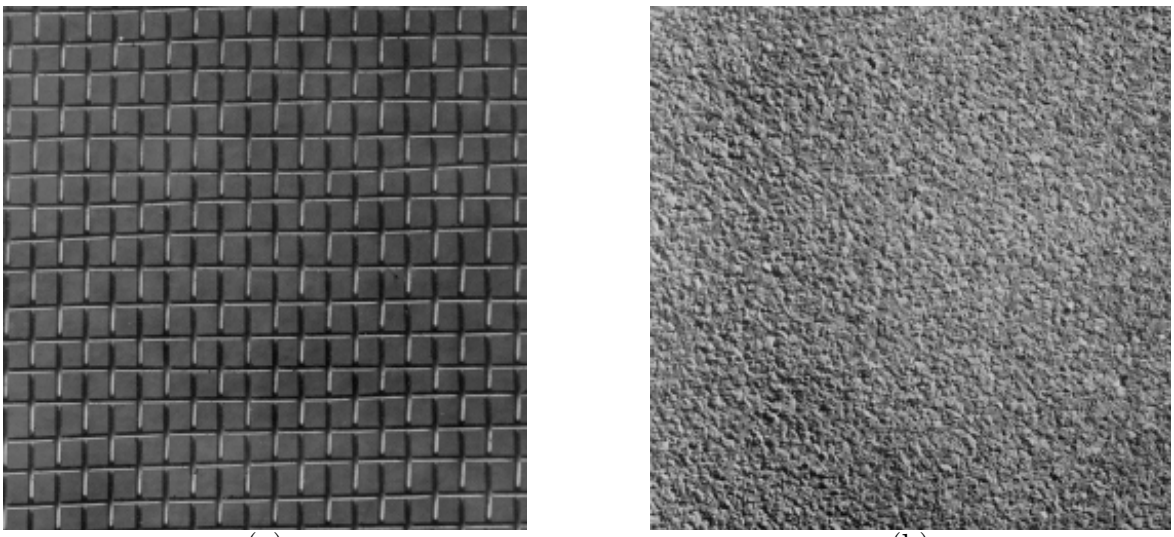
Figure 3.1 illustrates the architecture of the wavelet-based texture segmentation system described in this thesis. An image wavelet transform is the first stage of the process, which accepts textured images as input and produces the transform coefficients as the output. The *feature extractor* operates on these coefficients and produces a set of *feature vectors*. The array of feature vectors can be visualised as a series of feature images, as

shown in figure 3.1. Generally, the raw feature image needs to be processed further in order for the extracted features to be useful for segmentation; this is handled by the *feature conditioning* stage. Finally, the conditioned feature image is used by the *clustering* algorithm to perform the final segmentation, which produces the segmented image as output. The simplicity and modularity of this system means that the individual stages are swappable, meaning that alternative modules can be slotted into the system without disrupting the functionality of the other stages. For instance, the transform stage can easily accept a number of different transforms without altering the operation of other stages. This modularity allows us to concentrate on various aspects of the segmentation algorithm, and to study their effects on the overall performance. While it is conceivable that greater interaction between stages and possibly feedback (the dotted line in figure 3.1) may yield better segmentation performance [55], they also complicate the process. The notion of using feedback also mandates the introduction of some form of intervention into the algorithm, which may not be possible in real-world applications. This thesis examines the effect of different feature extraction methods on the overall segmentation performance without feedback or supervision. In a real-world system, the algorithms investigated here can be used as a front end, producing preliminary results before further enhancements or post-processing are performed with other, potentially more sophisticated, techniques.

3.1 Historical Texture Features

Traditionally, there have been two approaches to texture analysis: statistical and structural. Statistical approaches characterise textures by their statistical properties. They focus on uncovering the local properties of textures from the spatial statistics of the gray tone levels. Statistical approaches have found success with textures where *microtextures* are dominant. Microtextures refer to textures which are localised and contain rapid variations in image intensity (gray levels), as opposed to *macrotextures* which refer to textures with large patterns or have great regularity in their composition. Statistical approaches are fundamental in the sense that they operate directly on the raw gray tone or colour component values in the digitised images. As an alternative paradigm, structural approaches attempt to characterise textures by the mathematical rules underlying their appearance. Their focus is on finding the geometric rules which determine the arrangement of basic

texture primitives present in textures. The success of these approaches depend on a reliable mechanism for identifying the appropriate textural primitives, whose descriptions are usually in terms of statistical parameters. Therefore, structural approaches should more correctly be called statistical-structural approaches. These have found success mainly with textures where macrotextures are prevalent. Cases of both approaches are discussed further in the subsequent sections.



(a) (b)

Figure 3.2. Two examples of natural textures, with vastly different appearances. Sample (a) has a simple primitive and very regular placements, while (b) resembles output of a stochastic process with almost no regularity.

3.1.1 Statistical Texture Analysis

In his landmark paper, Haralick [27] isolated eight distinct historical statistical approaches to the measurement and characterisation of textures. These are: autocorrelation functions, optical transforms, digital transforms, textural edgeness, structural elements, spatial gray tone co-occurrence matrices, gray tone run lengths, and autoregressive models. A brief discussion of each of these methods serves as a good starting point in the description of statistical texture approaches. Other authors have written several evaluations [67, 15, 51] comparing statistical texture approaches among each other, as well as other approaches, for various applications.

The autocorrelation function method attempts to measure the relative sizes of the textural primitives by monitoring the decay of the autocorrelation function, defined as

$$C(m,n) = \frac{L_x L_y}{(L_x - m)(L_y - n)} \frac{\sum_{i=1}^{L_x - m} \sum_{j=1}^{L_y - n} I(i,j) I(i+m, j+n)}{\sum_{i=1}^{L_x} \sum_{j=1}^{L_y} I^2(i,j)} \tag{3.1}$$

where (m,n) is the displacement of interest for the autocorrelation function and $I(i,j)$ is the image intensity at pixel (i,j), and the image has dimensions $L_x \times L_y$. Textures with large texture primitives, or *textons*, have autocorrelation functions that decay slower (with increasing m,n) than those with smaller textons. In addition, highly regular textures (e.g. a brick wall) with strong periodicity is reflected in a corresponding periodicity in the autocorrelation function. In practice, the autocorrelation is computed for a limited number of displacements, and these values are taken to be the features for describing textures.

Optical transform methods use the diffraction patterns obtained from illuminating texture photographs as the textural features for identifying textured images. This method has found success in a surveillance application where classification of land use type based on textures is required.

Digital transform techniques use various transforms, such as Fourier and Walsh transforms, on linearised neighbourhoods in the texture. To use these techniques, an image is divided into smaller, non-overlapping blocks. These blocks are then unrolled into one dimensional vectors, and then these are subjected to the various digital transforms. Such techniques are common in surveillance as well, where the huge images must be divided into smaller blocks for efficient processing. The blocks may not be linearised. For example, the 2-D power spectrum, defined as the squared modulus of the Fourier transform of the image, operates on square blocks. The power spectrum is expressed in polar coordinates, which measures the energy in the image as functions of radius and angle. Effective features can be extracted from these functions: directional textures have predominantly angular dependence, while blob-like textures are predominantly radial dependent. These measures have been useful in discriminating between these different types of textures. Conners and Harlow investigated sample power spectral signatures derived from similar considerations in their study of different texture features [15].

Textural edgeness is an approach that attempts to characterise textures in terms of edgeness per unit area. Any edge-detection operator is possible for this approach. An example of textural edgeness description is

$$g(d) = \sum_{(i,j)} (|I(i,j) - I(i+d,j)| + |I(i,j) - I(i-d,j)| + |I(i,j) - I(i,j+d)| + |I(i,j) - I(i,j-d)|) \tag{3.2}$$

The individual measures for every pixel $I(i,j)$ is the average approximate gradient in the horizontal and vertical directions over a distance of $2d$ around the pixel. The function is then the overall average gradients over the entire image, and it appears similar to the autocorrelation function (in combined horizontal and vertical directions). Texture features can be extracted by choosing a number of different distances, d. Textures with a high density of edges (such as rugged surfaces) would have much larger gradients than those with a low edge density. The plot of $g(d)$ also illustrates the dominant size of textons present in the image.

The approach of structural elements applies methods in mathematical morphology to the assumed basic structures in image textures. In essence, this approach attempts to pattern match textures with a set of typical geometric constructs. Simple structural elements are defined, and these are used to "erode" the original image to form a series of new images. Features are then extracted from these images. This operation attempts to find the locations in an image which match some basic texture elements. These elements are typically constructed from lines and squares.

Spatial gray tone co-occurrence, or equivalently, the spatial gray level dependence method [15], measures the spatial distribution and local dependence among the gray tones present in the texture. The co-occurrence matrix $P_{\theta,l}(m,n)$ of an image $I(i,j)$ counts the number of times a pair of pixels of I, separated by a distance l at an angle θ, have intensities m and n. This matrix is clearly symmetric, and is very easy to compute. The size of the co-occurrence matrix depends on the granularity of the quantisation of

the pixel values, i.e. the number of gray levels in the image intensities.

$$I(x,y) = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 2 & 2 & 1 \\ 0 & 1 & 2 & 3 \\ 1 & 2 & 3 & 3 \end{bmatrix}$$

$$P_{(0^\circ,1)} = \begin{bmatrix} 2 & 2 & 0 & 0 \\ 2 & 2 & 4 & 0 \\ 0 & 4 & 2 & 2 \\ 0 & 0 & 2 & 2 \end{bmatrix} \qquad P_{(90^\circ,1)} = \begin{bmatrix} 0 & 2 & 1 & 0 \\ 2 & 2 & 3 & 1 \\ 1 & 3 & 2 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

$$P_{(45^\circ,1)} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 2 & 4 & 0 \\ 1 & 4 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} \qquad P_{(135^\circ,1)} = \begin{bmatrix} 0 & 0 & 3 & 0 \\ 0 & 4 & 0 & 1 \\ 3 & 0 & 2 & 2 \\ 0 & 1 & 2 & 0 \end{bmatrix} \tag{3.3}$$

The idea is to construct a number of co-occurrence matrices [8] for different spatial directions, and extract a set of statistical measures from them for analysis. Haralick has found a number of statistics generated from co-occurrence matrices that are useful as texture features; examples of these are:

$$\begin{aligned} \text{Energy} &= \sum_{i,j} P_{\theta,l}^2(i,j) \\ \text{Entropy} &= \sum_{i,j} P_{\theta,l}(i,j) \log P_{\theta,l}(i,j) \\ \text{Contrast} &= \sum_{i,j} |i-j|^m (P_{\theta,l}(i,j))^n \quad m,n>0 \\ \text{Local Homogeneity} &= \sum_{i,j} \frac{1}{1+(i-j)^2} P_{\theta,l}(i,j) \\ \text{Max probability} &= \max P_{\theta,l}(i,j) \end{aligned} \tag{3.4}$$

Co-occurrence matrix methods have been investigated by other researchers with moderate success. Chen and Pavlidis [11] used co-occurrence matrices in a hierarchical scheme to segment textured images. Conners and Harlow [15] applied co-occurrence features in discriminating visually distinct textures. The co-occurrence features are similar, in idea, to the autocorrelation function features. However, they contain significantly more

information on the spatial dependencies (two more variables) present in the image. A disadvantage of co-occurrence features is that they contain no information on the structure or shape of textons. As a consequence, co-occurrence features have mostly found success with microtextures.

The gray tone run length method characterises textures from the length, gray tone value and direction of gray tone runs in the images. Secondary statistical measures are extracted from these quantities for better results. Let $R(i, j, \theta)$ be the number of occasions where there is a run of length i of gray level j along the direction θ; some useful statistics are

$$\begin{aligned}
\text{Short run emphasis} &= \frac{1}{K}\sum_{i,j}\frac{R(i,j,\theta)}{j^2} \\
\text{Long run emphasis} &= \frac{1}{K}\sum_{i,j} j^2 R(i,j,\theta) \\
\text{Gray level uniformity} &= \frac{1}{K}\sum_{i}(\sum_{j} R(i,j,\theta))^2 \\
\text{Run length uniformity} &= \frac{1}{K}\sum_{j}(\sum_{i} R(i,j,\theta))^2 \\
\text{Run percentage} &= \frac{K}{L_x L_y}
\end{aligned} \tag{3.5}$$

where K is the total number of runs and L_x, L_y are the dimensions of the image. Any number of these statistics (for any direction) can be used as texture features [15].

The auto-regression approach attempts to construct appropriate statistical models to describe textures. This approach relies on the principle that the intensity of a pixel depends on its surrounding neighbours; texture features are taken to be the parameters of this dependency. Many popular statistical models for the dependence relationship have been used in this approach. In the 1980s, researchers developed methods based on Markov Random Fields (MRF) [51, 22, 16]. In this approach, the pixel gray level values in a texture are treated as the observations taken from a Markovian process. The random field model is characterised by the conditional distribution $P(I(i,j)|\text{neighbours of pixel}(i,j))$ relating the gray level of an image pixel at (i, j) given the values of its neighbouring pixels. Gaussian and binomial distributions are two examples that have been investigated in the literature.

The gray level difference method is described in [15]. It is similar in principle to the co-occurrence matrix discussed above. This method considers the difference in gray level values of two pixels separated by a given displacement vector. The probability density function of these differences is estimated by computing these gray level differences over the entire image. Texture features are extracted by calculating statistics from the estimated probability densities. These statistics are very similar to those described in equations (3.4) and (3.5), with the probability density function in place of the co-occurrence matrix and run length function, respectively.

A fairly recent class of texture features is based on fractals [51, 10]. These methods attempt to characterise textures by the fractal geometry in images. In natural textures, there are many rough and irregular objects, and these are very difficult to model or describe using geometric methods. Fractals are much better suited to describing these objects, and they also have the advantage of being insensitive to scale (i.e. multi-scale representation). The representation is also very compact, and this is appealing for pattern recognition tasks. The key computation involved in extracting fractal features is the determination of the *fractal dimension*. A simple method to estimate this parameter is the Differential Box Counting method. In this method, an $M \times M$ image is modelled as a 3-D mesh, with the z coordinate being the gray level, normalised into M discrete bins. The image is then divided into $s \times s$ equal-sized boxes, where $r = \frac{s}{M} \leq \frac{1}{2}$. Let the maximum and minimum values in the box at (i, j) be $z_{max}(i, j)$ and $z_{min}(i, j)$, respectively. Their differences for all boxes are summed together:

$$N(r) = \sum_{i,j=1}^{s} z_{max}(i, j) - z_{min}(i, j) + 1 \tag{3.6}$$

This process is repeated for a number of different r, and the fractal dimension, D, is determined from the resulting log-log plot of r versus $N(r)$ using the relation

$$D = -\frac{\log N(r)}{\log r} \tag{3.7}$$

The fractal dimension can be used as the texture feature, which is an extremely compact representation (dimension of 1). For segmentation experiments, the fractal dimension of small sub-windows around each pixel is calculated as the texture feature for that particular pixel. However, the fractal dimension is not a unique description for textures, so several different fractal dimensions should be calculated for robust segmentation.

3.1.2 Structural Texture Analysis

Statistical-structural approaches are based on the view that textures are composed of regular or semi-regular placements of textural primitives. These primitives are connected sets of cells with a common set of characteristics. The cells can simply be a region of pixels in the image, or they can be of a higher-level, containing a set of edges and polygons. The spatial interaction can be described by formal grammars. However, the placement rules can become rather complicated for real textures. The lack of flexibility in an inherently restrictive grammatical description presents a practical difficulty of such schemes. More commonly, a hybrid structural-statistical approach uses explicitly defined primitives (the structural part) with probabilistic description of placement rules (the statistical part). These two different parts of structural texture analysis will be discussed separately in this section.

Fundamentally, a primitive is defined as a set of connected pixels sharing some common list of attributes. These attributes may be simple quantities such as gray level value or edge direction. Higher-level attributes, such as the geometric shape of set, can also be used. Neighbourhood operators are often used on sets of pixels, and homogeneous regions of their outputs are grouped together to form the primitive. Depending on the variety of attributes sought by these operators, it is then possible to identify one or more *types* of different primitives present in a texture. Afterwards, the description of the image has been transformed from a simple matrix of gray level values into a more informative description in terms of primitives spatially scattered throughout the image.

The next step in a structural approach is the description of the spatial interaction between the identified primitives. The description can include information such as: the directional likelihood of two different primitives being adjacent to each other, the repeat probability of a primitive, the adjacency of two primitives, and the closest distance between any pair of primitives. Obviously, there are endless possible quantities that can aid the overall description of a texture. Depending on the application, the number and type of rules necessary for an adequate description change greatly. Historically, it has been useful to characterise textures either as weak or strong, depending on the level of spatial interaction between primitives. This distinction assists in attaining a description of the spatial interaction, as different techniques are needed for strong and weak textures.

For weak textures, the limited spatial interaction between primitives make precise grammatical descriptions infeasible. Instead, features are extracted directly from the primitives to describe the texture. In a sense, this is similar to purely statistical approach. The feature extraction methods include: histogram signatures, edgeness per unit area, run-length features and local extremum density. Many of these methods have been discussed in the statistical approach section. The only difference here being that these methods operate on attributes of the identified primitives. Indeed, the use of primitives instead of raw pixels is an additional layer of abstraction. For a single pixel, the most obvious, and perhaps only, attribute is its gray level value (or colour component values for colour textures), but for a primitive, there are many possible attributes that one can use to extract texture features from. As may be expected, primitives for weak textures are typically very small structures, sometime consisting only of a small number of pixels. Obviously, as the sizes of primitives decrease, these methods approach the statistical approaches outlined in section 3.1.1. As a result, there is considerable overlap between structural approaches for weak textures and statistical approaches.

For strong textures, the notion of abstraction in primitives is much clearer. High-level description in terms of placement of primitives is possible for strong textures. A method to achieve this is through the calculation of co-occurrence matrices from the primitives. Again, this is a generalisation of the co-occurrence matrix method described in section 3.1.1, and indeed becomes identical when the primitives are reduced to single pixels. However, the generalisation to larger primitives yield a rich variety of attributes that can be used to calculate the co-occurrence matrices. For example, angular orientation can replace the gray level value in the calculations, which would be far more effective for highly angular dependent textures. To be precise, consider an image divided into a set of primitives, $\mathcal{P}$. Each primitive can be considered to be a vector of attributes, $T(p) = [t_p(1), t_p(2), \cdots, t_p(k)], p \in \mathcal{P}$. Define a set of spatial relationship descriptors, $\mathcal{S}$. The spatial relationship between a pair of primitives (p_1, p_2) is described by a vector in $S(p_1, p_2) \in \mathcal{S}$. As an example, the individual components of such a vector may measure distance and angular separation between the primitives. The generalised co-occurrence matrix simply counts the number of times a pair of primitives (p_1, p_2), separated by some specified spatial description, have attribute vectors equal to T_1 and T_2, respectively. The generality of this description is quite powerful, but in practice, neither the attribute vectors

nor the separation descriptions are too intricate. Typically, they resemble co-occurrence matrices in section 3.1.1, except that the attributes may not be simple gray level values.

3.2 Filter-based Texture Features

The vast majority of early algorithms in texture analysis follow either the statistical or the multichannel approaches. However, neither approach is satisfactory for a general analysis of textures, where the relative importance of micro- and macrotextures vary. What is needed is a method to effectively isolate the defining qualities for a particular texture, whether it be a local property or a global arrangement. More recent techniques involve the use of a *multi-scale* analysis of the textured image. The main advantage of multi-scale approaches is that they are capable of zooming to the appropriate scales for different textures. Even within one texture, there may be important, defining characteristics at more than one scale, a difficulty not solved by the classical statistical and structural approaches. Fundamentally, multi-scale approaches rely on the use of multiple spatial filters with different frequency characteristics to perform the analysis. Hence, these approaches are often called filter-based or multichannel filtering approaches. Early incarnations of these methods typically used Gabor filters at different scales to characterise textures. More recently, these approaches have evolved to encompass the use of other decomposition techniques, such as wavelets. The virtues of wavelets have already been discussed in chapter 2. Despite the emphasis on the multi-scaled perspective, a significant part of multichannel approaches is statistical in nature. Therefore, there are many similarities between algorithms following the classical statistical and modern multi-scale approaches. The most obvious difference is that the former operates on the image intensity values directly, while the latter operates on the filtered image, or transform coefficients. Referring to the framework outlined in figure 3.1, the differences between statistical and multichannel approaches only arise in the existence of the first block (Wavelet Transform). The remainder of this section provides a brief summary of the existing filter-based approaches in the literature.

3.2.1 Paradigm of Filter-based Approaches

All image decompositions produce a series of image components from a single input image. One way to distinguish between different image transforms is by the *redundancy* introduced into the decomposition. Non-redundant transforms produce the same amount of data in the output as the input, while redundant transforms produce more data, even if the information content is not increased. For image analysis tasks, non-redundant transforms are often desirable from an algorithmic standpoint, due to computational efficiency achievable with more compact data sets. However, for texture analysis, redundancy is often required to achieve good characterisation, since textures are complex phenomena that may not be adequately described by the compact, non-redundant representations. An example of a redundant decomposition is the Gabor transform that is used in many cognitive image processing applications. Generally speaking, a transform decomposition attempts to isolate different types of information about the original image in its components. Orthogonal decompositions, for example, decorrelates the original information, with respect to some defined inner product, into independent channels. It is reasoned that feature vectors extracted from the decomposition coefficients would yield *signatures* which are capable of describing textures present in the image. A joint time-frequency decomposition yields components which contain image information at different resolutions and within different frequency bands. For example, a popular Gabor filter configuration uses filters at 4 scales and 6 orientations to produce 24 filtered images [62]. The design of the decomposition yields information in 4 separate dyadic scales, and information along 6 evenly-spaced directions with the same angular resolution.

Once an image is decomposed, features are extracted from the component images. A *feature element* is simply a statistic computed from the transform coefficients in the decomposed image. This statistic is often a simple rectified coefficient value, but it can generalise to any statistic. Examples include the phase of the coefficients and probability distribution parameters used to model the transform coefficients. The individual feature elements are collated to form a *feature vector*. The usual approach is to concatenate the feature elements together, producing a vector of values. While Gabor transforms, in various configurations, have been most commonly used in the literature, the recent trend has been to use more modern time-frequency decompositions like wavelet transforms. The focus of this thesis aligns with this trend; in particular, discrete and complex wavelet

transforms (see chapter 2) are the two techniques that will be examined in detail. Before explaining the details of wavelet-transform based texture features, a survey of existing filter-based techniques is presented first.

3.2.2 Survey of Existing Filter-based Approaches

The first filter-based approach to texture analysis is attributed to Laws [43]. His approach consists of convolving images with a set of "micro-texture" masks, which have been empirically designed to match primitive texture patterns. These masks are obtained by computing tensor products combined from several basic 1-D vectors. First, 3 basic vectors of length 3 are defined:

$$\begin{aligned} L_3 &= [\,1 \quad 2 \quad 1\,] \\ E_3 &= [\,-1 \quad 0 \quad 1\,] \\ S_3 &= [\,-1 \quad 2 \quad -1\,] \end{aligned} \tag{3.8}$$

These are designed to measure the average gray level, edge and local peak (spot), respectively. 3×3 masks can be computed by forming outer products of any pair of vectors, giving 9 different masks in total. By convolving the basic vectors with each other, a set of 5 distinct vectors of length 5 ($L_3 * S_3 = -E_3 * E_3$) are produced:

$$\begin{aligned} L_5 = L_3 * L_3 &= [\,1 \quad 4 \quad 6 \quad 4 \quad 1\,] \\ E_5 = L_3 * E_3 &= [\,-1 \quad -2 \quad 0 \quad 2 \quad 1\,] \\ S_5 = L_3 * S_3 &= [\,-1 \quad 0 \quad 2 \quad 0 \quad -1\,] \\ W_5 = E_3 * S_3 &= [\,1 \quad -2 \quad 0 \quad 2 \quad -1\,] \\ R_5 = S_3 * S_3 &= [\,1 \quad -4 \quad 6 \quad -4 \quad 1\,] \end{aligned} \tag{3.9}$$

By taking the outer products of these vectors, 25 5×5 masks can be generated. For example:

$$L_5L_5 = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \qquad E_5S_5 = \begin{bmatrix} 1 & 0 & -2 & 0 & 1 \\ 2 & 0 & -4 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 4 & 0 & -1 \\ -1 & 0 & 2 & 0 & -1 \end{bmatrix} \tag{3.10}$$

Macro-statistical features are computed from the coefficients of the filtered images. Various different statistics, such as mean, energy, variance, kurtosis and skewness, have all been proposed as possible feature measures. These statistics were used by Sharma and Singh [67] in classifying 944 textures from 4 categories. Hsiao and Sawchuk [30, 29] used Laws' masks and calculated the mean deviation of the convolved images as features for texture segmentation.

Rather surprisingly, the "filter-and-compute statistic" paradigm in Laws' method has more or less stayed intact in almost all filter-based texture analysis algorithms since proposed. Obviously though, the transforms (filters) and feature extraction processes have been greatly modified. For example, local linear transform techniques were studied by Unser and Murray [77, 78], who used them on both texture classification and segmentation problems. They used local orthogonal convolution masks as their filters, and then calculated the variance of the histogram (of the coefficients) as the texture feature. Multichannel filtering algorithms are intimately related to local linear transform methods, and can be regarded as a generalisation of such.

Among published works, the majority of multichannel filtering methods use 2-D Gabor filters (equations (2.62) and (2.63)) as the choice of transform. Jain and Farrokhnia [32] used Gabor filtering to extract features for segmenting textures. In their scheme, texture features are extracted from a subset of a total of 20 Gabor filtered images. The particular Gabor filter subset is chosen adaptively by sorting the energy values of all Gabor subbands, then choosing the minimal set that accounts for 95% of total intensity variations. The feature values are the rectified coefficients of the filtered subset. These features formed the basis of their unsupervised segmentation of texture mosaics. A similar feature extraction process is used in Jain and Karu [33], where a comparison was made between the use of 8 or 16 Gabor filtered images. These features are used by a neural network classifier to perform a range of classification tasks. Document segmentation and barcode localisation were used as application examples for the texture features. Randen and Husøy [62, 61, 63, 64] considered a range of different transforms in their study on texture analysis. In [63], they examined the merits of using optimally designed FIR filters. In this method, a FIR filter is designed to provide maximally separable filtered images from two different textures. The separability is defined by the Fisher criterion:

$$J_F = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{3.11}$$

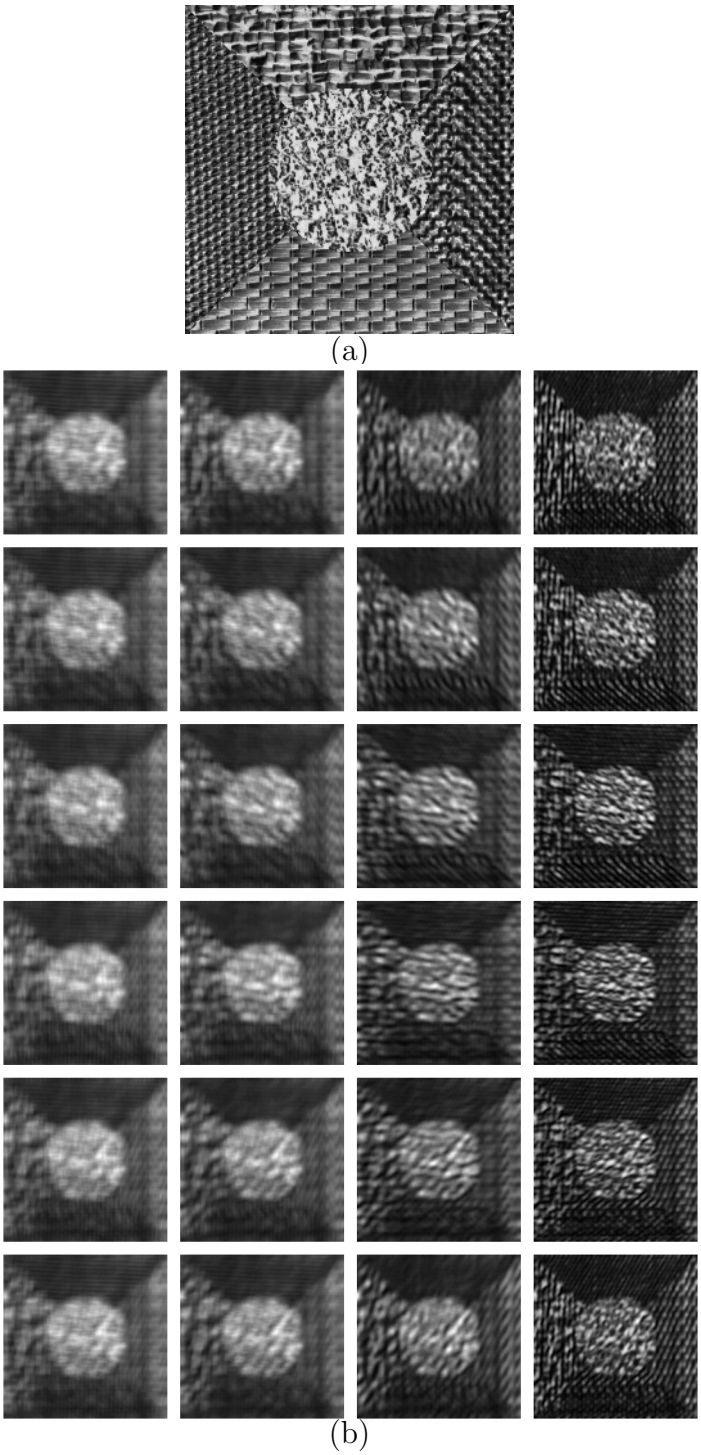


Figure 3.3. (a) Input texture mosaic; (b) the raw extracted features (coefficient magnitudes) from a Gabor decomposition, with 4 dyadic scales and 6 orientations, at $0°, 30°, 60°, 90°, 120°$ and $150°$. The 24 subbands are collated into images to illustrate the feature variations in different subbands.

where μ_1, μ_2 are the means of the filtered coefficients for the two different textures, and σ_1, σ_2 are the variances. The resultant FIR-filtered images can then be segmented using a simple two-class discriminant classifier. In [62, 61], the same authors compared the Gabor filters method with wavelet transforms, at fixed or multiple sampling rates, for texture feature extraction. Different decomposition structures were also examined. Throughout the comparison, they have used local energy function to extract features from coefficients. The resultant segmentation performance showed comparable results between Gabor and wavelet transforms. A comprehensive comparison of filter-based texture features for classification is found in [64]. Among the filters considered are: Laws' masks, ring/wedge filters, eigenfilters, wavelet transforms and various Gabor filters. There was no noticeable evidence to favour Gabor filters over wavelet transforms, although Laws masks and ring filters were inferior.

Teuner, Pichler and Hosticka [74, 55] applied dyadic Gabor filters in their feature extraction scheme. Filters are iteratively selected based on the *contrast* of the filtered images as follows. At the lowest resolution, a coarse decomposition is first computed using a Gabor filter bank with different centres and orientations. The transform image is smoothed using some local window. The contrast information is computed as:

$$C(u,v) = \frac{w(u,v) - \overline{w(u,v)}}{w(u,v) + \overline{w(u,v)}} \tag{3.12}$$

where $w(i,j)$ is the maximum magnitude of smoothed transform coefficient of Gabor filter with parameters u, v; $\overline{w(u,v)}$ is the average magnitude. Then the image is decomposed at a finer level with the next set of Gabor filters in the dyadic hierarchy, and contrast is again computed. The maximum contrast at this level (for all filters) is compared to those from the previous level. If all the contrasts are greater, then the dyadic decomposition will proceed to the next level, otherwise it will be terminated. Upon the eventual termination of the decomposition process, all the individual contrast values are sorted to determine the pecking order of the Gabor filters to be used for extracting features. The actual features are the coefficient energies, and the exact number of Gabor filters to be used is determined by a user-defined threshold for the contrast.

Ohanian and Dubes [51] used Gabor transform magnitudes as their texture feature when comparing with three other classes of texture features: co-occurrence, Markov Random Fields and fractals. These different methods were compared for texture classification. Hofmann, Puzicha and Buhmann [28] performed unsupervised segmentation on texture

mosaics using Gabor filter coefficients directly as features. In their scheme, twelve filters at four orientations and three octave-spaced scales are used in conjunction with the raw image to produce feature vectors. Reed and Wechsler [65] considered four joint space-frequency representations, and concluded that psuedo Wigner distribution (PWD) is suitable for texture segmentation. It can be shown that the Gabor representation is a special case of the Wigner distribution.

In [46], the means and variances of each Gabor-filtered image are selected as the features to characterise texture images. These features are used in a texture database searching application, where classification accuracy and speed of retrieval are both important objectives. A database is constructed with a bank of 24 Gabor filters at 4 scales and 6 orientations, and this database was able to achieve 74% search accuracy. In the interest of reducing the computational requirements, the authors also provided an adaptive filter selection method which dramatically reduces the number of Gabor filters required, and hence the retrieval times. However, this comes at the cost of a significantly reduced search accuracy. Their strategy computes the difference in spectral energies between the input image and the database average. Then, the four Gabor filters that account for most of the difference are selected to compute the texture features in the query.

In [23], Dunn *et al* used Gabor filters as the analytical tool for a model-based texture segmentation algorithm. In their approach, they considered textures to be either uniform or non-uniform. For uniform textures, the texture primitives are identical and separated regularly. Thus, it is possible to design tuned Gabor filters, with the centre frequency and orientation designed to match the size and orientation of the texture primitive. When the texture is filtered with the tuned Gabor filter, a large response will result; conversely, when the same filter is applied to a texture with a different primitive, a low response will be produced. This step signature draws the boundary between the different textures. In addition to textures with different primitives, the tuned Gabor filter can also distinguish between two identical textures that have been displaced from each other, i.e. a phase difference exists between them. A valley or ridge signature is produced from this kind of situation. For non-uniform textures, similar efforts to tune Gabor filters will not produce ideal results, but if two textures are sufficiently different, the Gabor filtered outputs will still exhibit significant output responses. The main difficulty with their approach is the selection of the appropriate parameters, since their entire methodology is based on tuning Gabor filters to respond to particular primitives. Also, for a multi-texture mosaic with

different constituent textures, multiple Gabor filters will be needed to respond to each individual texture.

The schemes described above utilise Gabor filters as the primary analytical tool for texture feature extraction. Later examples of filter-based methods exploited the progress made in the wavelet field, with different types of wavelet transforms used in place of Gabor filters.

Chang and Jay-Kuo [9] used tree-structured wavelet transforms, a variant of wavelet packet decomposition (section 2.4), to extract textural features. In their approach, each subband is repeatedly decomposed subject to a thresholding criterion: if the energy of a subband is significantly smaller than the highest energy subband at the same scale, it will not be subjected to further decomposition. This transform produces a non-redundant representation. They then use the *sorted* subband energy values as the texture features, which are then applied to a texture classification problem. It was found that using the first J dominant energy channels (J less than total number of possible subbands) gave the algorithm more robustness.

Unser [76] and Xie and Brady [84] developed redundant, shift-invariant wavelet frames (section 2.1.2) for feature extraction. Unser applied a discrete wavelet frame decomposition on textures and extracted channel variances as the features. He applied these features to both texture classification and segmentation problems. The channel variances are estimated with the use of a sliding window on each subband, and excellent classification results were obtained from this approach. For segmentation, a smoothed squared coefficient energy is used as the features instead of channel variance. This algorithm was able to effectively segment a simple texture mosaic. Xie and Brady applied wavelet frame decomposition to textures, and calculated local energy and phase as their texture features. They developed a method based on the Hilbert transform to decouple the energy and phase components for their segmentation experiments. This method was used to successfully segment a variety of texture images, from aerial photographs to Brodatz mosaics. Laine and Fan [41] used the *zero-crossings* from wavelet frame decompositions of textured images to extract their features. The zero-crossings are used to compute *envelopes* of the wavelet coefficients. This rectification method is an alternative to local extremum feature extraction. The values along the envelopes are used directly as the texture features; this has the benefit of being adaptive, and preserves boundaries very well. These features have

been used to produce accurate segmentations, but they tend to also produce residual regions, which must be cleaned up with post-processing filters. In another work, Laine and Fan [42] investigated the use of different wavelet transforms, including critically sampled and unsampled decompositions, in extracting features for texture classification. They concentrated on using wavelet energy and entropy signatures as the texture features.

Simoncelli and Portilla [68] characterised textures using over 1000 statistics computed from wavelet transform coefficients. They used what is called a "steerable pyramid" as the wavelet transform, which utilises filters designed to have arbitrary orientations, unlike the standard wavelet filter pairs. The feature set extracted was used to synthesise textures from the compact representation.

Porter and Canagarajah [56] examined the use of wavelet coefficients to determine the dominant characteristic of an image. A texture is considered as smooth if the low-resolution subbands (higher levels in the transform) have significantly more energy than the high-resolution subbands; otherwise, the image is considered textured. This distinction enabled them to use different features for the smooth or textured parts of an image with mixed types. Jasper *et al* [34] attempted to characterise textures by adaptively designing orthogonal wavelet filters that minimises the energy in the detailed subbands in the transform, i.e. maximal energy compactness in the low-pass subband. Their technique have been useful in identifying defects in denim, since the defective parts in the material would yield unusually strong responses in the high-pass subbands.

Kam and Fitzgerald [35] considered a general multiscale image segmentation algorithm (independent of features). They applied their algorithm to segmentation of textures, using tree-structured wavelet frame features similar to Chang and Kuo's method [9]. de Rivaz and Kingsbury [20] considered using Gabor and complex wavelet features for fast texture image retrieval. The characterisation of textures was done using mean and standard deviation of the coefficient magnitudes in each subband. Complex wavelet features was found to give comparable retrieval performance when compared with Gabor features, but at much reduced computation costs.

Van de Wouwer *et al* [80] used the discrete wavelet transform to extract texture features for classification experiments. In particular, they used a combination of wavelet energy, mean deviation, histogram signatures and co-occurrence statistics to effect characterisation. Only detailed wavelet coefficients are used in computing the statistics. The

wavelet coefficient histograms are modelled as a family of exponential functions; the signatures are parameters of the best-fitting member of that family. They also investigated the validity of the exponential model, and found it to give satisfactory fit to most naturally occurring textures in their database. When computing co-occurrence features, the coefficients are first quantised into discrete bins, and statistics such as those in equation 3.4 are used as features. Van de Wouwer *et al* [79] late considered the problem of colour texture classification. Colour information are typically stored in three channels, instead of the single channel in gray level images. Raw colour images are usually represented as a combination of separate red, green and blue (RGB) components, but a number of different colour spaces (e.g. YUV, YIQ) are useful for different applications. In this work, the authors compared features extracted from average wavelet energy intensity values (i.e. gray scale wavelet energies) with wavelet energy correlation signatures in four different colour spaces. The correlation signatures are simply the cross-correlation across different colour planes (in respective colour space). It was found that colour correlation signatures performed better than gray level wavelet energies in texture classification experiments. In particular, certain colour spaces yielded much better results than others.

Pan and Wang's paper on texture segmentation [52] compares separable and non-separable 2-D wavelet frames for extracting texture features. In addition, comparison is also made between pyramidal and tree-structured decomposition for both types of wavelets. They also proposed using extremum density measure as the texture feature, and provided a comparison with the more common energy and entropy features. This measure has the advantage of being a more direct measure to the coarseness of an image (a major attribute of textures), and tends to have better numerical properties than wavelet coefficient statistics.

3.3 A Novel Feature Extraction Method

Two particular types of wavelet transforms discussed in Chapter 2 are the discrete wavelet transform (DWT) and Dual-Tree Complex Wavelet Transform (DT-CxWT). Texture feature extraction from these different transforms are discussed separately in this section.

3.3.1 Features From Discrete Wavelet Transforms

A Discrete Wavelet Transform (DWT) performed on an image yields a non-redundant representation. The result of a DWT displays the familiar repeated sub-division of low-frequency subbands, due to the 4-times downsampling (for 2-D data) at each level of the transform (see figure 2.12). Orthogonal, and more generally, bi-orthogonal transforms completely decorrelate the image energy present in the subbands, while preserving the total image energy.

Consider an input image of size $M \times N$ pixels subjected to a d-level DWT. This decomposition produces $3d+1$ DWT subbands. The feature vector for each spatial location (each pixel in the lowest level subbands) is the concatenation of corresponding coefficients in each of the subbands, analogous to the approach for Gabor filters highlighted in section 3.2. In other words, these feature vectors have dimension $L = 3d + 1$. It is reasoned that, since each wavelet subband provides information at different scales and orientations, such a collection of wavelet coefficients would yield a multiresolution description of the textured image which can be used as a signature for textures. The multiple scale and orientation information is of particular interest to texture analysis, given the multiresolution nature of textures. The length of the feature vectors can be controlled by varying the wavelet transform depth, d. Although higher level decompositions produce subbands at more resolutions, very long feature vectors must be used with caution. In pattern recognition, it is very well-known that long feature vectors may reduce the efficiency of a classifier. This is known as the *curse of dimensionality*, which refers to the tendency that too many dimensions tend to confuse, rather than aid, classifiers. In addition, the wavelet coefficients at high levels provide very low resolution information about the image. This may hinder the accuracy of the segmentation process, as the coefficient values are more likely to span an area containing more than one texture. In other words, at very low resolutions, the wavelet coefficients are greatly downsampled, which would lead to considerable spatial blurring in the features, and generally reduce the segmentation accuracy.

A practical difficulty arises from the use of the DWT due to the contraction of subband sizes at higher levels. Recall that subbands only have one-quarter the size of its parent at each level of a DWT. In constructing feature vectors from subbands of different sizes, an interpolation method is required. This problem does not arise in full-resolution

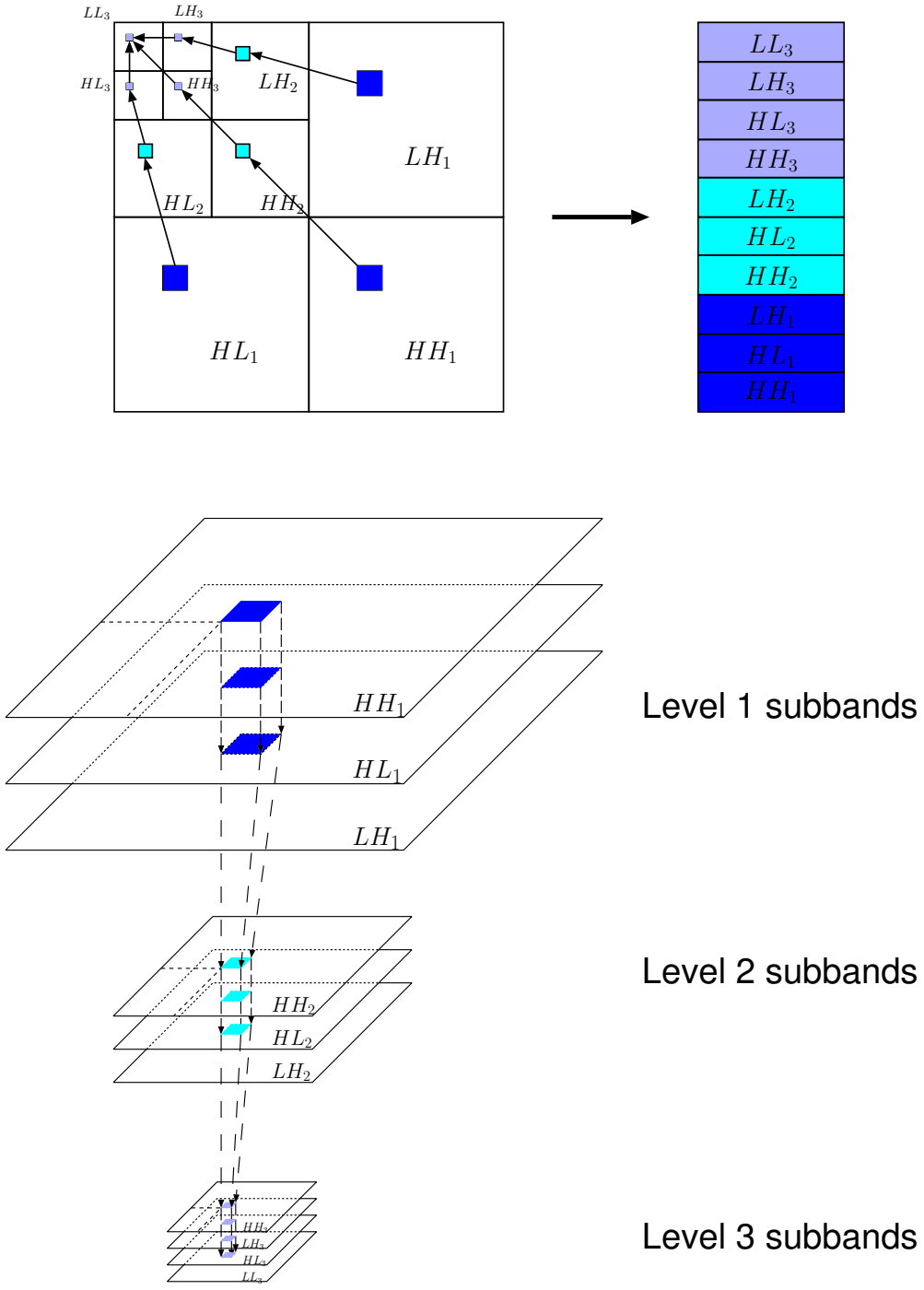


Figure 3.4. The pyramidal feature extraction scheme from wavelet transform coefficients, for a 3-level DWT.

representations like wavelet frames or Gabor transforms. In this approach, the higher level subbands are simply dilated until they assume the same size as the first level subbands ($\frac{M}{2} \times \frac{N}{2}$ pixels). The process is illustrated in figure 3.4 for a 3-level DWT decomposition.

Mathematically, this is written as

$$\begin{aligned} & & \mathbf{w}(i,j,0) &= LL_d([\frac{i}{2^{d-1}}],[\frac{j}{2^{d-1}}]) \\ \text{and} & & \mathbf{w}(i,j,3(d-k)+1) &= LH_k([\frac{i}{2^{k-1}}],[\frac{j}{2^{k-1}}]) \\ & & \mathbf{w}(i,j,3(d-k)+2) &= HL_k([\frac{i}{2^{k-1}}],[\frac{j}{2^{k-1}}]) \\ & & \mathbf{w}(i,j,3(d-k)+3) &= HH_k([\frac{i}{2^{k-1}}],[\frac{j}{2^{k-1}}]), \quad 1 \leq k \leq d \end{aligned} \tag{3.13}$$

where $\mathbf{w}$ is the extracted feature vector, and $[\cdot]$ is the integer division operation. LL, LH, HL and HH represents the subbands based on the filter (low- or high-pass) used in each dimension of the image.

This means that each coefficient in the $k-$th level subbands will be repeated in $2^{k-1} \times 2^{k-1}$ extracted feature vectors. The method used here is perhaps a little *ad-hoc*, but it does appeal for several reasons. Firstly, it is simple and quick to compute, with almost no overheads. More sophisticated interpolation schemes are possible, but those, too, introduce artificiality in the data. Indeed, a conventional wavelet frames representation will be more mathematically elegant if a full-rate analysis is desired. Secondly, the undesirable artificial blockiness introduced by this scheme can be smoothed by windowing techniques, which is performed in the feature conditioning stage (section 3.4). Finally, by introducing partial redundancy in the feature vectors, it is utilising multiresolution implicitly, without the need to design specific pyramidal clustering algorithms. The low-resolution coefficients are more heavily involved in the clustering process, simply because they are repeated in more feature vectors. True pyramidal schemes also tend to use low resolution information more heavily, often as a cue to improve clustering at higher resolutions (the lower level wavelet coefficients). This makes it possible to take advantage of conventional, robust classifiers without compromising the general multiresolution nature of our approach. However, a drawback of the current algorithm is that it can only produce segmented images at half the resolution of the original input image. Obviously, this has a negative impact on the segmentation accuracy, but it does offer lower computational complexity as a tradeoff. Only a quarter of the number of pixels are present when compared to full-rate approaches. An alternative way to achieving full resolution segmentations is to concatenate the zero-level wavelet coefficients, that is, the original image intensities, to the feature vectors. This would mean the addition of another layer on top of the stack

in figure 3.4 by using the original image. This method would also increase the feature vector length by 1, since the original image is virtually like another subband.

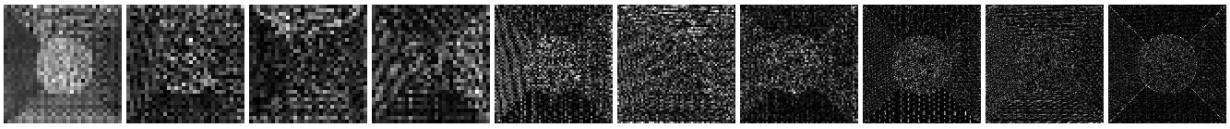


Figure 3.5. Raw extracted features (coefficient magnitudes) from a DWT, using Daubechies 9-7 pair, 3-levels and symmetric extension. The input image is shown in figure 3.3(a). The 10-element feature vectors are collated into images to illustrate the feature variations in different subbands.

The present scheme uses the wavelet coefficient magnitude as the texture feature; this approach has also been pursued by other researchers [62, 63]. Analogous to Gabor filter-based schemes, many other statistics computed from the coefficients are possible alternatives as texture feature. Examples include: coefficient energy [9, 84], mean [20, 42], standard deviation [20], histogram [80], entropy [42] and even co-occurrence features based on wavelet coefficients [80]. The phase of wavelet coefficients also provides information that may be used as texture features [84]; this is behind Laine and Fan's idea to use the zero-crossings of wavelet coefficients to extract texture features [41].

3.3.2 Features From Complex Wavelet Transforms

As discussed in chapter 2, the DWT suffers from a lack of shift-invariance and limited directionality. For texture analysis, it is often desirable to have a more isotropic analytical tool since natural textures can exhibit features at arbitrary orientations. The DT-CxWT offers better performance than DWT in this regard, but it also exhibits the desirable properties of DWT. This superior performance comes at a cost of 4 times redundancy in the representation for 2-D data.

With a DT-CxWT decomposition, the feature extraction scheme described in the section 3.3.1 must be modified to accommodate the increase in the number of subbands. This is done by extracting the features from each of the four individual transform images using the process described in the previous section, then the final feature vector is the concatenation of these feature vectors, as shown in figure 3.6. The consequence of using a DT-CxWT is a 4-times increase in the feature vector length, a direct consequence

of the redundancy of this transform. An alternative would be to compute the *complex magnitudes* of the DT-CxWT coefficients as the features. Recall from section 2.6 that the four subbands from a 2D DT-CxWT consist of the real and imaginary parts of two wavelet trees. Hence, by taking the complex magnitude leads to a reduction in the redundancy, and therefore the feature vector length, by a factor of 2. This means that the DT-CxWT feature vectors length is at a more economical factor of two greater than for the DWT.

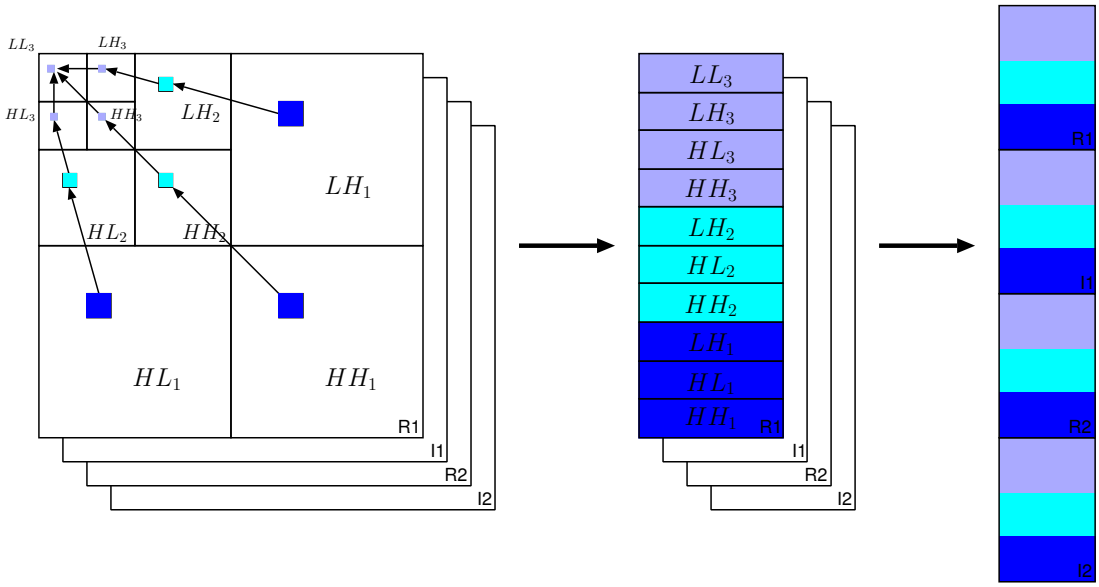


Figure 3.6. The pyramidal feature extraction scheme from DT-CxWT coefficients.

Using a magnitude appeals intuitively, since it corresponds to an energy measurement. The phase of the DT-CxWT coefficients contain information that is most useful for motion analysis in video sequences [44], since the phase shift is approximately linearly dependent on the spatial shift of image features. Preserving such information may not be beneficial to texture analysis, especially in the light of dimensionality considerations.

3.4 Feature Conditioning

The feature conditioning phase is essentially an intermediate processing layer between the raw features and the clustering algorithm (see figure 3.1). It should produce features that are friendly to the chosen classifier, but it should not do so at the expense of decreasing the features' information content. Figure 3.8 details the feature conditioning process used

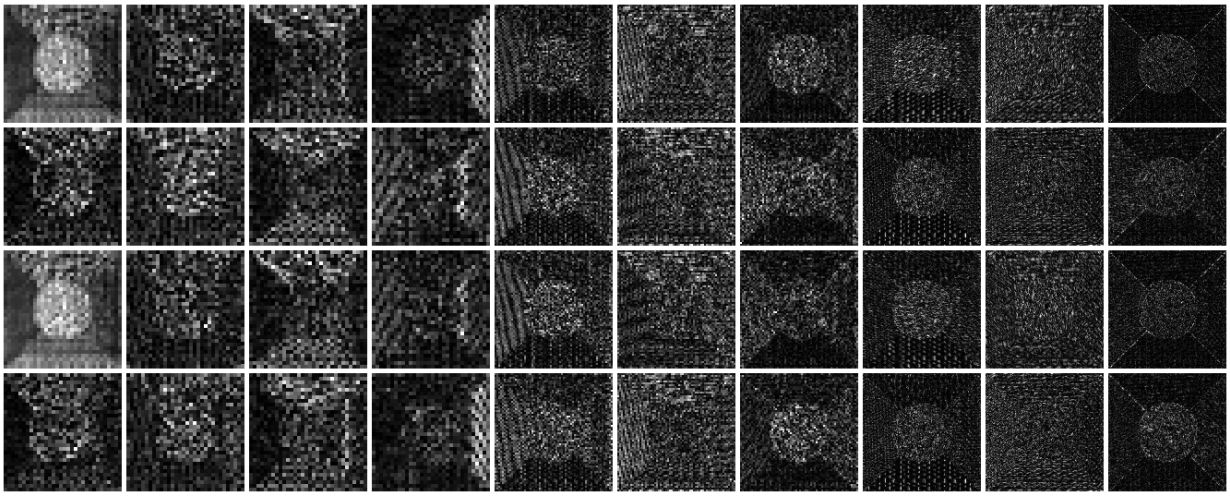


Figure 3.7. Raw extracted features (coefficient magnitudes) from a DT-CxWT, using the Kingsbury filter pairs, 3-levels and symmetric extension. The input image is shown in figure 3.3(a).

in this thesis. It consists of four steps: feature reduction, normalisation, smoothing and a non-linear transformation. The following subsections will discuss these steps in turn.

3.4.1 Feature Reduction

The role of feature reduction (or feature selection) has long been recognised in pattern recognition. In many problems, high-dimensional data are necessary to adequately describe complex objects and to distinguish between them. However, as the dimensionality rises, the extra information may not assist in discriminating between different classes. Indeed, too much information may actually cloud the distinction between classes, if a significant part of those features are non-descriptive. This is commonly referred to as the *curse of dimensionality* [2]. To avoid this problem, it is often desirable to reduce the dimensionality of the feature data through some selective procedure. The appropriate selection of features depend greatly on the the intended classifier or clustering algorithm, and to a certain extent, the feature extraction method as well. In some instances, the feature reduction step is unnecessary, and features are fed directly as input to the clustering algorithm. These are typically for cases where the feature extraction step has yielded a condensed set of features, and the clustering algorithm is sufficiently powerful to deal with the features directly. However, most sophisticated texture segmentation algorithms

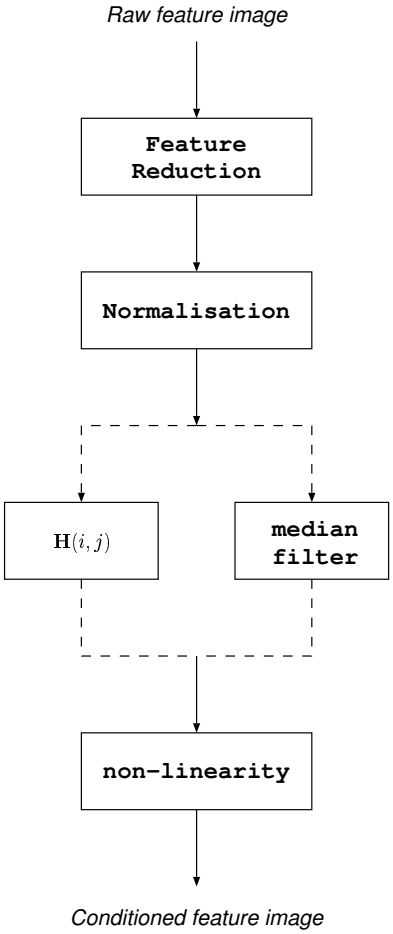


Figure 3.8. Block diagram of the feature conditioning process.

use some form of feature reduction. Many methods have been investigated by pattern recognition researchers, and a sample of them will be described below.

The most common dimensionality reduction method is the Karhunen-Loéve transform, also known as the Hotelling transform or Principal Component Analysis (PCA). In this method, the set of features are projected onto a set of orthogonal axes, obtained through a rotation of the feature space. The particular set of orthogonal axes are chosen to be the normalised eigenvectors of the covariance matrix of the (feature) vectors. Mathematically, a set of vectors $\mathcal{S} = \{\mathbf{x}\}$ with mean $\overline{\mathbf{x}}$, has a covariance matrix $\mathbf{C_x}$ defined by

$$\mathbf{C_x} = E\{(\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^T\}$$

where $E\{\cdot\}$ is the expectation operator.

Since $\mathbf{C_x}$ must be real and symmetric, it is possible to find a set of orthonormal eigenvectors of $\mathbf{C_x}$,

$$\begin{aligned} \mathbf{C_x e_i} &= \lambda_i \mathbf{e_i}, \quad 1 \leq i \leq n \\ \mathbf{e_i} \cdot \mathbf{e_j} &= \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \end{aligned}$$

where n is rank of $\mathbf{C_x}$. Without loss of generality, assume the eigenvalues are ordered as $\lambda_1 > \lambda_2 > \cdots > \lambda_n$. If the matrix $\mathbf{A}$ is defined by

$$\mathbf{A} = [\mathbf{e_1 e_2} \cdots \mathbf{e_n}]^T$$

then the transformation

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \overline{\mathbf{x}})$$

is the K-L transform. The principle behind the K-L transform is that the axes are aligned to the components with maximal energy. Therefore, the K-L transform has the useful ability to find the least square error approximation to a given feature set using a reduced number of axes (components in feature vector) - the transformation matrix $\mathbf{A}$ in equation (3.4.1) would simply have fewer rows than the dimensionality of vectors $\{\mathbf{x}\}$. Much of the mathematics of the K-L transform are extracted from [25], and more details on the theory and application of K-L transforms in image processing can be found in that book.

The feature selection step is sometimes built into the feature extraction methods directly. Chang and Jay-Kuo [9] exploited the natural tree structure of their wavelet transform to serve as a feature reduction tool. Jain and Farrokhnia [32] and Teuner *et al* [74] used adaptive filter selection schemes, which effectively selects the features that the extraction step produces. Greater details on these schemes were presented in the section 3.2.2.

A popular method to select feature subsets is by the use of discriminants or separability measures. These provide a means to measure whether clusters in a feature set is capable of being separated. One example of this technique is the spatial separation measure, defined as the ratio of intra-cluster to inter-cluster distances. With a training set, it is possible to calculate these distances explicitly. A small value for this ratio would indicate that the feature set is likely to be separable. The main advantage of this method is its simplicity and intuitiveness. These appealing aspects have made spatial separation a usable method for feature selection. On the other hand, this simple measure can be

distorted by very large clusters (in the feature space) and provide inaccurate description of the features. The main problem with this measure is that it does not properly account for the locations of class centroids [24]. Teuner *et al* [74, 55] used contrast information (equation (3.12)) to select features. This has the advantage of having very low computational complexity. Fatemi-Ghomi [24] has considered two different separability measures for feature selection: the distance histogram and the two-point correlation function. The distance histogram method constructs the histogram of all pair-wise distances in the feature space. It is similar to the spatial separation measure in spirit; it relies on a separable feature set having a histogram with distinct peaks corresponding to the inter-cluster and intra-cluster distances. Again, a separable set may not always give rise to well-defined peaks, and there are clear cases of a separable set with a single-peak histogram. However, the distance histogram does provide an intuitive visual cue to selecting suitable features. The two-point correlation function method is a generalisation of the distance histogram. Instead of constructing a histogram of pair-wise distances, it computes the probability density of distances. It is constructed to give an unbiased statistical measure, and generally has better reliability than the distance histogram. However, it suffers from computational difficulties for high input space dimensions. In this situation, an enormous amount of input data is necessary to estimate the densities, and the computational problems greatly reduce the effectiveness of the measure.

De Backer *et al* have investigated several non-linear dimensionality reduction techniques [66]. They compared four different techniques: multidimensional scaling (MDS), Sammon's mapping, Self-Organising Map (SOM) and Auto-Associative Feedforward Neural Networks (AFN). Multidimensional scaling produces an output space whose distance between a given pair samples matches that of the corresponding pair in the original, higher-dimensional space as much as possible. This preserves the degree of (dis-)similarity between feature samples (metric MDS), or at least the rank order of the distances (non-metric MDS). Sammon's mapping produces a mapping that minimises the sum of weighted distance differences between the input and output spaces. The weights are designed to place greater emphasis on smaller distances, which is important for fine distinction between different classes. Self-Organising maps are a topology-preserving technique. It produces its mapping through the training of a set of neurons, spaced regularly in the output space. These neurons have a set of weights with the same dimensionality as the input

space. The weights of each neuron are trained to match the features within a local neighbourhood in the input space (minimises a squared error criterion). An auto-associative feedforward network is a multi-layer neural network, which is trained such that it reproduces the original features in the output layer. The idea is to use the middle hidden layer as the representation in output space. Thus, the number of neurons in the hidden layer would be the dimension of the output space. In this method, it is possible to use multiple layers to achieve good non-linear representations in the hidden and output layers. These four dimensionality reduction techniques were tested with classification experiments performed on both artificial and real-world data sets. The real-world datasets were features extracted from gray scale and colour texture databases. The non-linear techniques were also compared against the linear PCA. It was found that, for aggressive dimensionality reduction (very low dimensionality in the output space), the self-organising map was able to achieve the best classification rates for all experiments. However, at higher output dimensions, the linear PCA was able to achieve very similar or better classification rates as the non-linear techniques.

More recently, Pan has attempted to use genetic algorithms (GA) to perform feature reduction [52]. In this method, he used the spatial separation ratio as the criterion for the fitness function in the GA. Each gene is constructed as a binary string indicating whether a feature component is used. The usual genetic operators such as crossover (reproduction) and mutation are applied. The optimal gene gives the final reduced feature set, which is then used for segmentation experiments. The use of GA to select features consistently produced better results than the full feature set, indicating a genuine improvement in the separability in the reduced set.

The DWT and DT-CxWT features extracted from sections 3.3.1 and 3.3.2 have quite moderate dimensionality. For example, when employing a transform of depth 3, the DWT and DT-CxWT schemes would lead to feature vectors of length 10 and 40, respectively. These dimensions are reasonable, and the features are capable of providing good segmentation accuracy (see a discussion of results in Chapter 5). Consequently, feature reduction is only optionally implemented in this algorithm for simplicity.

3.4.2 Normalisation

The raw features extracted from the transform coefficients are unsuitable for use in clustering algorithms. First of all, the energy-compacting nature of wavelet transforms means that the low-pass subbands have far higher energies than the high frequency subbands. Typically, more than 90% of energy is compacted into the LL subband at each level of a wavelet transform. Figure 3.9 illustrates this behaviour for 3-level DWTs of the ensemble of texture mosaics shown in Appendix A. When the raw features are fed into typical classification algorithms, the low-frequency subband(s) tend to dominate the decision-making process, essentially ignoring the finer information contained in the mid- and high-frequency subbands. In other words, the classification will be mainly based on average brightness levels, and not on texture information. Thus, a mechanism of balancing the dynamic ranges in the feature values across all the components is required.

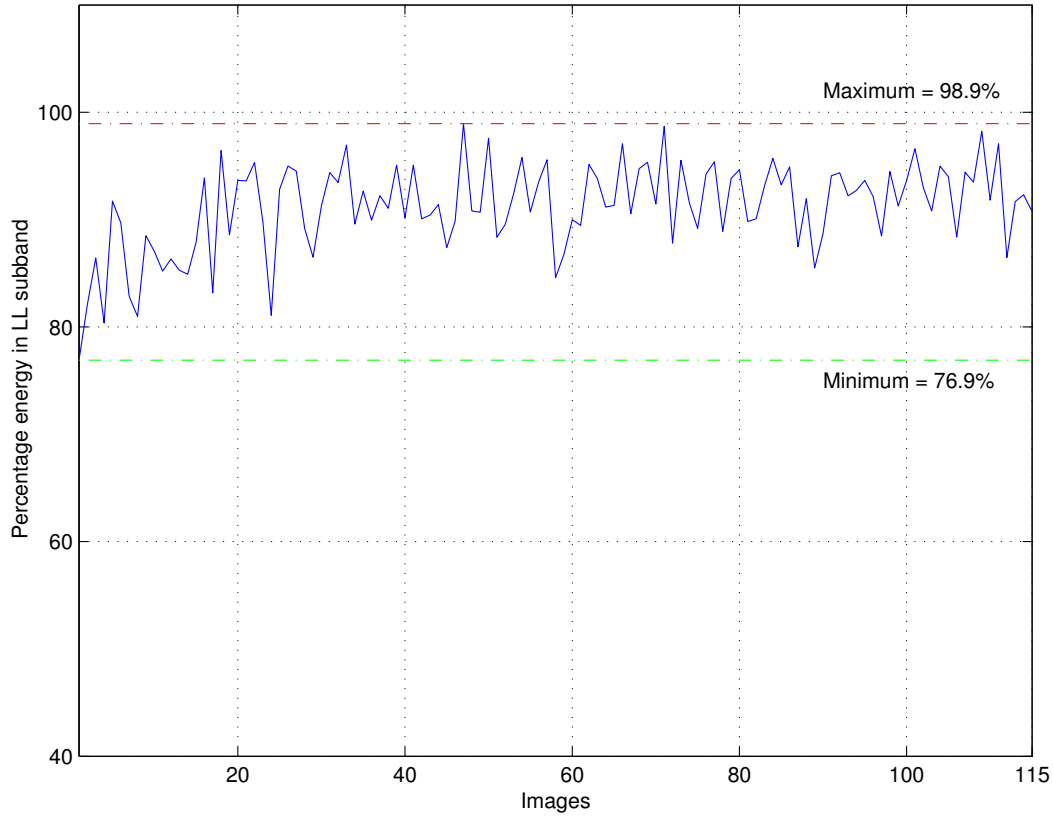


Figure 3.9. Energy contained in the LL subband for a collection of texture mosaics, expressed as a percentage of total image energy. Daubechies 9-7 pair is the analysis pair used in the 3-level transform with symmetric extension.

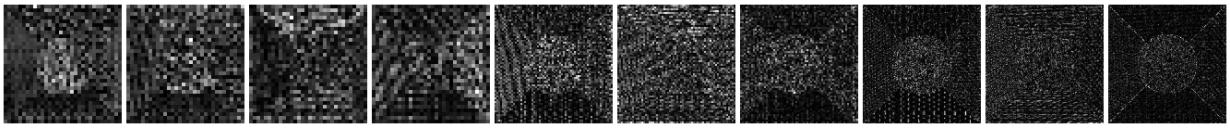


Figure 3.10. Normalised features from a DWT, with the same setup parameters as in figure 3.5.

The first step of the feature conditioning process is to compute the normalised wavelet features at each pixel. Recall that the raw feature values are simply the absolute magnitudes of dilated wavelet coefficients. These magnitude values are shifted and normalised to have zero mean and unit variance for each subband. This step ensures the higher energy subbands (i.e. the LL subband) do not dominate the clustering process later.

3.4.3 Windowing

In addition to numerical problems, raw wavelet energy features tend to have very poor smoothness. Fundamentally, textured images are generally not smooth, with an abundance of large spatial discontinuities. This behaviour also manifests itself in the wavelet transforms of textures in the form of large spikes and troughs in the coefficients. However, in the interest of pattern recognition, it is necessary to encourage greater similarities in the features inside a homogeneously textured region. In particular, the high frequency wavelet transform coefficients have rapid spatial variations, as a result of the fine details present in many example textures. This would cause the high frequency features extracted from neighbouring pixels to differ significantly, even if these pixels belong to the same texture. Obviously, this will have a negative impact on the segmentation result. Thus, a smoothing operation that concentrates on extracting local information, rather than pixel-precise information, is required.

Therefore, the second feature conditioning step involves spatially smoothing the features. An adequate amount of smoothing on the wavelet coefficients greatly assists the clustering process (refer to figure 3.1), both in terms of segmentation accuracy and convergence speed. Smoothing by low pass filtering (also called *windowing*) is a common technique for achieving this. An appropriate choice of window is important in controlling the amount of smoothing applied to the texture features. The windowing process is

represented by

$$\mathbf{x}_s(i,j,k) = \sum_{i',j' \in H} H(i',j') \cdot \mathbf{w}(i-i', j-j', k) \tag{3.14}$$

The smoothing window is applied to each subband by filtering component k in the raw feature vectors $\mathbf{w}(i,j,k)$ with window filters $H(i,j)$ to give smoothed features, $\mathbf{x}_s(i,j,k)$. Symmetric extension is used at the subbands' edges to avoid artificial discontinuities. An important consideration here is the shape and spatial extent of the windows. The window should be sufficiently large for effective smoothing, yet not so great as to blur out texture region boundaries. The optimal window is difficult to estimate, and an empirical approach is taken in this thesis. One can view the smoothing process from the perspective of textons. Textures with relatively large textons are likely to require larger smoothing windows for adequate smoothing, and vice versa. The shape of the window also plays a role in the effectiveness of the smoothing process.

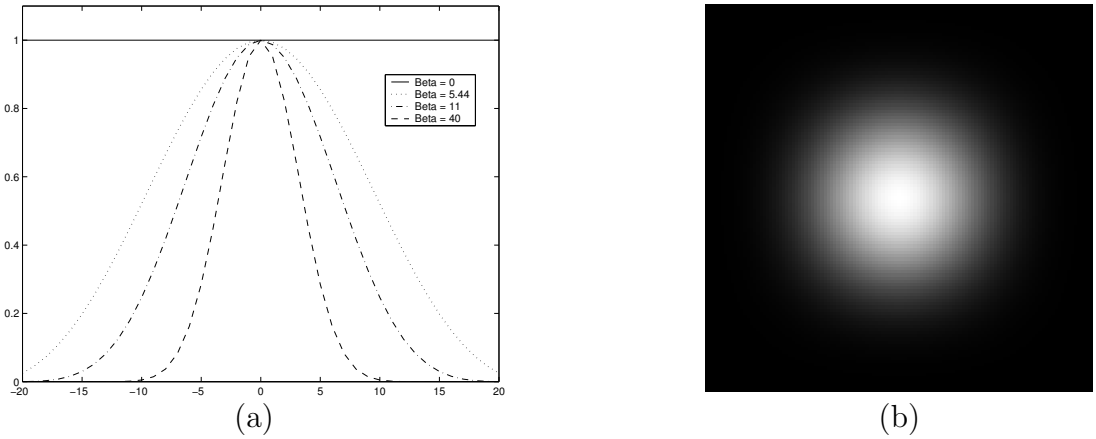


Figure 3.11. The Kaiser window functions, in (a) 1D and (b) 2D. For the 1-D plots, β is varied while the size of the window is fixed; the 2-D image shows a window of size 121 and $\beta = 11$. The size is chosen to be so large to better illustrate the shape of the window; a smaller window with the same β will have identical shape.

Traditionally, rectangular or Gaussian windows are the most popular. The rectangular window, while simple, tends to introduce boundary problems, due to its anisotropic nature. Texture information is generally not restricted over rectangular regions. The Gaussian window is commonly used to overcome this problem, as it is known to have the optimal time-frequency trade-offs. However, other windows offer better frequency resolution or side-lobe attenuation than the Gaussian window. Two such examples are the

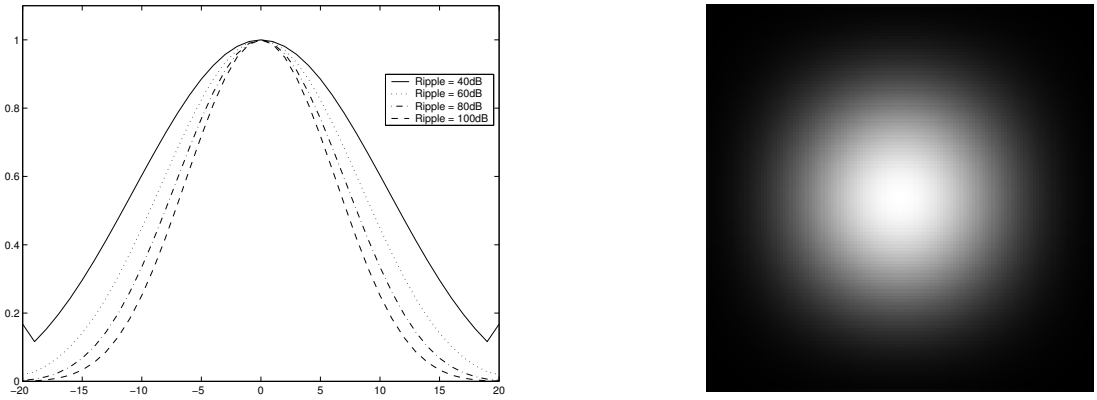


Figure 3.12. The Chebyshev window functions, in (a) 1D and (b) 2D. For the 1-D plots, ripple value is varied while the size of the window is fixed; the 2-D image shows a window of size 121 and ripple set at 60dB.

Kaiser and Chebyshev families. The Kaiser window is most often used in the design of FIR filters, where it excels in providing low sidelobe levels in filters. The Kaiser function is defined as

$$w_K(n) = \begin{cases} \frac{I_0(\beta\sqrt{1-(n/N_1)^2})}{I_0(\beta)} & -N_1 \leq n \leq N_1 \\ 0 & \text{else} \end{cases} \tag{3.15}$$

where $I_0(x)$ is the zeroth-order Bessel function of the first kind, and β is a parameter that controls the shape of the window. For example, for $\beta = 0$, the Kaiser window is equivalent to the rectangular function, and for $\beta = 5.44$, the Kaiser window is similar to a Hamming window. From a filter design's perspective, the value of β controls the level of the sidelobes in the frequency response. In the current context, the value of β may be used to control the relative sizes of the textons we wish to describe, since β directly controls the relative spatial extent of the window (see figure 3.11(a)). Like the Kaiser window, the Chebyshev window is mostly used in filter design applications. Chebyshev filters produce constant ripple responses in time domain, and exhibits the minimum main-lobe width for a specified sidelobe level in the frequency domain.

An alternative to filtering by low-pass filter is with median filters. The median operator is insensitive to extreme values (unlike any mean), and is therefore tolerant of outliers. It has been utilised in many image processing applications to perform smoothing. To perform median filtering, an $M \times M$ window is centred on a pixel in a feature image.

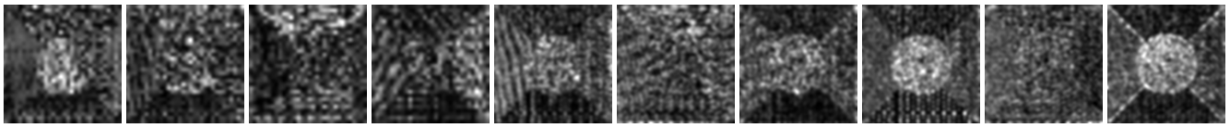


Figure 3.13. Smoothed features from a DWT with a Kaiser window. This has the the same setup parameters as in figures 3.5 and 3.10. The smoothing window used is a Kaiser window, size 11 and $\beta = 8$.

The median value of the M^2 pixels masked by the window is used as the smoothed feature for that pixel.

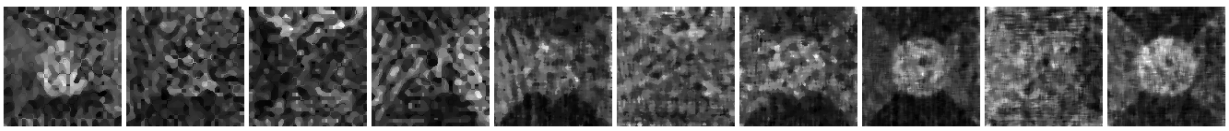


Figure 3.14. Smoothed features from a DWT with median filtering. This has the the same setup parameters as in figure 3.13. A 7×7 median filter is used to achieve the smoothing.

3.4.4 Non-linear Transformation

After smoothing, the features are further conditioned by a non-linear transform, which limits the dynamic ranges of the different feature components. In addition to the earlier normalisation step, this processing further avoids the problem of having a small number of feature vector components dominate the clustering decisions, which deprives the clustering process of equal consideration for all the available information. For the feature processing in the experiments, the following sigmoidal non-linearity is used:

$$\mathbf{x}_{st}(i,j,k) = \tanh(\alpha \mathbf{x}_s(i,j,k)) \tag{3.16}$$

where α is a freely variable parameter. Jain [32] first suggested the use of the sigmoidal non-linearity for feature conditioning, and likened this operator to a "blob-detector". In effect, this transformation "recognises" when a feature has significant value or not. This function was also used by [61],[64] for their texture feature processing. It has been suggested that a value of $\alpha = 0.25$ is most suitable for Gabor filter-based features. Figure 3.15 shows the sigmoidal function for a range of different α. Other feature conditioning processes, like different non-linearities or a soft-thresholding scheme, are possible. However,

these schemes were found to give equal or inferior segmentation performance from experimentations.

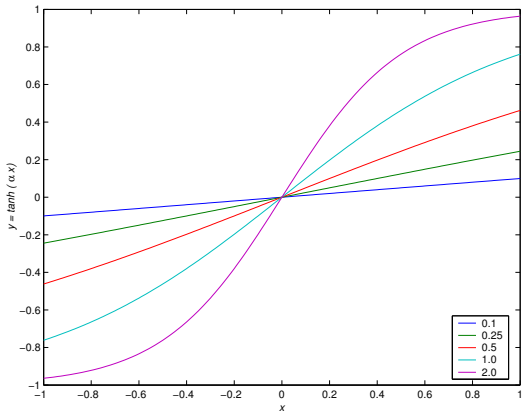


Figure 3.15. tanh for different values of α.

tanh for different values of α.

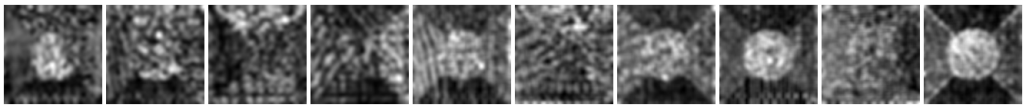


Figure 3.16. Extracted wavelet features for a texture mosaic. These features have been processed through all the feature conditioning steps described here, with Kaiser window smoothing.

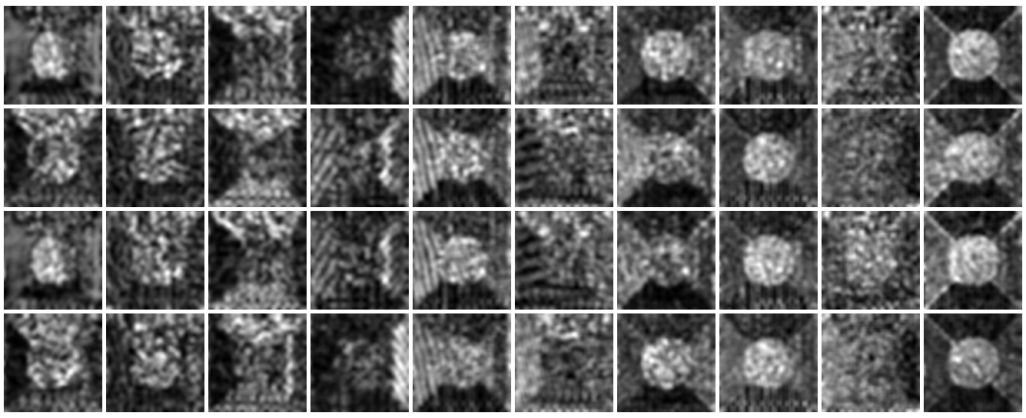


Figure 3.17. Extracted complex wavelet features for a texture mosaic. The 40-element feature vectors are collated into images to illustrate the feature variations in different subbands.

In general, the feature conditioning phase can become complicated and overloaded with a large set of parameters. Good parameter combinations are generally data dependent, and are difficult to determine *a priori*. Ideally, the system should be capable of segmenting textured images with minimal prior knowledge and auxiliary inputs (additional parameters). A key advantage of the feature extraction algorithm described in this chapter is that it only requires a small number of free parameters to produce consistent and effective segmentations.

3.5 Summary

This chapter discusses texture feature extraction. Throughout the history of texture analysis, a large part of researchers' efforts have been spent on uncovering descriptive, compact features for textures. This has proved to be rather difficult, due mainly to the complex nature of textures as a phenomenon. Early efforts have concentrated on pure statistical or model-based (structural) approaches, but neither has been fully satisfactory. With greater advances in mathematical signal decomposition, a myriad of multiresolution approaches were made possible. These have been found to be more suitable for textures in general. A novel feature extraction method based on DWT and its extension to support DT-CxWT was proposed. These methods grew out of earlier techniques for extracting texture features from other multiresolution representations. The extracted texture features must then be subjected to a series of post-processing steps, collectively known as the feature conditioning process. These include feature reduction, numerical normalisation, windowed smoothing and non-linear transformation. The end result of this processing is a set of texture features which can be clustered to produce segmented regions.

This page is blank

Chapter 4

Texture Feature Clustering

The final phase of the texture segmentation algorithm is the clustering phase (figure 3.1). In the context of this thesis, clustering of the image pixels is achieved by classifying the conditioned feature vectors into appropriate groups. Therefore, for all intents and purposes, the term "clustering" will be used interchangeably with "classification" in the remainder of this thesis.

This chapter discusses several generic classification and clustering algorithms. In pattern recognition literature, numerous classifiers have been studied, and it is beyond the scope of this work to examine them all for texture segmentation. Instead, a number of these will be discussed, while the majority of results in this thesis are generated with the K-means algorithm.

4.1 K-means Clustering

A popular and commonly-used clustering algorithm in pattern recognition is the *K-means Clustering* algorithm [26]. This is an example of classifiers based on the general method of evolutionary search. The algorithm begins with an arbitrary initial clustering and subsequently consider each sample (feature vector) as candidates of reallocation. The cluster updating and reallocation are reiterated until further reallocations are no longer possible. This general class of algorithms has the advantage of simplicity and guaranteed convergence to at least some local extremum. However, there is the risk that the algorithm

may converge on a non-global optimum; such behaviour is also exhibited by all hill-climbing type of optimisation algorithms. To be more precise, the K-means clustering algorithm can be described by the following steps:

1. **Initialisation** - the feature image of $n = N_1 \times N_2$ samples is randomly initialised. The cluster means, $\{\mu_i | 1 \leq i \leq K\}$, are arbitrarily determined, while the number of clusters, K, is specified by the user;

2. **Update clusters** - for every sample, $\mathbf{x}(i,j), 1 \leq i \leq N_1, 1 \leq j \leq N_2$, the distances to all the cluster means are computed; its membership is reassigned to the cluster whose mean is closest to the sample, i.e. $\mathbf{x} \in C_k \iff d(\mathbf{x}, \mu_k) < d(\mathbf{x}, \mu_l), 1 \leq l \leq K, l \neq k$. This is repeated for all n samples;

3. **Compute cluster means** - for each cluster, the mean is recomputed by averaging the features of all members in that cluster;

4. **Reiteration** - Steps 2-3 are reiterated until no further reallocation is possible.

Figure 4.1 illustrates the K-means clustering process. Generally, K is assigned as the desired number of clusters. In some pattern recognition applications, it is possible to specify K *a priori* to the clustering process. In texture segmentation, it is not always possible to do so. In our experiments, this parameter is fed to the clustering algorithm manually, i.e. it is assumed that the number of classes is already known. In practical applications, this scenario isn't entirely unrealistic, as the number of textures likely to be present in an image is usually known. For example, in aerial surveillance, one might only be interested in segmenting a few distinct, known land use regions in a photograph. Since a value of K is required to be known, the K-means algorithm is usually classified as being a supervised algorithm. In the traditional sense, this kind of supervision is distinctly different from, say, neural network classifiers. The supervision is not exercised in such an explicit form as in the expensive training phase of neural network algorithms. Rather, the supervision is an inherent limitation within which the K-means classifier must function.

The computational complexity of the algorithm is $O(ndKT)$, where n is the the number of samples (feature vectors), d is the number of feature components, K is the number of clusters (classes) and T is the number of iterations. In theory, the K-means algorithm may not always converge. Such can be the case for very ill-conditioned feature

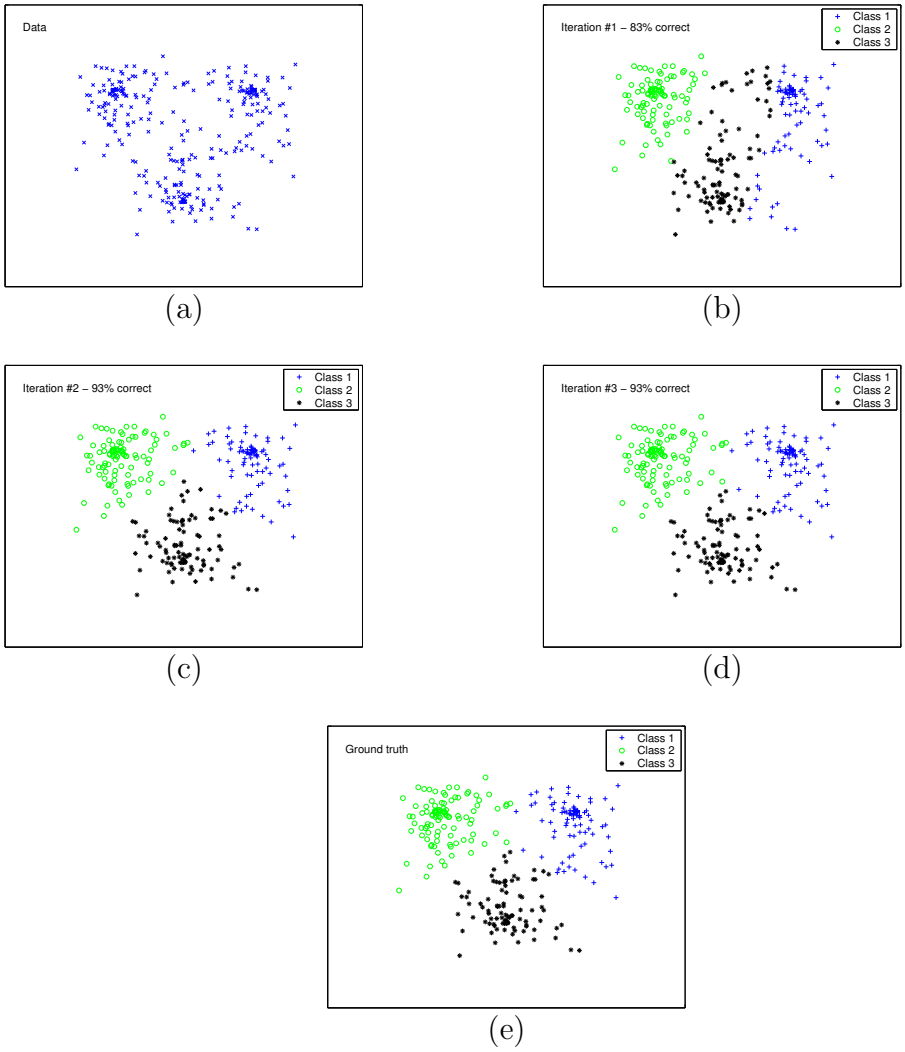


Figure 4.1. The K-means clustering algorithm in action for a 2-D dataset. (a) Input data, 150 samples from 3 different distributions; (b)-(d) clustered data after 1 (b), 2 (c) and 3 (d) iterations; (e) Ground truth clustering. While the algorithm takes 3 iterations to converge to a segmentation, it can be seen that the error rate has not improved since the 2nd iteration. The errors have simply been re-distributed between the different clusters.

sets with inappropriate initialisation. In such instances, the numerical difficulties inherent in the problem prohibits the K-means algorithm to generate meaningful results, resulting in the failure to converge. In practice, the algorithm is terminated after a set number of iterations (about 1000) for the experiments, and the results are discarded. For these cases, a different initial condition may yield better results; Section 4.1.1 has more details on this.

For its simplicity, K-means clustering is widely employed in many tasks. However, it has several shortcomings; poor computational scalability, a habitual convergence to local extremum and the need for *a priori* knowledge of K. Fast algorithms for computing K-means clustering do exist, but performance issues are not within the scope of this dissertation. For the vast majority of cases studied here, algorithm performance has not been an issue, even with the use of relatively modest equipment. In general, it takes several seconds to compute the segmentation for most test images on a 450MHz x86 PC. Much more interesting, in the context of this thesis, are the latter two shortcomings. These will be examined in the following sections.

4.1.1 Initialising K-means Clustering

A major issue with the K-means clustering process is with the initialisation step. The important fact to be realised when using K-means is that the process is entirely deterministic, once given an initial condition. In a problem of even modest complexity, there may exist many local extrema which the algorithm will converge to, herein lies the fundamental problem of K-means. The most common way of starting the process is allocating arbitrary class labels to the individual samples, or feature vectors in this instance. A particularly simple way to do this would be to assign class labels cyclically on the individual samples. That is,

$$c_0(i) = i \bmod K \tag{4.1}$$

where $c_0(i)$ is the initial class label for the i-th sample. For the two dimensional case, the individual feature vectors are labelled in a row-by-row fashion. A simple alternative way to initialise the problem would be to use random labelling to begin with. Then the K-means algorithm is run many times, using a different random initial starting point every time. Hopefully, the random nature of these different initialisations would lead to a variety of output solutions, some of which may bear close resemblance to, or even be, the actual optimal solution. However, the complexity of the problem is practically unmanageable, for case of texture segmentation. An $N \times N$ image comprising K textured regions has KN^2 distinct possible initialisations. Obviously, for any image of useful size, this is quite a large number of cases to consider. Therefore, an intelligent choice of initial clustering is useful, not only in securing a near-global optimal clustering, but it may also speed up the process tremendously, by reducing the necessary number of

iterations for convergence. This is especially true for large data sets. A computationally intensive way to initialise the K-means algorithm would be to perform a mini-K-means clustering on a small subset of samples. In particular, select p samples from the total set, and perform a K-means clustering on this small problem. The result of this sub-problem should be less sensitive to its initialisation, provided that the set of p samples is representative of the overall set. The resulting K class means can be used to initialise the main problem. Ironically, the K-means algorithm suffered the most from initialisation problem when used on pathological data sets. For experimentation, a test data set of perfectly uniformly distributed classes was fed to the algorithm; in this set, each feature is simply an integer between 1 and K. This simple, highly separable data set lead to a lot of numerical problems with the K-means clustering. The class means converged prematurely and several individual clusters merged together into a fewer number of giant clusters. These are due to the arbitrary fashion of initial labelling, and the resultant similarity of all class means, leading to very fickle and numerically ill-conditioned class relabelling decisions. These phenomena occurred irrespective of the initialisation scheme (cyclic or random); very few initialisations actually led to the correct, globally-optimal solution. It was found that, by performing many separate K-means runs, the algorithm is capable of producing a small handful of local extrema. It was clear that a pre-processing step that is capable of initialising the problem is very desirable, particularly if the main algorithm seems to be producing unstable results. Fortunately, though, it is found that the segmentation output is quite insensitive to the initial conditions for the features extracted by the process described in Chapter 3.

4.1.2 Distance Measures in K-means

A central part of the K-means clustering algorithm is the computation of distances between each sample and the cluster means. In most cases, metric measures are used, and by far the most popular is the weighted Euclidean distance,

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{k=1}^{m} w_{ijk}(\mathbf{x}_i(k) - \mathbf{x}_j(k))^2}{\sum_{k=1}^{m} w_{ijk}} \tag{4.2}$$

$d(\mathbf{x}_i, \mathbf{x}_j)$ is the Euclidean distance between m-component samples, $\mathbf{x}_i$ and $\mathbf{x}_2$, and w_{ijk} is a generalised set of weights that adjusts the relative importance of different components. The samples are referred to by a single index, instead of the full two-dimensional indices, for convenience. In practice, the weights w_{ijk} are uniform for all pairs of samples, and can therefore be incorporated directly onto the sample components, thus saving the need to compute during iterations. The unweighted Euclidean distance is more generally known as the L^2-norm. A general L^p-norm is defined as

$$d(\mathbf{x}_i, \mathbf{x}_j) = \left[\sum_{k=1}^{m} (\mathbf{x}_i(k) - \mathbf{x}_j(k))^p \right]^{1/p} \tag{4.3}$$

Apart from the Euclidean metric, other important special cases are the L^1 and L^∞-norms. The former is also known as the "Manhattan", or taxi-block, metric, while the latter is the maximum component difference measure:

$$L^1 = \sum_{k=1}^{m} |\mathbf{x}_i(k) - \mathbf{x}_j(k)| \tag{4.4}$$

$$L^\infty = \max |\mathbf{x}_i(k) - \mathbf{x}_j(k)| \tag{4.5}$$

Statistically, the Malahanobis distance takes into account the different variances in the individual components:

$$d_M(\mathbf{x}_i, \mathbf{x}_j) = \left[\sum_{k,l=1}^{m} (\mathbf{x}_i(k) - \mathbf{x}_j(k)) c_{kl} (\mathbf{x}_j(l) - \mathbf{x}_j(l)) \right]^{\frac{1}{2}} \tag{4.6}$$

where c_{kl} are the elements of the covariance matrix for the components of the feature vectors. The advantage of the Mahalanobis distance is that no single component dominates in the distance metric calculations, which can lead to a deterioration in performance.

Non-metric distance measures can also be used in K-means. The general similarity measure is a scoring system, where a higher value indicates greater similarity between two samples. This is the exact opposite of distance metrics. An example of a similarity measure is

$$s(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{m} 1 - |\mathbf{x}_1(i,j,k) - \mathbf{x}_2(i,j,k)| \tag{4.7}$$

assuming the components of the samples are normalised to have a dynamic range between 0 and 1. This type of measure is especially appropriate for binary data, where each component may represent a particular attribute of the sample. In this case, the similarity

measure in equation (4.7) is simply the total number of attributes shared between two samples. A similar interpretation does not extend to continuous data, and such measures are not commonly used for such cases.

4.1.3 Estimating K

Since K is the vital parameter in the K-means algorithm, its estimation is of prime importance in all applications. In the simplest form, K is simply supplied by the user, and the algorithm would faithfully produce a K cluster output, with the possibility of empty clusters. However, in practical applications, the exact number of clusters in a data set is not necessarily known. It is more reasonable to expect a range of possible numbers, but the basic K-means algorithm cannot handle multiple K values. Researchers have devised several methods to overcome this difficulty. Some are based on experimentation with a number of different K values, i.e. K-means is run repeatedly with different parameter values. A precise value of K is determined by analysing the outputs, with respect to some measure of merit for the clusterings. Generally, these measures examine the self-similarity of individual clusters, as well as the relative disparity between different clusters. Examples of these include the average cluster radius and inter-cluster distances. The choice of these measures varies from problem to problem. The final value of K is taken to be that which yields the best clustering, subject to the measures chosen. Another popular method of estimating K begins with a K-means run with a large value, K_{max}, supplied by the user. In the extreme, this may be as large as n, the total number of samples in the data set. The outputs are then analysed and candidates for pair-wise merging are searched for. Usually, the criteria for ranking these candidates are similarity or distance measures. For example, the Gaussian distance between class means can be used as the measure. The merging is done iteratively until a value of K is reached where further merging does not improve the clustering. Typically, the total squared error sum is used as the measuring criterion. This is simply the sum of the squared distances from all the samples to their respective class means. In Pelleg's X-means algorithm [54], the reverse approach is taken. Lower and upper bounds for K are specified by the user, and the algorithm then performs a conventional K-means clustering using K_{min}. Afterwards, every cluster is split into two children, and a localised K-means ($K = 2$) algorithm is run on all of them, using randomly determined initial class means. All the split clusters are scored against the

original parent cluster, using a scoring system based on an estimated Gaussian posterior probability distributions for the clusters. If the children's combined score exceeds that of the parent cluster, then the parent is replaced by its offsprings, increasing the overall K by one. Otherwise, the offsprings are ignored. The X-means algorithm then reiterates until K exceeds the specified upper bound, at which stage the algorithm terminates and returns the result with the highest score as the final output. The X-means algorithm has the advantage of speed and a guided means to estimate K. It has been tested on Gaussian data sets and found to achieve good results.

4.1.4 Algorithmic Variations of K-means Clustering

Due to it long history and popularity, numerous variations on the basic K-means algorithm exist. It is beyond the scope of this thesis to provide an exhaustive treatment. Instead, a few of the most relevant variations are briefly described in this section.

Dynamic K-means Clustering

Due to its popularity and long history, many variations of the fundamental K-means clustering (or "naive", according to [53]) exist. The most basic variation lies with the calculation of the class means after the update pass. In the traditional algorithm, new means are calculated after every sample has been relabelled. A dynamic update procedure, in which class means are always updated with every change in class membership, can be used instead. This method has the advantage of the means always being up-to-date in representing the true mean of all membership samples. However, the computational burden is greatly increased, because *two* class means must be recomputed with every change in sample class, since both the previous and new class means need to be updated. However, a dynamically updated K-means algorithm may require fewer iterations to converge, depending on the particular feature set.

1-pass K-means Clustering

An interesting, yet crude, variant is the 1-pass K-means algorithm. Instead of beginning with some initial clustering and a fixed K, this algorithm requires only a threshold. The

algorithm only makes one pass over the feature set, making a single decision on every sample, hence its name. Initially, the first sample is allocated to its own class. With the next sample, the distance to the the first class centre is computed. If this distance exceeds the given threshold, the new sample is allocated to its own class, else it is deemed to belong to the existing class. This process is iterated over all the samples. Clearly, n samples can lead to a worst-case scenario of $K = n$ after the algorithm converges, but this is unlikely to happen in practice, since the maximum cluster radius threshold is freely adjustable. This is often an exceptionally efficient algorithm in clustering highly separable clusters in a data set. Due to the lack of re-iterating, the algorithm is very fast. Quite interestingly, this simplified variant of K-means may be used as an initialisation mechanism, for it is very good at locating "obvious" clusters. As can be expected, it works perfectly for the pathological test feature sets described in Section 4.1.1.

Outlier Detection

The idea of a maximum allowable cluster radius is useful even in conventional K-means clustering. Outlier contamination of clusters is a practical problem for K-means; a small handful of outliers is enough to skew cluster means sufficiently to greatly distort the final result. In extreme cases, the presence of outliers may even mislead the algorithm into merging two distinct clusters. However, for the texture features under study, this problem is not too great. An outlier detection scheme can be based on the variance of feature space distances. This would mean augmenting the basic single K-means iteration with an extra processing step. After computing the cluster means (step 3), the variations of the intra-cluster distances (distances between mean and all member samples) are examined. Another pass of all samples is then performed, where all samples that are further than a given threshold from its cluster mean are classified as outliers, and are discarded from their respective cluster. This forces another recomputation of the cluster means, and will leave a portion of samples unclassified. Usually, the distance threshold is taken as one standard deviation greater than the cluster radius (average intra-cluster distance). For a quasi-Gaussian distribution, this would leave approximately 16% of the samples unclassified after every iteration. The reassignment of the unclassified samples to any particular cluster in the next iteration is not treated as a regular reassignment, that is, they are not considered in the termination condition, or else the algorithm would never converge. Theoretically, the outliers cannot be conclusively said to belong to any particular cluster,

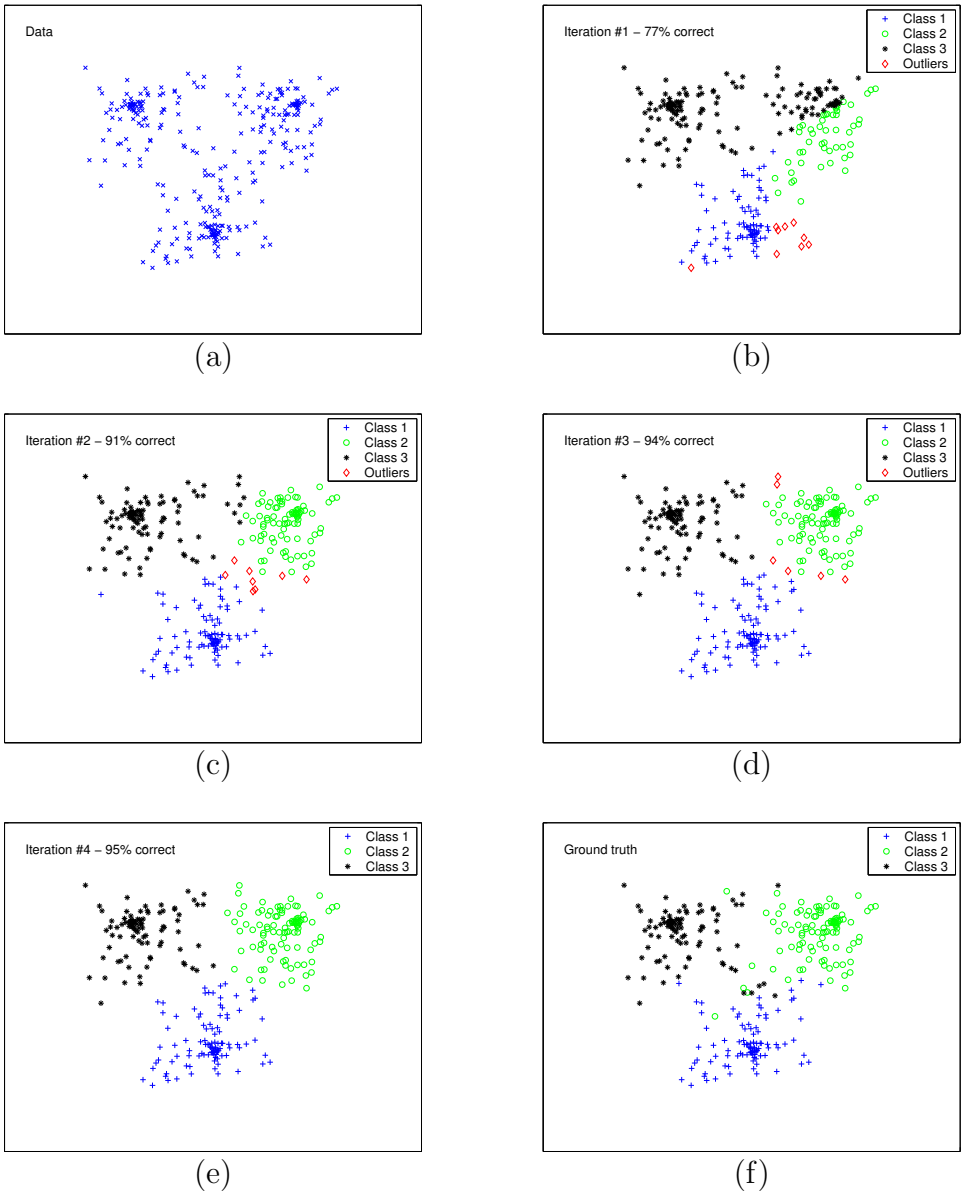


Figure 4.2. The K-means clustering algorithm with outlier detection. (a) Input data, 150 samples from 3 different distributions; (b)-(e) clustered data after 1 (b), 2 (c), 3 (d) and 4 (e) iterations; (f) Ground truth clustering.

and hence their reassignments does not warrant the same consideration as a direct cluster reassignment. Of course, the outliers can, and often do, become a integral part of a cluster in later iterations. An illustration of the outlier detection process is shown in figure 4.2.

This outlier detection mechanism allows the K-means algorithm to have more stability and robustness, and is especially useful in recovering from suspect initialisations.

Experiments revealed that this is indeed more stable; it produces very similar segmentations in the best case, but has much better results in non-best cases. Obviously, it does not produce miracles if other experimental parameters are inappropriate. The disadvantage of the K-means algorithm with outlier detection is the computation load. The need to compute cluster radii and variances, and the recomputation of cluster means greatly increases the execution time of these algorithms. The elimination of outliers also lead to more compact clusters (smaller radii), which is likely to increase the number of reassignments in every iteration, and in turn increases the number of iterations before convergence.

4.2 Fuzzy K-means Clustering

In the classic K-means algorithm, each sample is assumed to have a unique class membership, that is, it belongs to exactly one cluster. However, it is sometimes desirable to consider each sample to have a certain *probability* of belonging to each particular cluster. This is precisely the idea behind the Fuzzy K-means algorithm. Traditionally, this algorithm is more commonly known as fuzzy c-means in the literature. However, it is felt that consistency in terminology is more important here, considering its heavy dependence on the classic K-means. The extra flexibility of simultaneous multi-class membership of samples can lead to more robust segmentations. In particular, the fuzzy K-means avoids the problem of wrongful membership assignment. Such problems, especially during early iterations, can severely hamper segmentations produced by the classic K-means algorithm. In fact, this is one of the motivations behind the studying of various initialisation schemes for classic K-means.

The fuzzy K-means algorithm begins with the same set of n samples as in section 4.1. The steps are very similar to that of the classic K-means algorithm:

1. **Initialisation** The initial cluster membership probabilities, $P_{ij} = P(\mathbf{x} \in C_i|\mathbf{x}_j)$ are randomly initialised, subject to the usual constraints on probabilities:

$$\left.\begin{array}{l} \sum_{i=1}^{K} P_{ij} = 1 \\ 0 \leq P_{ij} \leq 1 \end{array}\right\} \quad 1 \leq j \leq n \tag{4.8}$$

2. **Compute class means** the cluster means are estimated by

$$\mu_i = \frac{\sum_{j=1}^{n} [P_{ij}]^b \mathbf{x}_j}{\sum_{j=1}^{n} [P_{ij}]^b} \tag{4.9}$$

where b is a free parameter which produces the desired fuzziness in the cluster membership;

3. **Update clusters** the cluster membership probabilities are updated by

$$P'_{ij} = \frac{d_{ij}^{\frac{1}{1-b}}}{\sum_{l=1}^{K} d_{lj}^{\frac{1}{1-b}}}, \quad \text{where} \quad d_{lj} = d(\mu_k, \mathbf{x}_j) \tag{4.10}$$

4. **Reiteration** Steps 2-3 are reiterated until the total change in the cluster membership probabilities are below a given threshold, ie.

$$\sum_{i,j} |P'_{ij} - P_{ij}| < \epsilon, \quad \epsilon > 0 \tag{4.11}$$

5. **Cluster Determination** each sample is assigned to the particular cluster that has the greatest membership probability.

$$P_{ij} = \begin{cases} 1, & \text{if } d_{ij} = \min\{d_{ik} \, \forall k\} \\ 0, & \text{otherwise} \end{cases} \tag{4.12}$$

This last step removes the fuzziness to produce the final clustering.

In practice, the value of ϵ greatly affects the rate of convergence of the algorithm. Obviously, a smaller value of ϵ would require more iterations before the algorithm terminates. A trick that may be applied is the hard assignment of clusters during each iteration. For this method, every sample's probability vector, $\{P_{ij} | 1 \leq i \leq K\}$, is checked after step 3; if the membership probability for cluster m exceeds a pre-determined value, say 0.95, then the sample is hard assigned to that particular cluster. That is, the probability vector is set to $P_{ij} = 1$ if $i = m$, 0 otherwise. This greatly improves the convergence speed of the algorithm, without sacrificing much performance in the segmentation results.

The fuzziness parameter b is a free parameter, subject to the constraint that its value must exceed unity. It governs the amount of "blending" allowed between the clusters, or the possibility of varying degrees of simultaneous memberships a sample can have. A large value of b allows a greater degree of fuzziness in every sample's membership, and

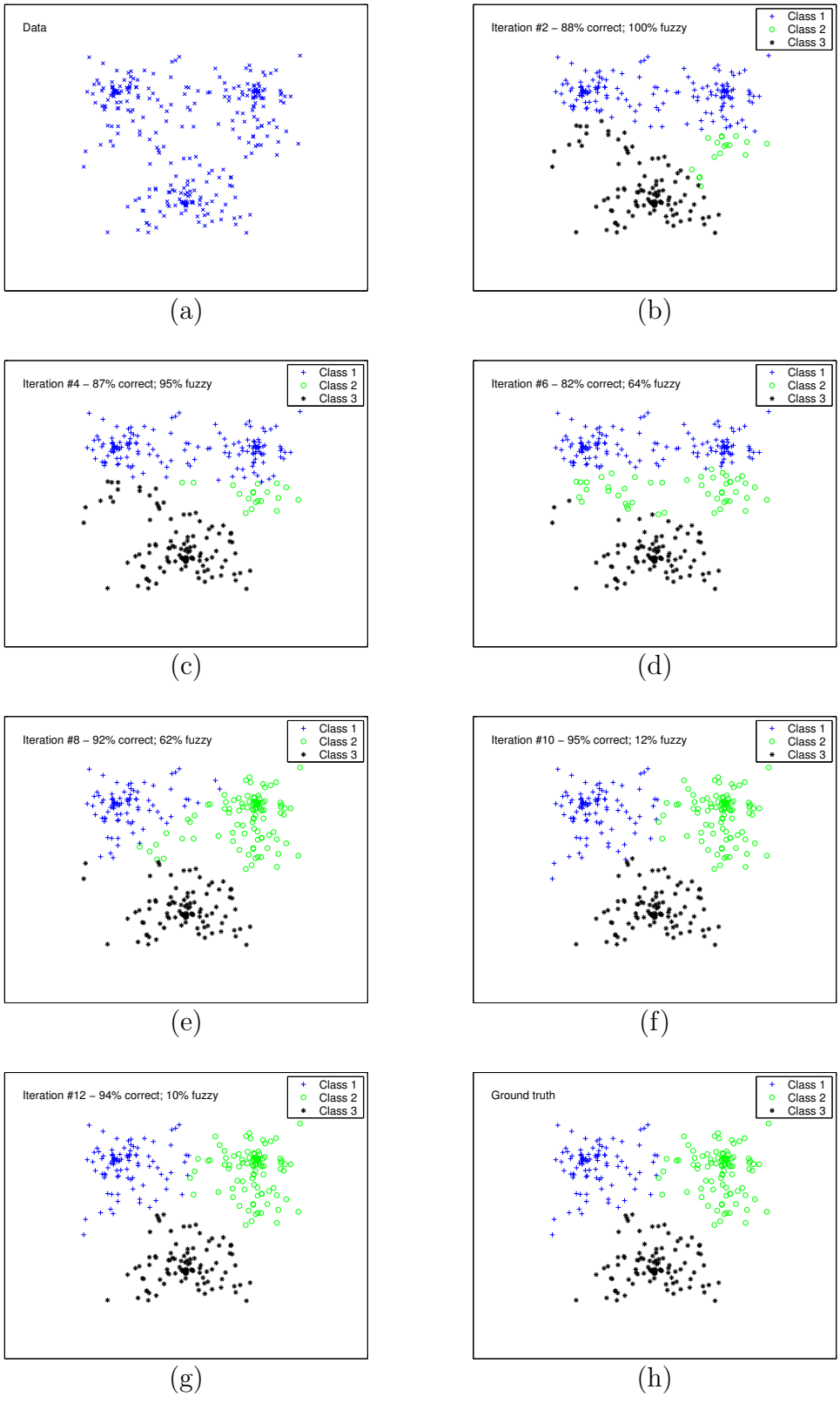


Figure 4.3. The Fuzzy K-means clustering algorithm in action for a 2-D dataset. (a) Input data, 150 samples from 3 different distributions; (b)-(g) clustered data after 2 (b), 4 (c), 6 (d), 8 (d), 10 (e), 6 (f) and 12 (g) iterations; (h) Ground truth clustering.

tends to require more iterations before the algorithm converges. As $b \to 1$, the fuzziness is reduced, and the algorithm approaches the simple K-means algorithm. Mathematically, as $b \to 1$, equation (4.10) approaches equation (4.12), which is the clustering decision rule for conventional K-means algorithm. For computational reasons, $b = 2$ yields the simplest implementations, and are used for most experimentations. It also produces the familiar Euclidean metric for measuring the distance between samples.

The complexity of fuzzy K-means is greater than conventional K-means, mostly from the computation of similarity functions during the membership probability updates. In the texture segmentation experiments, it was found that the fuzzy K-means algorithm is significantly slower than the K-means algorithm for the same test cases.

4.3 KLM Algorithm

The Koepfler-Lopez-Morel (KLM) algorithm [39] is an example of an agglomerative hierarchical clustering algorithm. The classical and fuzzy K-means algorithms are based on a flat representation of the feature samples. There is no real structure in this representation, where the samples are simply considered in a sequential, linear manner. In many real world data clustering applications, the data exhibit strong structure, where clusters may themselves be composed of smaller sub-clusters. These problems naturally lend themselves to a hierarchical solution, in which the data clustering is most effectively done in an embedded structure. Textures inherently exhibit significant internal structure. Indeed, purely random fields are usually perceived as noise, and not as textures. The particular feature extraction process described in chapter 3 focuses mainly on local neighbourhood information, and should therefore exhibit some degree of structure in the resultant feature vectors. Typically, a hierarchical representation also yields significant complexity reduction, and therefore has faster implementations.

Hierarchical clustering procedures can be divided into two categories: agglomerative and divisive. An agglomerative, or bottom-up, algorithm begins with many singleton clusters and achieves its result by successively merging clusters. This is in contrast with divisive (top-down) algorithms, which starts with a single, all-encompassing cluster, and forms the segmentation by successively splitting clusters.

The KLM algorithm is particularly attractive in that it only needs the user to provide one free parameter. It works by considering each sample as an individual cluster, then repeatedly merging these fine segmentations. The merging decision rules are derived from the minimisation of some specified global energy functional.

Generally speaking, a good segmentation should consist of a collection of homogeneous regions devoid of small holes in the interior, and separated by simple, smooth boundaries. Therefore, it is essential for the particular functional in the KLM algorithm to include separate terms that measure the self-similarity of regions and the relative complexity of the boundaries. Adjustable weights can then be introduced to control the relative emphasis between the dual (and often conflicting) aims of having highly homogeneous regions and simple, short boundaries. The particular functional used in the KLM algorithm is a bare minimum that satisfies these requirements. Assume that the image of feature vectors, $\mathbf{x}(x, y)$ is defined on an area Ω. Denote by $\mathcal{K}$ the whole boundary set, so the set of regions is written as $\Omega \backslash \mathcal{K}$. The functional is defined as

$$E(\mathcal{K}) = \int_{\Omega \backslash \mathcal{K}} \|\mathbf{u} - \mathbf{x}\|^2 \, \mathrm{d}x\mathrm{d}y + \lambda l(\mathcal{K}) \tag{4.13}$$

where $\mathbf{u}(x, y)$ is the mean feature vector of each cluster and $l(\mathcal{K})$ is the total length of boundaries. Generally, the $\| \cdot \|^2$ is a weighted distance measure on the difference vector $\mathbf{u} - \mathbf{x}$. The parameter λ is freely adjustable, and it directly affects the "fineness" of the final segmentation. A large λ leads to coarse segmentations with large regions, since boundaries are heavily penalised by the large λ. Conversely, small values of λ produce fine segmentations. An advantage of this algorithm is that it eliminates thin regions from the final segmentation implicitly, without the need of a separate size control parameter like many other clustering algorithms. Mumford and Shah have performed a complete mathematical analysis of the properties for this functional [50], that proves the properties discussed here.

The description of the algorithm is as follows:

1. Choose a value for λ. Construct the initial segmentation, K_0 for the image $\mathbf{g}(x, y)$. This takes every pixel to be a separate region; the region averages, $\mathbf{u}$, are then trivially the same as the pixel values $\mathbf{g}$ everywhere. The region tree data structure can now be built.

2. Compute the merging costs for each pair of neighbouring regions, $|O_i|, |O_j|$:

$$\Delta E = \frac{|O_i| \cdot |O_j|}{|O_i| + |O_j|} \cdot \|\mathbf{u_i} - \mathbf{u_j}\|^2 - \lambda \cdot l(\delta(O_i, O_j)) \tag{4.14}$$

   where $|\cdot|$ is the region area, and $l(\delta(O_i, O_j))$ is the common boundary between clusters i and j. Merge the pair with the lowest merging cost if it is negative; this ensures that each merging operation will decrease the overall functional value. Update the region tree after merging; all processing is confined to this data structure, and no computations are needed on the actual image.

3. Repeat step 2 until further merging is not possible, i.e. either there are no pairs with negative merging costs or only one region remains.

This algorithm is a slight departure from the one outlined in [39]. A problem with this algorithm is the selection of λ, the controlling parameter. In experiments performed, no assumption is made about λ; instead, a range of different values are iterated over sequentially. However, this would remove the nice pyramidal property of the KLM algorithm. In a supervised problem where the ground truth is known, it's a simple matter of comparing the truth with the final segmentations to evaluate their merit. From the experiments, it was found that the final energy functional (see equation (4.13)) of the segmented image provides a good indication of the quality of the final segmentation. More precisely, good segmentations are usually obtained near the minimum value of the functional when plotted as a function of λ. This is not surprising, since the KLM algorithm is based on the minimisation of the functional. For an unsupervised problem, this becomes a useful criterion for estimating the optimal value for λ. However, the functional can assume artificially low values when the algorithm fails completely, so it is still necessary to visually inspect the segmentation results.

Koepfler *et al* have applied their algorithm to several image segmentation examples. The results they obtained are rather impressive, particularly on the two-texture composite images, which are segmented with perfect accuracy. Interestingly, they were able to achieve this using simple energy features from a Haar wavelet frames decomposition, which is much simpler than the feature extractor described in this thesis. The great advantage of the KLM algorithm lies with its simple, optimisation theoretical basis.

From experiments, it was discovered that KLM algorithm suffers from a few problems. In its early stages, very small clusters are merged together. Due to their small sizes, these

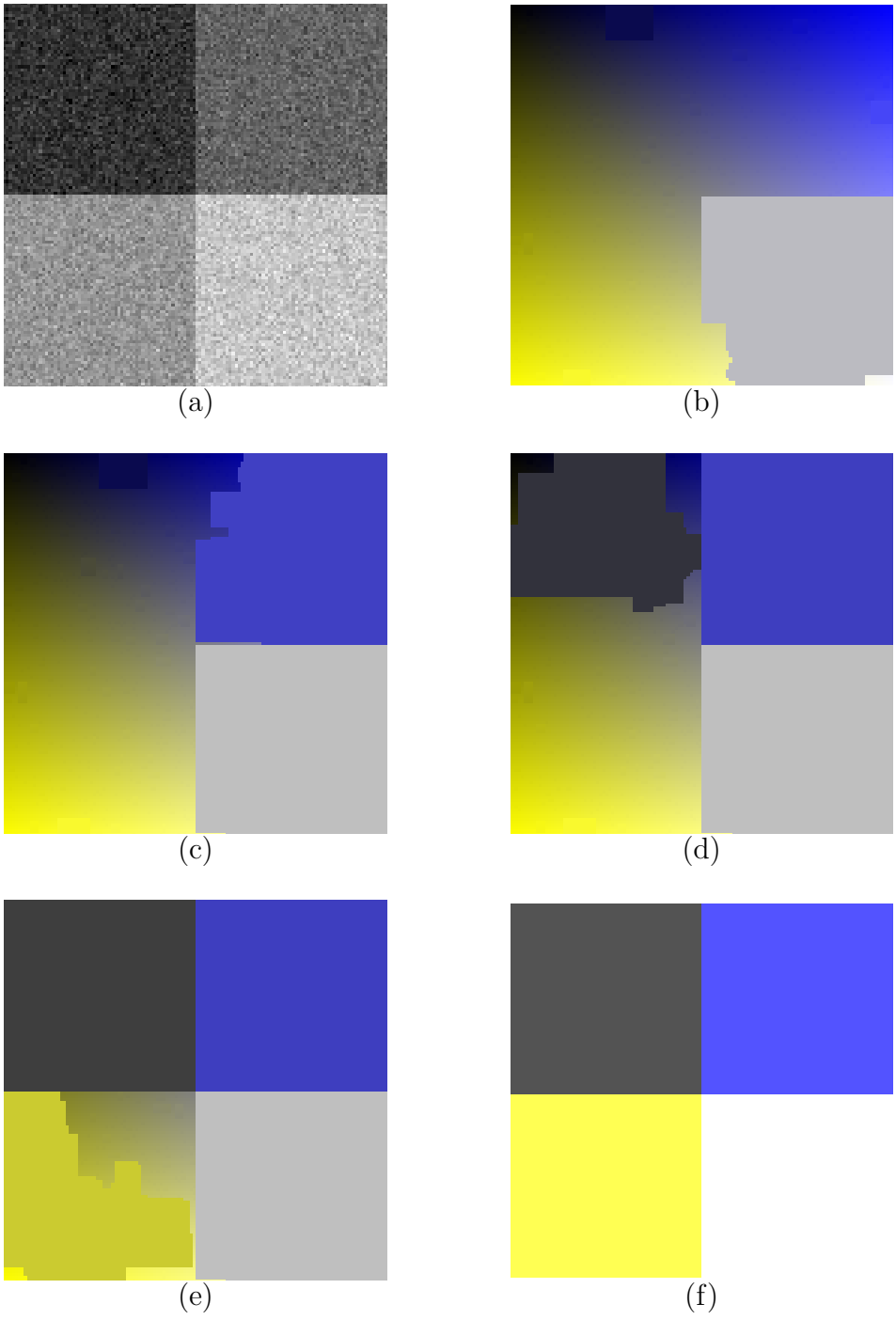


Figure 4.4. The KLM clustering algorithm in action for a 2-D dataset. (a) the input image; (b)-(e) intermediate clusterings and (f) the final segmentation. For this simple artificial example, the KLM is capable of producing a perfect result.

clusters do not contain all the texture information, but rather only a small subset of

that. This would increase the likelihood of incorrect merging, where two clusters from dissimilar textures are merged by mistake. Unfortunately, in the KLM algorithm, once two clusters are merged, they will not be separated again. Thus, any erroneous mergings would persist in the final result, causing high misclassification rates. Whatsmore, if a prematurely merged cluster contains more than one texture, it greatly increases the likelihood of further erroneous mergings. Another problem in applying KLM algorithm was noticed in experimentations. It was found that the large clusters were produced one by one. In the beginning, one large cluster tends to form while the numerous other tiny clusters remain as they are. It is until one cluster is completely grown before another major cluster grows. Ideally, all the major clusters grow simultaneously. The existence of a single large cluster makes the development of other major clusters more difficult, because the single cluster tends to dominate in the merging decisions. If the samples are well spread in feature space, then this may not be a major concern, but for texture features described in Chapter 3, the KLM algorithm had significant difficulties dealing with them.

A flaw of the KLM algorithm is its inability to accurately locate curved edges. This particular problem is due to the fact that the energy functional given in equation (4.13) penalises a diagonal boundary more than a horizontal or vertical one. Rather, this problem lies with the implementation of the functional, which divides an image into a collection of discrete pixels. For instance, a diagonal edge across a single pixel is allocated an edge length of 2 units instead of the actual geometric length of $\sqrt{2}$ units. This flaw will need to be addressed in real world applications where edges along arbitrary directions can exist. This difficulty can be solved by additional code that correctly compensates for diagonal edges, but this is likely to come with greatly added computational expense.

The KLM algorithm has difficulty with estimating the number of clusters. Unlike the K-means and fuzzy K-means algorithms, where the number of clusters must be provided, the KLM algorithm terminates when further merging does not yield a lower value of the functional. For different feature sets extracted from the same input image, it is not uncommon to have segmentations with different numbers of clusters. Obviously, this can be altered by specifying the desired final number of clusters, and force the algorithm to continue or stop merging operations until this value is obtained. However, this is likely to achieve a non-optimal value for $E(\mathcal{K})$.

The most computationally expensive step in the KLM algorithm is the cost calculation. This problem can be alleviated by saving computed costs in memory. In fact, the only costs that change during a merging operation are those that involve the two merged regions. A dynamic merge tree data structure enables this to be done very quickly. It was found that this simple optimisation yielded an order of magnitude improvement in code speed. However, even with this improvement, the high complexity in each merging step still makes the KLM algorithm much slower than the K-means algorithm.

4.4 Neural Networks

Neural network classifiers have been popular choices in texture problems. The motivation for studying neural networks came from the desire to mimic the operation of animal brains, which follow a different paradigm to digital computers. Animal neural systems are well adapted to process massive, parallel tasks, and are capable of learning and intuition. In contrast, digital computers follow the classic von Neumann paradigm, and are well suited to performing simple sequential tasks with stunning efficiency. In artificial intelligence studies, computers are asked to perform similar tasks handled by biological organisms, such as pattern recognition. Such efforts led to the development of a basic framework for neural computing. In this framework, biological systems are modelled as a collection of individually simple processing nodes, called *neurons*, with vast interconnects between them. There are many types of neural networks, but the neurons are nearly identical; the differences mainly lie in the architecture of the network (arrangement of nodes) and learning rule. A neuron is a weighted sum of its various inputs, followed by an *activation function*, a non-linearity that helps control the propagation of signals within the network.

Typically, the activation function is either a hard threshold or a sigmoidal non-linearity. The operation of a neuron is summarised as

$$y_i = f(\sum_{j=0}^{N} w_{ij} x_j) \tag{4.15}$$

Neurons are grouped together to form *neuron layers*; these layers are stacked on top of each other to produce a complete neural network, figure 4.6. There is an input and an output layer, and an arbitrary number of *hidden* layers sandwiched in between. Typically, the neurons in one hidden layer are connected to some or all of the neurons in

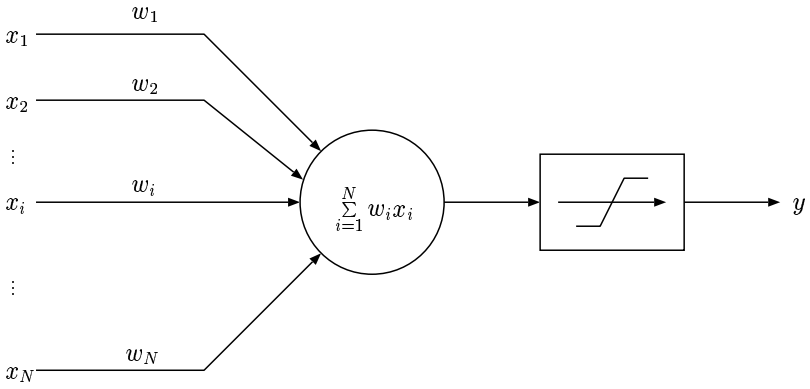


Figure 4.5. An artificial neuron. The input lines to the neuron model the action of dendrites in biological brains; weightings place different emphasis on the input stimuli, which is then subjected to the activation function. The output line models the axon.

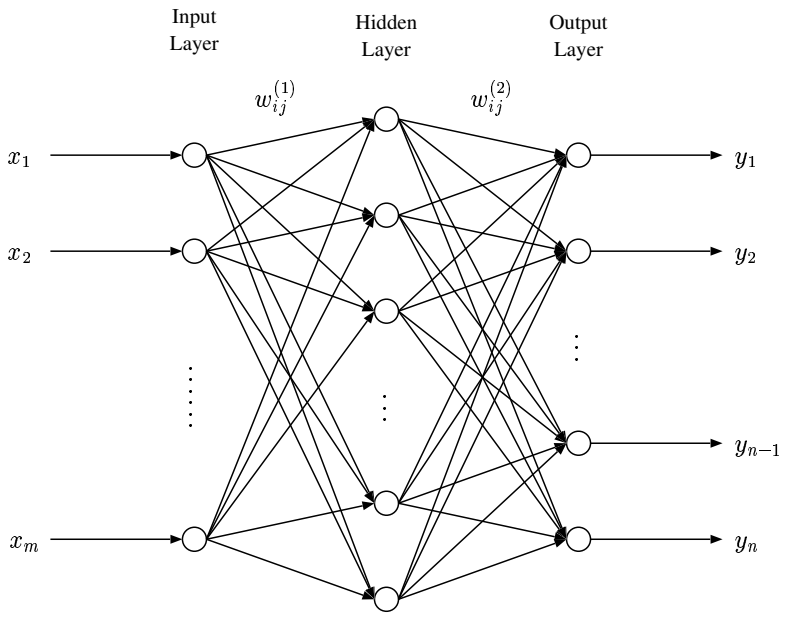


Figure 4.6. A multilayer perceptron with 3 layers, illustrating the multilayer neural network architecture. This particular network is a forward propagation network, using supervised, back-propagation learning.

the layers immediately above or below, although neurons in a layer may also be arranged in a matrix for general neural networks. Training of the network means determining the values of the weights, w_{ij}. This process can be accomplished in two manners: supervised or unsupervised. Most common is the supervised variety, which trains the network with

samples that have known desired outputs. In these cases, the quality of the training set is vital to the overall performance of the trained network on unseen data. Pattern association applications are examples where supervised training is useful. In contrast, unsupervised training on neural networks are used when the desired outputs on known samples are not available. An example of this type of training is self-organisation in Kohonen Feature Map neural networks. Unsupervised neural nets are, arguably, closer to real biological learning systems than supervised networks. They are more suited to pattern recognition type of applications, where the network needs to group together the input samples according to their similarity.

Neural networks have been applied, with great success, in a wide range of problems. However, choosing the appropriate architecture, training algorithm, and training set (for supervised networks) requires considerable skills and experience. There is a well-known difficulty with neural networks: a particular network trained with one set of data does not necessarily perform well in general. Every time a new problem is encountered, the neural network needs to be re-trained, or even re-designed, with a new test set. For texture segmentation, this can lead to considerable difficulties, since each input image is a new problem from the classification perspective. However, the value of neural networks may lie in the post-processing of segmented images. For example, they may be used to improve the segmentation, by training with samples from the centres of clusters, then re-classifying those from the edges.

4.5 Support Vector Machines

The device that is Support Vector Machines (SVMs) has received much attention recently in the machine learning community. SVMs are universal learning machines that are applicable in a wide range of pattern recognition problems. In particular, they solve these problems without requiring prior knowledge, thus giving them unparallelled generality. In fact, it has been shown that SVMs contain large sets of neural networks, radial basis function and polynomial classifiers as special cases. Despite the impressive resume, SVMs are quite simple devices, from an analytical perspective. Although SVMs are not used as the primary clustering tool in this thesis, their remarkable potential makes them a strong possibility in the future. This section provides a discussion of the theory behind SVMs.

4.5.1 Learning Machines

SVMs are based on structural risk minimisation from statistical learning theory. Traditionally, classifiers learn by training over a large number of sample data, known as the *training set*. Using notation common in the literature, this training set can be expressed as l sample-label pairs $\{\mathbf{x}_i, y_i, i = 1, \ldots, l\}$. Each vector $\mathbf{x}_i$ is a sample drawn from the system (distribution) to be learned, and y_i its known label (or class). The machine assumes the form of a "black-box" function, $f(\mathbf{x}, \alpha)$, which determines the label y for an input sample $\mathbf{x}$. The set of free parameters is denoted by the symbol α; these determine the architecture and details of the particular machine. The machine is then trained (i.e. determine α) by minimising the empirical mean error, or *risk*:

$$R_{emp}(\alpha) = \frac{1}{2l} \sum_{i=1}^{l} |f(\mathbf{x}_i, \alpha) - y_i| \tag{4.16}$$

where $|f(\mathbf{x}_i, \alpha) - y_i|$ is called the loss.

It is reasoned that, by mimicking the mappings found in the training set, the machine would be able to generalise to *any* sample drawn from the system. However, there is no guarantee that the trained machine will perform well on all possible samples. Indeed, much of the general performance largely depends on the size and quality of the training set. Another fundamental consideration in determining the performance of a learning machine is its *capacity*. To achieve optimal learning performance, the right balance must be achieved. A machine with too much capacity for the training data do not generalise well because the training is inadequate. This leads to performance degradation as the machine attempts to distinguish more details within the patterns than there really are. Conversely, a machine with too little capacity with respect to the training data is incapable of distinguishing all the patterns because it cannot handle the full complexity present in the problem. A crude real-world analogy would be the competence level of a person at a specific job. An incompetent person cannot complete the job satisfactorily, but an over-competent person may attempt to solve more of the problem than is required, which leads to inefficiency. Thus, a learning machine must have sufficient capacity to handle the problem that it has been asked to solve, and the correct amount of training must be applied to yield good general performance.

4.5.2 Capacity: the VC Dimension

The capacity can be seen as a measure of how fast a learning machine is able to converge to the true mean, the mathematical representation of the underlying process, as the number of training samples increases. In considering this convergence process, Vapnik derived an upper bound on the performance of any learning machine.

Before discussing the fundamental bound on a learning machine, it is necessary to first introduce the Vapnik Chervonenkis (VC) dimension. Consider a two-class classification problem; a learning machine $f(\alpha)$ produces the output ± 1, corresponding to the two classes for each input sample $\mathbf{x}$. For any set of l samples, there are 2^l possible permutations of labels. If a set of machines $\{f(\alpha)\}$ can generate all of these permutations, then the set of functions is said to *shatter* the set of samples. The VC dimension of a set of functions (i.e. a class of machines), $\{f(\alpha)\}$, is defined as the maximum number of samples that can be shattered by $\{f(\alpha)\}$. A simple illustration of this concept can be found in Burges' excellent tutorial on SVM [6]. One of the most important results for VC dimensions is the following:

> The VC dimension of the set of oriented hyper-planes in $\mathbf{R}^n$ is $n+1$, where an oriented hyper-plane is defined by the pair $\{H, \mathbf{n}\}$, where H is a set of points lying on the hyper-plane, and $\mathbf{n}$ is the unit normal vector.

This result means that a given set of points can be shattered by oriented hyper-planes, provided the dimensions of the hyper-planes are high enough. However, if a set of functions has a VC dimension n, it does *not* imply that it can shatter *any* set of n points. Thus, VC dimension is a measure of capacity, but not a guarantee on a machine's performance.

Assume a general process described by some probability density distribution $p(\mathbf{x}, y)$. That is, for a given sample vector $\mathbf{x}^{'}$, the probability of the output being $y^{'}$ is $P(\mathbf{x}^{'}, y^{'})\mathrm{d}\mathbf{x}\mathrm{d}y$. In training a learning machine $f(\mathbf{x}, \alpha)$ with the set $\{\mathbf{x}_i, y_i, i = 1, \ldots, l\}$, the objective is to minimise the risk as defined in equation (4.16) with respect to the parameter set α. Upon completion of the training, the machine has an expectation risk of

$$R(\alpha) = \frac{1}{2} \int |f(\mathbf{x}, \alpha) - y_i| p(\mathbf{x}, y)\, \mathrm{d}\mathbf{x}\mathrm{d}y \tag{4.17}$$

The bound on the expectation risk is given by

$$R(\alpha) \leq R_{\mathrm{emp}}(\alpha) + \sqrt{\frac{h(\log(2l/h) + 1) - \log(\eta/4)}{l}} \tag{4.18}$$

where $R_{\text{emp}}(\alpha)$ is the empirical risk measured on the training set and h is the VC dimension. The value $\eta, 0 \leq \eta \leq 1$ is chosen to represent the desired confidence interval of the machine. The most noticeable aspect of equation (4.18) is $R(\alpha)$'s independence of the probability density underlying the process. Thus, this bound depends solely on the capacity of the machine, for any given choice of η. The nature of the process does not affect the performance of the learning machine. In other words, for a given machine, we can be confident that it will be able to perform to a maximum risk. The VC dimension is a measure of the capacity of a learning machine, and is a property of a class of functions $\{f(\mathbf{x}, \alpha)\}$. Interestingly, the VC dimension does not always depend on the number of parameters, as it may appear so intuitively. Equation (4.18) does not tell anything about the performance of a given machine for a specific data set; it is merely a guide to the worst case scenario for the machine. In general, it is desirable to minimise the expected risk by choosing a machine with a low VC dimension. However, this will not always produce the optimal solution, as it has been known that some high h machines are capable of performing very well, even though the bound on the expected risk is very large. Indeed, there are well-known pathological cases where an infinite VC dimension machine performs very well.

4.5.3 Linear SVM

Perhaps the simplest case to illustrate the principle of an SVM is a linear machine trained on separable data. Assume the data in a two-class problem, $\{\mathbf{x}_i, y_i | 1 \leq i \leq l, y_i \in \{-1, 1\}, \mathbf{x} \in \mathbf{R}^n\}$, are separable. That is, a plane of the form

$$\mathbf{x} \cdot \mathbf{w} + b = 0 \tag{4.19}$$

is capable of separating samples from the two classes. This plane is called the *decision plane*. The linear, separable SVM is illustrated in figure 4.7. In other words, for any input sample $\mathbf{x}_i$, the decision function is

$$f(\mathbf{x}_i) = \text{sign}(\mathbf{x} \cdot \mathbf{w} + b) \tag{4.20}$$

The process of determination of the decision function is called the *training* of the linear SVM. Since the trained SVM will perfectly classify all samples in the current problem,

the samples of both classes are bounded by a pair of hyper-planes

$$\begin{aligned} \mathbf{x} \cdot \mathbf{w} + b \geq 1, \quad \text{if} \quad y_i = 1 \\ \mathbf{x} \cdot \mathbf{w} + b \leq -1, \quad \text{if} \quad y_i = -1 \end{aligned} \tag{4.21}$$

The distance between these planes is called the *margin*.

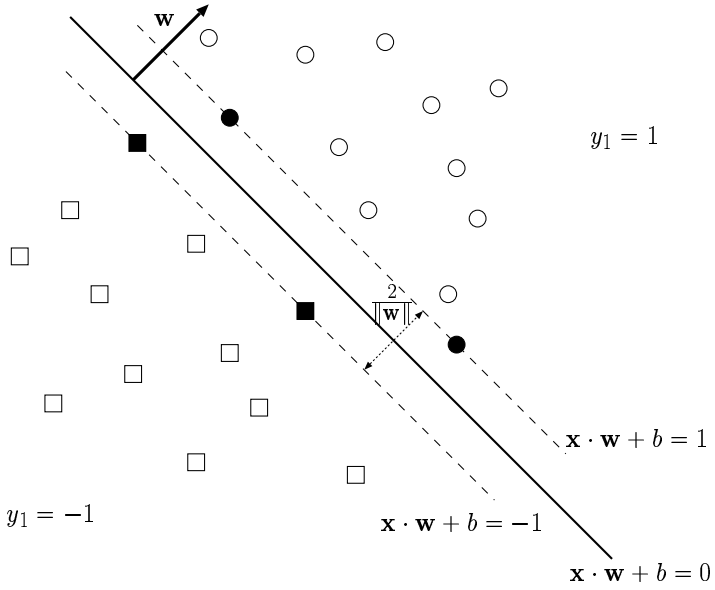


Figure 4.7. A linear, separable SVM. The decision surface is the solid line; support vectors are coloured solid, while other samples are hollow. From elementary geometrical considerations, the margin is shown to be $\frac{2}{\|\mathbf{w}\|}$.

The optimal decision plane is defined to be one that maximises margin between the two classes. By using a Lagrangian formulation [6], construction of the optimal hyper-plane reduces to a convex quadratic optimisation problem with linear constraints. These problems have been studied in mathematics for many years, and are well-understood. Many algorithms exist in the optimisation literature, and an exhaustive discussion of such methods is far beyond the scope of this work. Indeed, one of the great attractions of SVMs came with the discovery that optimal classification can be transformed into a quadratic programming (QP) problem, thus instantly allowing access to a great library of works. In summary, the training of an SVM is the same problem as the following:

Minimise

$$L = \frac{1}{2} \parallel \mathbf{w} \parallel^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i \tag{4.22}$$

with respect to $\mathbf{w}, b$*, subject to the constraints* $\frac{\partial L}{\partial \alpha_i} = 0, \alpha_i \geq 0$.

The Karush-Kuhn-Tucker (KKT) conditions are necessary and sufficient for solutions of a SVM [6]. These conditions state that all solutions for $\mathbf{w}, b, \alpha_i$ must satisfy

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \tag{4.23}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \tag{4.24}$$

$$\alpha_i (y_i (\mathbf{x}_i \cdot \mathbf{w}) - 1) = 0, \quad \forall i \tag{4.25}$$

The set of Lagrange multipliers, α_i, have to be determined numerically in practice. Very rarely do QP problems lend themselves to analytic solutions. Notice that there is a Lagrange multiplier for every training vector $\mathbf{x}_i$. In practical situations, most of these multipliers are zero, leaving only a subset of them contributing to the vector $\mathbf{w}$. Those data points whose multipliers do not vanish are called *support vectors*; non-support vectors may be removed from the training set without affecting the optimal solution. Interestingly, the use of the term "support" leads to a mechanical analogy of the process. In this interpretation, each support vector is considered to be exerting a force of magnitude $\alpha_i y_i$ on the decision boundary; the conditions in equations (4.23)-(4.25) are then simply the laws of mechanics governing a system in translational and rotational equilibrium.

The formulation described above requires a straight-forward modification for cases where the samples are not perfectly separable. Mathematically, it means the inclusion of additional slack variables, ξ_i, in the Lagrangian's formulation to handle the classification errors. The purpose of these variables is to penalise the misclassifications by increasing the overall objective function (the Lagrangian, L), through the relaxation of the margin constraints:

$$\begin{aligned} \mathbf{x} \cdot \mathbf{w} + b &\geq 1 - \xi_i, \quad \text{if} \quad y_i = 1 \\ \mathbf{x} \cdot \mathbf{w} + b &\leq -1 + \xi_i, \quad \text{if} \quad y_i = -1 \\ \xi_i &\geq 0 \qquad \forall i \end{aligned} \tag{4.26}$$

The rest of the formulation is the same as for the separable case; the modified Lagrangian is

$$L = \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i (y_i (\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i) - \sum_{i=1}^{l} \mu_i \xi_i \tag{4.27}$$

where C is a user-defined parameter to control the severity of misclassification penalty, μ_i are the extra Lagrange multipliers for the slack variables ξ_i. The conditions for solution are almost identical to the separable case (equations (4.23)-(4.25)), except for

$$\alpha_i(y_i(\mathbf{x}_i \cdot \mathbf{w}) - 1 + \xi_i) = 0, \quad \forall i \tag{4.28}$$

Figure 4.8 illustrates the operation of linear SVMs on non-separable data.

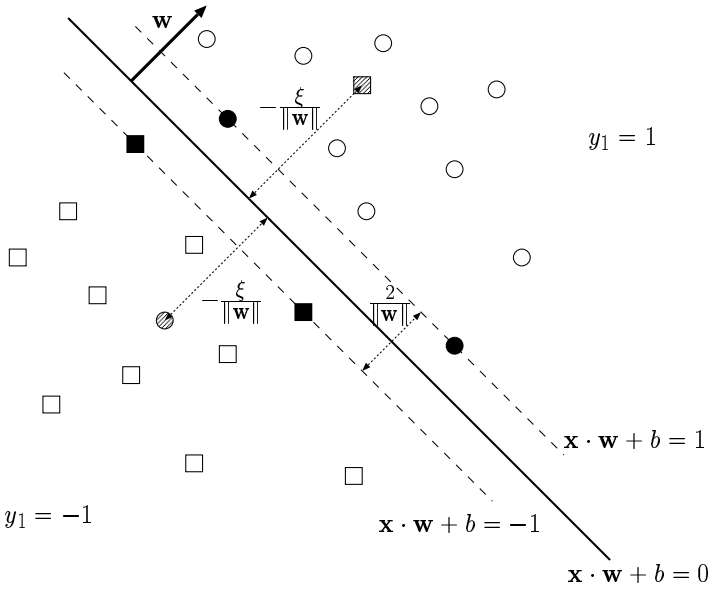


Figure 4.8. A linear, non-separable SVM. The decision surface is the solid line, and margin surfaces are the dashed lines. Support vectors are coloured solid, error samples are hashed while all other samples are hollow.

4.5.4 Non-Linear SVM

For most classification problems, the decision surfaces are non-linear. In order to extend SVMs to handle such surfaces, an old technique is used. The data are transformed to a higher, possibly infinite, dimension space. The idea is that the decision surfaces will be simple hyper-planes in the higher dimension, which then allows the use of techniques discussed above to solve the problem. Let such a transformation be $\Phi : \mathbf{R}^d \mapsto \mathcal{H}$, where $\mathcal{H}$ is the target Hilbert space. In general, it is very difficult to determine the appropriate mapping, Φ, for an arbitrary data set. The really important trick to make all this work is through a *kernel* function. In the formulation and solution of the QP problem in

section 4.5.3, it is discovered that the training data only ever appear in dot product form, $\mathbf{x}_i \cdot \mathbf{x}_j$. Explicit knowledge of the sample vectors is not necessary at any stage of SVM training and testing. Now, if there exists a function $K(\mathbf{x}_i, \mathbf{x}_j)$ such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \tag{4.29}$$

then it is possible to use exactly the same methods from linear SVMs. The solutions will assume the same form, and whatsmore, the training will have similar complexity as in the lower dimension. The result of the training will be an SVM that operates on $\mathcal{H}$, thus producing curved decision surfaces in $\mathbf{R}^d$. However, since little explicit knowledge of $\mathcal{H}$ is acquired, it is impossible to determine $\mathbf{w}$. However, as with the training, this knowledge is unnecessary in the testing, for the decision function will become

$$f(\mathbf{x}) = \sum_{i=1}^{N_S} \alpha_i y_i \Phi(\mathbf{x}) \cdot \Phi(\mathbf{s}_i) + b = \sum_{i=1}^{N_S} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + b \tag{4.30}$$

where N_S is the number of support vectors, $\mathbf{s}_i$. Therefore, the computation of $f(x)$ also requires knowledge of the kernel function only. Generally, a particular choice of kernel does not lead to a unique $\mathcal{H}$ or Φ. This is of tremendous advantage, because this allows one particular kernel function to solve a range of problems with different mappings and mapped spaces. Some of the common choices for the kernel function are

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p \tag{4.31}$$

$$K(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}} \tag{4.32}$$

$$K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} - \delta) \tag{4.33}$$

The existence of a suitable kernel function is crucial to non-linear SVMs. Mercer's condition provides the mathematical test for allowable kernel functions; a kernel $K(\mathbf{x}, \mathbf{y})$ satisfies equation (4.29) if

$$\int K(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) \, \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y} \geq 0 \quad \forall \, g(\mathbf{x}) \in L^2(\mathbf{R}^d) \tag{4.34}$$

4.5.5 Multi-class Problems

The basic formulation of SVM is based on the two-class classification problem, and so they are naturally binary classifiers. There have been several proposals to adapt or extend the

basic SVM to perform multi-class classification tasks. Collectively, these schemes can be broadly categorised as either combinational or true multi-class. The combinational approach basically divides a multiclass problem into many 2-class SVM problems, and then seek to fuse all the results together. On the other hand, true multiclass SVMs extend the formulation to consider all the classes at once. This approach often leads to very large optimisation problems, which make their solution very computationally intensive.

Some of the earliest multi-class SVM are based on the "one-against-all" method [31, 82]. For a k-class classification problem, k individual SVM classifiers are constructed, each testing for membership of one particular class. The final classification result is obtained from some fashion of voting among all the classifiers. Another popular approach is based on the construction of $\frac{k(k-1)}{2}$ "one-against-one" binary classifiers to separate each class from every other class. The outputs of these classifiers are again subjected to a voting mechanism to determine the final classification. Combinational multi-class SVMs have been widely applied in many applications.

True multiclass SVMs are constructed by generalising equation (4.27) to consider all the classes at once. Mathematically, this means summing all the contributions from all k classes in the Lagrangian to be minimised

$$L = \frac{1}{2}\sum_{m=1}^{k} \parallel \mathbf{w_m} \parallel^2 + C\sum_{i=1}^{l}\sum_{m\neq y_i} \xi_i^m - \sum_{i=1}^{l}\sum_{m\neq y_i} \alpha_i(y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i^m) - \sum_{i=1}^{l}\sum_{m\neq y_i} \mu_i \xi_i^m \tag{4.35}$$

The constraints are also modified to handle all the classes

$$(\mathbf{w_{y_i}} \cdot \mathbf{x_i}) + b_{y_i} \;\geq\; (\mathbf{w_m} \cdot \mathbf{x_i}) + b_m + 2 - \xi_i^m, \tag{4.36}$$

$$\xi_i^m \geq 0, \qquad i = 1, \cdots, l, \quad m \in \{1, \cdots, k\} \backslash y_i \tag{4.37}$$

with all the variables assuming their previous meaning. Unfortunately, the extended formulation for the multiclass SVM imposes a severe computational penalty on its solution. Compared to a 2-class SVM, a k-class problem increases the number of variables by a factor of $k-1$. Such hefty computational loads have limited the application of true multiclass SVMs. For large problems with many data vectors, such as texture segmentation, combinational methods are more practical [31]. However, more efficient optimisation techniques and faster computers in the future may make true multiclass SVM a practically feasible solution.

4.5.6 Concluding Remarks

SVMs are interesting and powerful learning machines, but they are a relatively recent discovery, which makes this a very active field of research. Plenty of successful applications of SVMs are already in existence, and they are generally thought to perform no worse than conventional classifiers such as neural networks. The strong performance of SVMs, from a conceptually simple formulation, is a pleasant surprise, although the reasons for this are not certain. A practical problem of deploying SVMs is the immense computational resources required for their training. As stated above, SVM training is a QP problem, which does not yield analytical solutions in general. Many algorithms exist for solving the QP problems, and they all offer various advantages for different data sets. Unfortunately, these algorithms do not scale efficiently, and the current state of computing technology means that SVM may not be feasible for some practical applications. For this work, SVMs have not been used as a classifier mainly due to computational concerns. In the future, new algorithms and advances in digital technology may become enabling factors that will allow more widespread use of SVMs in pattern recognition tasks.

Apart from numerical difficulties, another problem prohibits the SVM from use in the current context. From the outset, SVMs are designed to solve two-class problems. There are usually more than two textures in the general segmentation problem. Some extensions to SVMs for handling multiclass problems were briefly discussed, although these techniques are still in their relative infancy. The tremendous potential and elegance of SVMs make them a possible future direction for texture segmentation research.

4.6 Summary

The texture features obtained from the extraction process described in Chapter 3 must be processed by a clustering algorithm to produce a segmentation. Several known techniques for clustering data are presented in this chapter. Among them, K-means is the most heavily used algorithm in this thesis. Although simple in theory, it is effective and capable of producing good clusterings for a good feature set. The fuzzy K-means algorithm is a generalisation of the classic K-means, allowing simultaneous multi-class membership. Neural network classifiers have been heavily applied in pattern recognition problems. However, they require careful consideration in their training to be effective. Finally,

the relatively new support vector machines are discussed. Intrinsically, they are very powerful devices which can adapt themselves to suit the problem, and are therefore general learning machines. While this thesis mainly relies on the K-means algorithm to produce segmentations, it must be noted that neural networks and SVMs can be valuable for post-processing tasks. For example, the output of the K-means clustering can be used to select training samples for neural networks, and the trained network can reclassify the K-means segmentations to refine the results.

This page is blank

Chapter 5

Texture Segmentation Experiments

Following the discussions of wavelet-based texture feature extraction and clustering algorithms, this chapter contains the central results of this thesis. In order to examine the effectiveness of the proposed algorithms, an extensive set of segmentation experiments were performed. The work combines the different techniques described in previous chapters into a full segmentation system. Section 5.1 outlines the methodologies and scope of the experiments. In section 5.2, the properties of extracted wavelet texture features are examined using a variety of techniques. In particular, the separability of the extracted wavelet feature sets are investigated. Sections 5.3 and 5.4 present the results from segmentation experiments using conventional and fuzzy K-means clustering, respectively. In section 5.5, a modified K-means clustering algorithm is presented, which is specifically proposed for texture segmentation. This is followed by extensive results from experiments using the new method, and a discussion of the properties of this technique. Finally, the chapter concludes with a discussion of the results and a comparison with published results in the literature for similar experiments.

5.1 Data and Experimental Setup

5.1.1 Input Images and Methodology

The experiments described in this chapter are conducted on texture mosaics composed of textures from the Brodatz album [5], or are Brodatz-like in nature [14]. The vast majority

of the test cases are downloadable from the World Wide Web [14, 59]. Details and images of all input cases can be found in Appendix A. In choosing the input images, a conscious decision was made to select popular, readily obtainable mosaics which have been used by other researchers. This allows direct comparison of results in this thesis with algorithms already in the literature.

The ground truths of all input images are known in the experiments described in this chapter, and these are shown in Section A.1. This allows the computation of the number as incorrectly classified pixels by performing a pixel-by-pixel comparison between the segmentation result and the ground truth. This number serves as the one and only measure of performance in our experiments. More commonly, this measure is expressed as a percentage of *all* pixels in the image. It must be stressed that the sole purpose of the ground truths is for performance evaluation, not as a means for deriving training sets. All the segmentations are computed without prior training of the classifier(s).

The aim of the experiments is to examine the performance of the proposed algorithm through large numbers of experiments. It is believed that the true measure of the usefulness of a texture segmentation algorithm can only be established through empirical evidence. There are a total of 115 mosaics listed in Appendix A, which makes this one of the most exhaustive studies among similar efforts.

5.1.2 Feature Parameters

Chapters 3 and 4 detailed the feature extraction and clustering procedures. The complicated nature of these processes means that a number of parameters must be supplied to the algorithm to completely define the operations that make up the whole process. A conscious attempt has been made to minimise the number of parameters required for the functioning of our texture segmentation system. Ideally, the texture segmentation system should be able to adapt itself to the input image and automatically estimate appropriate values for the internal parameters. However, this is a lofty goal, and is rarely achieved in any pattern recognition problem. In the current algorithm, the user needs to specify the following parameters for feature extraction and conditioning:

- Wavelet transform type and depth
- Choice of smoothing window

- Numerical non-linearity

In this thesis, experiments are performed for a range of values for these parameters, in the interest of comparison. From a feasibility standpoint, it is necessary to limit the range of parameters for the experiments to keep the number of experiments manageable with limited computational resource. The choices for these ranges will be discussed and justified in turn.

The wavelet transform is the primary analysis tool for extracting texture features. Following discussions in Chapter 2, two different wavelet transform types are experimented with: conventional DWT and the newer DT-CxWT. Gabor transform, while abundant in the literature, is not part of the focus of this work. However, results from Gabor filter-based schemes will be used as a yardstick for comparison purposes. Conventional DWT is desirable from an algorithmic efficiency standpoint, because it is a non-redundant representation. However, the approximate shift-invariance and superior directionality properties of DT-CxWT make it a more appealing choice. The direct comparison between the two types of transforms is a major focus of this thesis. The most obvious parameter in the wavelet transform is the analysis filter set. Linear-phase filters are employed in this thesis; these are important in image processing applications. The Daubechies 9-7 pair and the Kingsbury pairs are chosen for the DWT and DT-CxWT, respectively. Certainly, many other DWT filters exist, but they do not offer vastly different properties; for example, all DWT filters do not allow shift-invariance. The differences between the various DWT filters are not entirely relevant in the current context. Arguably, a more important wavelet transform parameter is the transform depth. In the literature, examples of applications using more than 5 levels are rare. For texture segmentation, such a deep transform is very undesirable for several reasons. Firstly, the lowest resolution (deepest) subbands are highly downsampled ($2^d : 1$), and features at such scales would cause considerable blurring of texture boundaries. Secondly, a deep transform produces many subbands in the decomposition ($L = 3d + 1$ for a tree-structured transform), and hence feature vectors with high dimensions. This greatly increases the volume of data in the process, and therefore the computational effort required for each segmentation. Moreover, high feature dimensionality can lead to problems with the classification (Chapter 4). On the other hand, it is well-known that most textural information are located in the mid-frequency subbands [9], which means that depths of at least 2 are necessary to effectively capture

these mid-frequencies in separate subbands. As a result, transform depths of 2 and 3 are chosen for the experiments in this thesis.

As discussed in Chapter 3, the role of smoothing is vital in conditioning texture features for clustering. For the experiments, two types of smoothing are considered: low-pass and median filtering. Low-pass filtering smooths features by calculating the weighted average over a small neighbourhood using a normalised mask. Median filtering computes the median feature value over a spatial neighbourhood. Generally speaking, median filters have the advantage of being insensitive to outliers, but whether this property assists texture feature extraction remains to be seen. Median filters of sizes 3×3, 5×5 and 7×7 are used in our experiments; very large median filters lead to extreme blurring. Smoothing with low-pass filters is performed using masks constructed from 1-D Kaiser windows in the experiments. As discussed in Section 3.4.3, the Kaiser family of windows has better time-frequency characteristics than Gabor. The choice of window size is a trade-off between texture smoothing and boundary localisation - better smoothing also blurs boundaries. It has been determined that, for the size of the input images, a window size of 11×11 pixels is adequate. The β parameter for the Kaiser windows (equation 3.15) is varied for a range of different values, which are indicated in the various result tables. Changing β has the effect of varying the spatial weighting of the windows — increasing β narrows the width of the window peak (see figures 3.11(a) and 3.11(b)). For these experiments, the values of β are chosen to be 0, 1, 2, 3, 5, 8, 13, 21, 34 and 55. These values give a good range of different peak widths (see figure 5.1).

As discussed in section 3.4.4, a sigmoidal non-linearity (equation 3.16) is used to condition the smoothed features. This step is shown to improve the numerical conditioning of the texture features. This is the least crucial of the feature conditioning steps; it has been found that the value of α in equation 3.16 does not have a very big effect on the overall segmentations. In light of this, a fixed value of $\alpha = 0.25$ is used in the experiments.

5.1.3 Clustering Parameters

For clustering, the parameter set depends on the choice of algorithm. For example, all variants of K-means (including fuzzy) requires the user to supply the number of clusters, K and neural network classifiers need a training set derived from the extracted features. In principle, support vector machines are universal learning algorithms, and should not

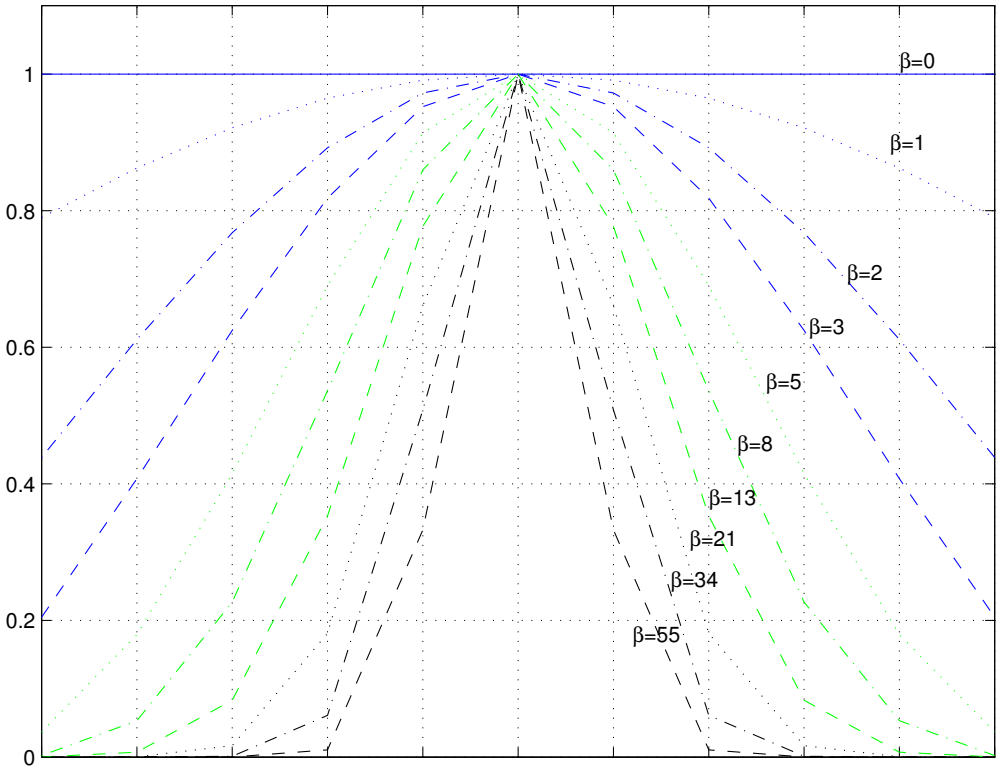


Figure 5.1. Plots of ten 11-tap Kaiser windows used in the experiments. Shown are the normalised 1-D windows; 2-D masks are obtained by taking outer products of the 1-D windows.

require any extra input parameters. However, in practice, a number of parameters are needed for the numerical algorithms used to solve the quadratic programming problem.

The vast majority of the results in this thesis are obtained from K-means clustering, so the choice of K-means parameters is described in detail. Firstly, the correct values of K are fed into the algorithm for all experimental cases, since all the ground truths are known. In practical cases where the ground truths may not be known, it would be necessary to find estimates for the true number of clusters. A vital element of the K-means algorithm is the class updating process in each iteration. In this thesis, a static update scheme is employed in all experiments involving K-means clustering. In our experience, dynamic updating of class means has exhibited tremendously high sensitivity to the precise state of the initialisation — two different initialisations can lead to dramatically different results. While this phenomenon is also observed for static updating in some instances, the effects are far less pronounced. The reason for this is due to the rapid change of class means during the first few iterations of the K-means algorithm, when many pixels switch labels.

A poor initialisation coupled with dynamic updating will result in a large number of erroneous labelings after the first iteration, which will then lead to very poor segmentations. Obviously, this instability is undesirable, and for this reason, static updating is preferred in the experiments.

The initialisation method deserves careful consideration, since the K-means algorithm is completely deterministic once the initial state is specified. In the experiments, the cyclic labeling method, described in Section 4.1.1, is used. Ideally, many iterations with a random initialisation scheme should be used to avoid the danger of producing locally optimal clusterings, but experiments have shown that cyclic initialisation provides generally good results without the need to run multiple iterations of the algorithm, thereby saving computational time by at least an order of magnitude. Table 5.1 shows the performance comparison between random and cyclic initialisation schemes over a set of 5-texture mosaics. The particular parameter values are detailed in the accompanied caption. While the comparison is performed for one combination of parameters over a subset of all input images, it does, however, adequately illustrate the overall trend. Due to the extraordinary computational requirements of running many randomly initialised iterations, it is infeasible to exhaustively compare the two initialisation schemes. It is observed that cyclic initialisation does not generally obtain an error rate as low as the minimum found by multiple runs with different random initialisations. However, the results are very close, often within a fraction of a percent in the error rate. Whatsmore, cyclic initialisation is able to produce better results than the median of a large number of repeated random runs. The only exception to this rule is for the input image "My 5a" (see Appendix A), where the cyclic initialisation actually produces a lower error rate than the minimum found from the random initialisations. There certainly exist cases where the cyclic labeling proves inappropriate, leading to unusually poor segmentations (a local optimum); section 4.1.1 has already discussed this problem in more detail. In general, though, it has been observed that the cyclic labeling scheme often gives good segmentation results when compared to random initialisations repeated 10 times or more. The reliability of the cyclic initialisation scheme found in our experiments may be due to the nature of our data set, where the cluster areas do not differ greatly from image to image. For such data, it is reasonable to expect the K-means algorithm to benefit from an initialisation scheme that produces a balanced initial population. However, the cyclic

initialisation scheme also assumes no information about the dataset; the initial cluster centres are nearly identical to each other, since the cluster memberships are equally scattered across the entire image. The fact that K-means is able to achieve good segmentations for such a simple initialisation scheme is an endorsement of the feature sets.

Table 5.1. Comparison between random and cyclic initialisation schemes. The mean segmentation error rates (%) are shown. Feature extraction parameters: DT-CxWT with Kingsbury filter pairs, depth 3, Kaiser $\beta = 3$. The random results are obtains from 100 runs.

Image	Cyclic (%)	Random Min (%)	Random Max (%)	Random Median (%)
My 5a	14.91	17.68	42.18	18.34
My 5b	2.87	2.87	4.14	3.42
Nat 5b	2.12	2.09	32.97	2.10
Nat 5c	2.66	2.66	23.80	2.66
Nat 5m	2.98	2.96	33.92	2.96
Nat 5v2	4.25	4.24	39.92	4.38
Nat 5v3	5.19	5.19	37.92	5.21
Nat 5v	3.53	3.48	39.95	3.55

Table 5.2. Comparison between Euclidean and Manhattan metrics. Error rates (%) for a set of 5-texture mosaic's segmentation results obtained from both metrics are shown in the box. Feature extraction parameters: $\beta = 3$, Kingsbury DT-CxWT filter pairs, transform depth 3. The mean error rate of the L^1 metric is 3.40, with a 95% confidence interval of (2.42, 4.37); mean error rate of L^2 metric is 3.76%, with the corresponding confidence interval of (2.34, 5.18).

Image	L^1 (%)	L^2 (%)	Image	L^1 (%)	L^2 (%)
My 5a	4.66	6.59	Nat 5m	2.97	3.00
My 5b	2.14	1.82	Nat 5v2	4.25	5.17
Nat 5b	2.12	2.31	Nat 5v3	5.11	5.19
Nat 5c	2.46	2.52	Nat 5v	3.44	3.49

The choice between Euclidean and Manhattan distance measures in the K-means clustering does not have a demonstrable effect on the segmentation performance. Table 5.2 shows the segmentation performance for a set of 5-texture mosaics. The differences in error rates between the use of L^1- and L^2-norms are small in most instances, but they favour L^1 in all cases except one. The mean error rates are 3.40% and 3.76% for the

Manhattan and Euclidean distance metrics, respectively. Due to the small sample size ($N = 8$ images) in this comparison, the confidence intervals for the two means are quite large - (2.42, 4.37) for the Manhattan metric and (2.34, 5.18) for Euclidean. Although the error rate difference between the two metrics is small, the smaller confidence interval of the Manhattan distance makes it slightly better than the Euclidean distance. Therefore, the Manhattan distance is selected for the experiments. In addition, the calculation of L^1-norms is usually more efficient than L^2-norms on general purpose digital hardware for our feature values, which are represented as double-precision floating point numbers. However, this advantage is offset by the need for slightly more iterations of the algorithm in the case of the L^1-norm, a trend noticed from our experiments. Considering both algorithmic and segmentation performances, there does not appear to be any compelling reasons to choose one metric over the other, but the Manhattan distance is slightly better.

In summary, a cyclic initialised K-means clustering algorithm with static update is used for all experiments. This provides the most desirable combination of computational and performance characteristics among the different variants of K-means discussed.

5.2 Feature Properties

The details of the feature extraction process have been described in Chapter 3 and experimental details are disclosed in the previous section. In this section, the properties of the extracted features in the experiments are examined. Chiefly, the focus is on the *separability* of the features; this is very important if the features are to be clustered to produce the final segmentation. The purpose of examining the separability of feature sets is to gain a better picture of these sets, which would give a general indication of the characteristics of the feature extraction process. There are numerous separability measures for feature sets, and it is obviously infeasible to examine them all. In this section, three different techniques are employed: spatial separation, feature contrast and distance histogram. While each of these techniques has its own advantages and disadvantages, it is hoped that, by using all three techniques, it is possible to gain a better appreciation of the separability of the extracted features.

5.2.1 Spatial Separation Criterion

The spatial separation criterion was discussed in section 3.4.1. It was described as a method to reduce feature dimensionality. However, it can also be used to gauge the discrimination power of a particular feature set. To recap, the spatial separation criterion is defined as the ratio of average intra-cluster to inter-cluster distances. A small value tends to illustrate a separable feature set; a value much lower than 1 would be desirable. To examine the spatial separability of feature sets, the criterion is calculated for all the feature sets extracted in our experiments. The calculations are simple, as the ground truths for all input images are known. The results are summarised in table 5.3. It can be seen that the spatial separability criterion is significantly less than unity for all cases. In fact, most cases yield a spatial separability ratio below 0.5, which is a healthy sign, with regard to clustering. However, this does not prove that the feature sets can be successfully clustered to give accurate segmentations; it is merely an indication that the classes may possess good separability. Ultimately, the feature sets must be evaluated by the final segmentation results produced.

Perhaps a better method to gauge feature separability is to compare the intra-cluster distance of each class with its inter-cluster distances from the other clusters. That is, to compare the intra-cluster distance of cluster i (d_{ii}) with the average distances $d_{ij}, i \neq j$ from the other clusters. The reasoning for this is, if a class has its intra-cluster distance smaller than the average inter-cluster distances to all other clusters ($d_{ii} < d_{ij} \forall j \neq i$), then it is reasonable to assume that class can be separated from the others, using a distance-based clustering algorithm. In the experiments, features were extracted for 115 input images, using 10 different values for β. It was found that 98.4% of the 1150 extracted feature sets using 3-level DT-CxWTs satisfy this separability criterion. For 3-level DWT features, a slightly lower 97.2% of feature sets satisfy this criterion. These figures show that the vast majority of the features extracted from our process have good potential to produce quality segmentations.

From the results of this section, it can be seen that the spatial separability of both DWT and DT-CxWT features (3-level transforms) is good. The DWT features give slightly lower separation ratios than DT-CxWT in most cases. However, the differences in all cases are very small, often within 0.02. This evidence is not sufficient to establish the superiority of one transform over the other, since the SSR does not directly correspond to

Table 5.3. Spatial Separability ratios (SSR) for DWT and DT-CxWT features. Feature extraction parameters: Kingsbury filters, depth 3, Kaiser smoothing windows. Manhattan distance is used as the metric.

Image	DWT	DT-CxWT	Image	DWT	DT-CxWT	Image	DWT	DT-CxWT
D4-D84	0.69	0.56	bonn 24	0.38	0.38	bonn 63	0.46	0.46
D5-D92	0.60	0.64	bonn 25	0.40	0.42	bonn 64	0.47	0.47
D8-D84	0.30	0.31	bonn 26	0.37	0.38	bonn 65	0.32	0.32
D12-D17	0.43	0.46	bonn 27	0.35	0.38	bonn 66	0.36	0.38
My 5a	0.41	0.43	bonn 28	0.38	0.38	bonn 67	0.39	0.41
My 5b	0.40	0.43	bonn 29	0.32	0.34	bonn 68	0.39	0.41
Nat 10	0.53	0.53	bonn 30	0.29	0.28	bonn 69	0.53	0.53
Nat 10v	0.52	0.54	bonn 31	0.38	0.41	bonn 70	0.44	0.48
Nat 16b	0.50	0.52	bonn 32	0.48	0.51	bonn 71	0.34	0.36
Nat 5b	0.55	0.52	bonn 33	0.33	0.34	bonn 72	0.31	0.35
Nat 5c	0.50	0.47	bonn 34	0.42	0.45	bonn 73	0.50	0.49
Nat 5m	0.51	0.52	bonn 35	0.31	0.31	bonn 74	0.44	0.45
Nat 5v2	0.59	0.58	bonn 36	0.41	0.41	bonn 75	0.40	0.42
Nat 5v3	0.65	0.66	bonn 37	0.38	0.40	bonn 76	0.40	0.41
Nat 5v	0.51	0.55	bonn 38	0.45	0.47	bonn 77	0.41	0.42
bonn 00	0.37	0.38	bonn 39	0.38	0.37	bonn 78	0.29	0.30
bonn 01	0.57	0.57	bonn 40	0.31	0.31	bonn 79	0.48	0.51
bonn 02	0.47	0.49	bonn 41	0.50	0.51	bonn 80	0.30	0.32
bonn 03	0.38	0.38	bonn 42	0.34	0.35	bonn 81	0.52	0.51
bonn 04	0.33	0.33	bonn 43	0.36	0.37	bonn 82	0.36	0.38
bonn 05	0.35	0.37	bonn 44	0.44	0.44	bonn 83	0.36	0.36
bonn 06	0.42	0.46	bonn 45	0.45	0.45	bonn 84	0.37	0.37
bonn 07	0.30	0.31	bonn 46	0.42	0.43	bonn 85	0.47	0.49
bonn 08	0.72	0.72	bonn 47	0.43	0.45	bonn 86	0.37	0.40
bonn 09	0.37	0.37	bonn 48	0.39	0.39	bonn 87	0.42	0.41
bonn 10	0.42	0.46	bonn 49	0.37	0.37	bonn 88	0.34	0.35
bonn 11	0.39	0.41	bonn 50	0.42	0.44	bonn 89	0.40	0.40
bonn 12	0.38	0.39	bonn 51	0.34	0.35	bonn 90	0.38	0.39
bonn 13	0.44	0.45	bonn 52	0.36	0.38	bonn 91	0.40	0.43
bonn 14	0.34	0.34	bonn 53	0.30	0.32	bonn 92	0.37	0.38
bonn 15	0.39	0.42	bonn 54	0.37	0.39	bonn 93	0.50	0.54
bonn 16	0.35	0.37	bonn 55	0.42	0.44	bonn 94	0.45	0.48
bonn 17	0.34	0.35	bonn 56	0.45	0.44	bonn 95	0.45	0.47
bonn 18	0.44	0.47	bonn 57	0.51	0.52	bonn 96	0.35	0.38
bonn 19	0.35	0.35	bonn 58	0.39	0.39	bonn 97	0.35	0.36
bonn 20	0.33	0.35	bonn 59	0.37	0.39	bonn 98	0.43	0.44
bonn 21	0.40	0.41	bonn 60	0.38	0.41	bonn 99	0.38	0.38
bonn 22	0.49	0.50	bonn 61	0.37	0.40			
bonn 23	0.30	0.33	bonn 62	0.49	0.50			

the segmentation results from feature sets. The only conclusion that can be drawn from the SSR results is that both DWT and DT-CxWT are capable of producing feature sets that are potentially separable by a distance-based classifier.

5.2.2 Feature Contrast

Feature contrast was discussed in section 3.4.1 as a means to measure the separability of a feature set. Equation 3.12 gives the definition of contrast. In order to examine the separability of the features, each extracted set has its contrast measured as follows: for

every combination of input image (115 in total) and β parameter (10 different values), the contrasts for all feature components are calculated; the minimum and mean over all components are recorded. The results are shown in table 5.4. The contrast ratio never exceeds unity if all the feature values are non-negative, as they are in this context. A large value of the contrast ratio shows greater contrast. For example, for a perfect 2-class, single feature problem (i.e. the features are scalars and have value 0 or 1) with equal population, the contrast would be $\frac{1}{3}$. However, the nature of this measure is very sensitive to outliers; the presence of a small number of large values in the feature components would artificially inflate this number. The smoothing process in feature extraction alleviates this problem significantly. Another problem with the contrast measure is that it is difficult to ascertain its meaning for a multi-class problem. However, the contrast ratio is extremely simple to compute, and hence demands very little computation resources. It provides a quick way to gauge the separability of a feature set.

From the tables, it is clear that the contrasts in all feature sets are quite good. The average contrast for all cases is 0.41, while the mean minimum contrast is 0.25. In particular, for the two-class problem where the contrast ratio has the most meaning, the values are nearly $\frac{1}{3}$. As a general trend, DT-CxWT features have higher average contrast ratios than DWT features. The contrast ratio measure favours the DT-CxWT features, unlike the SSR in section 5.2.1. However, it must be stressed once again that values of contrast ratio do not have a direct implication on the segmentation results of an image.

5.2.3 Distance Histogram

While the two previous methods in this section produce numerical measures of separability, the distance histogram method provides a visual cue to the separability of a feature set. This method has been shown to be a useful tool in determining the separability of a feature set [24]. For a separable feature set, the average intra-cluster distance is usually very different to the average inter-cluster distances. When the distances in feature space are plotted as a histogram, there should be distinct peaks corresponding to the inter- and intra-cluster distances. Thus, the existence of distinct peaks in feature space distance histogram provides visual evidence of feature set separability. This phenomenon should be most pronounced for a two-class feature set, because of fewer possible class combinations for any pair of feature vectors. To be more precise, there are only three different pairings

Table 5.4. Feature contrasts for DWT and DT-CxWT features. Feature extraction parameters: Daubechies 9-7 pair and Kingsbury DT-CxWT filter pairs, depth 3, Kaiser smoothing window. Listed are the average and minimum (best) contrast ratios obtained for each image, from all values of the Kaiser parameter, β.

	DWT		DT-CxWT			DWT		DT-CxWT	
Image	Ave	Min	Ave	Min	Image	Ave	Min	Ave	Min
D4-D84	0.26	0.20	0.30	0.17	bonn 20	0.37	0.24	0.39	0.25
D5-D92	0.26	0.20	0.30	0.17	bonn 21	0.37	0.26	0.39	0.26
D8-D84	0.28	0.21	0.31	0.16	bonn 22	0.38	0.26	0.40	0.27
D12-D17	0.30	0.22	0.33	0.18	bonn 23	0.40	0.28	0.42	0.28
My 5a	0.34	0.25	0.37	0.24	bonn 24	0.43	0.31	0.46	0.30
My 5b	0.33	0.27	0.35	0.27	bonn 25	0.39	0.23	0.40	0.23
Nat 10	0.34	0.27	0.35	0.27	bonn 26	0.39	0.22	0.41	0.23
Nat 10v	0.35	0.27	0.37	0.28	bonn 27	0.40	0.22	0.41	0.24
Nat 16b	0.37	0.29	0.40	0.32	bonn 28	0.41	0.24	0.43	0.26
Nat 5b	0.41	0.33	0.44	0.33	bonn 29	0.44	0.27	0.47	0.30
Nat 5c	0.30	0.22	0.30	0.21	bonn 30	0.36	0.25	0.36	0.22
Nat 5m	0.30	0.22	0.30	0.21	bonn 31	0.36	0.26	0.37	0.22
Nat 5v2	0.31	0.22	0.32	0.21	bonn 32	0.37	0.27	0.38	0.23
Nat 5v3	0.34	0.24	0.34	0.23	bonn 33	0.39	0.28	0.39	0.25
Nat 5v	0.38	0.26	0.38	0.26	bonn 34	0.41	0.30	0.43	0.29
bonn 00	0.35	0.25	0.37	0.24	bonn 35	0.36	0.22	0.37	0.24
bonn 01	0.34	0.26	0.37	0.24	bonn 36	0.36	0.22	0.37	0.23
bonn 02	0.36	0.28	0.39	0.25	bonn 37	0.36	0.23	0.38	0.23
bonn 03	0.39	0.28	0.41	0.26	bonn 38	0.38	0.25	0.40	0.26
bonn 04	0.43	0.31	0.46	0.31	bonn 39	0.42	0.27	0.44	0.30
bonn 05	0.35	0.20	0.37	0.16	bonn 40	0.39	0.30	0.41	0.26
bonn 06	0.36	0.20	0.37	0.16	bonn 41	0.39	0.29	0.41	0.27
bonn 07	0.37	0.21	0.38	0.18	bonn 42	0.40	0.28	0.42	0.27
bonn 08	0.38	0.22	0.40	0.19	bonn 43	0.42	0.29	0.44	0.29
bonn 09	0.42	0.26	0.44	0.24	bonn 44	0.45	0.32	0.48	0.31
bonn 10	0.39	0.28	0.41	0.27	bonn 45	0.43	0.28	0.44	0.29
bonn 11	0.40	0.28	0.41	0.27	bonn 46	0.43	0.28	0.44	0.29
bonn 12	0.40	0.29	0.42	0.27	bonn 47	0.44	0.28	0.46	0.29
bonn 13	0.42	0.32	0.43	0.28	bonn 48	0.45	0.30	0.47	0.30
bonn 14	0.45	0.35	0.47	0.31	bonn 49	0.47	0.34	0.50	0.34
bonn 15	0.33	0.17	0.34	0.16	bonn 50	0.41	0.27	0.44	0.27
bonn 16	0.33	0.16	0.34	0.16	bonn 51	0.41	0.27	0.44	0.27
bonn 17	0.33	0.17	0.35	0.16	bonn 52	0.42	0.28	0.45	0.28
bonn 18	0.35	0.18	0.37	0.17	bonn 53	0.43	0.29	0.47	0.28
bonn 19	0.38	0.20	0.41	0.23	bonn 54	0.47	0.29	0.50	0.28
bonn 55	0.40	0.31	0.41	0.29	bonn 78	0.40	0.29	0.42	0.28
bonn 56	0.40	0.31	0.41	0.29	bonn 79	0.44	0.32	0.45	0.32
bonn 57	0.40	0.30	0.42	0.30	bonn 80	0.41	0.23	0.42	0.25
bonn 58	0.42	0.29	0.44	0.32	bonn 81	0.41	0.23	0.42	0.25
bonn 59	0.46	0.33	0.48	0.35	bonn 82	0.42	0.23	0.43	0.25
bonn 60	0.36	0.19	0.37	0.21	bonn 83	0.43	0.24	0.45	0.26
bonn 61	0.36	0.19	0.37	0.21	bonn 84	0.45	0.27	0.48	0.30
bonn 62	0.37	0.20	0.38	0.22	bonn 85	0.36	0.27	0.38	0.27
bonn 63	0.39	0.22	0.39	0.23	bonn 86	0.37	0.27	0.38	0.26
bonn 64	0.42	0.24	0.43	0.26	bonn 87	0.38	0.28	0.39	0.26
bonn 65	0.36	0.17	0.37	0.15	bonn 88	0.40	0.30	0.41	0.28
bonn 66	0.36	0.17	0.37	0.15	bonn 89	0.44	0.32	0.44	0.33
bonn 67	0.37	0.17	0.38	0.16	bonn 90	0.41	0.32	0.44	0.31
bonn 68	0.39	0.18	0.40	0.17	bonn 91	0.41	0.31	0.44	0.30
bonn 69	0.42	0.20	0.44	0.23	bonn 92	0.42	0.33	0.45	0.31
bonn 70	0.39	0.24	0.41	0.23	bonn 93	0.44	0.34	0.46	0.32
bonn 71	0.39	0.27	0.41	0.23	bonn 94	0.47	0.36	0.50	0.35
bonn 72	0.40	0.28	0.42	0.24	bonn 95	0.41	0.24	0.41	0.24
bonn 73	0.42	0.29	0.44	0.25	bonn 96	0.41	0.23	0.41	0.25
bonn 74	0.46	0.33	0.47	0.29	bonn 97	0.41	0.22	0.42	0.26
bonn 75	0.35	0.22	0.38	0.22	bonn 98	0.42	0.24	0.44	0.26
bonn 76	0.35	0.21	0.38	0.22	bonn 99	0.45	0.30	0.47	0.31
bonn 77	0.37	0.24	0.39	0.24					

possible: (1) both vectors from class 1, (2) both from class 2 and (3) they are from different classes. If the feature set is readily separable, then it is reasonable to expect the distance histogram to exhibit three peaks, corresponding to these cases. Since the intra-cluster distances are typically smaller than inter-cluster distances, the peaks for the same-class feature pairs are on the left-hand side of the distance histogram. Very often, these two peaks overlap each other, so only a single peak appears on the plot. The inter-cluster distances appear as a separate peak to the right of the intra-cluster peak(s). Figure 5.2 illustrates the relevance of the peaks for a feature set extracted from a 2-texture mosaic. The plots clearly show the desired properties for a separable feature set, with the intra-cluster histograms possessing peaks at a far lower distance value than the inter-cluster histogram. The intra- and inter-cluster histograms are constructed with the knowledge of a known ground truth of the 2-texture mosaic.

For a multi-class problem, the average intra-class distance can be quite different for the different classes. Also, the average inter-class distances between any pair of classes can vary significantly. The result is the presence of multiple peaks which may overlap one another, hence reducing the usefulness of the distance histogram as a visual tool to estimate separability. Figure 5.3 illustrates this particular problem for a five-texture mosaic. While there is only a single peak in the histogram, there are clearly multiple "knee" points, which is an indication that there are several peaks overlapping each other. This phenomenon is apparent for the case illustrated in figure 5.4(a).

The problem is even more pronounced for the sixteen texture mosaic, as shown in figure 5.4. Due to space constraints, distance histograms of separate intra- and inter-cluster combinations are not individually calculated. From the distance histogram over the whole image, it is apparent that all the individual intra- and inter-cluster peaks overlap to form a smooth distribution of distances.

In light of the previous discussions, the distance histogram method is best suited to gauge the separability of two texture problems among the techniques described in this section. In order to examine the merits of DT-CxWT and DWT feature extraction schemes, the histograms of four different two-texture mosaics are computed.The same feature conditioning parameters are used for both transforms. In other words, the only difference in the feature extraction process is with the type of transform used, so a direct comparison is valid. The images can be found in Appendix A. Figures 5.5 and 5.6

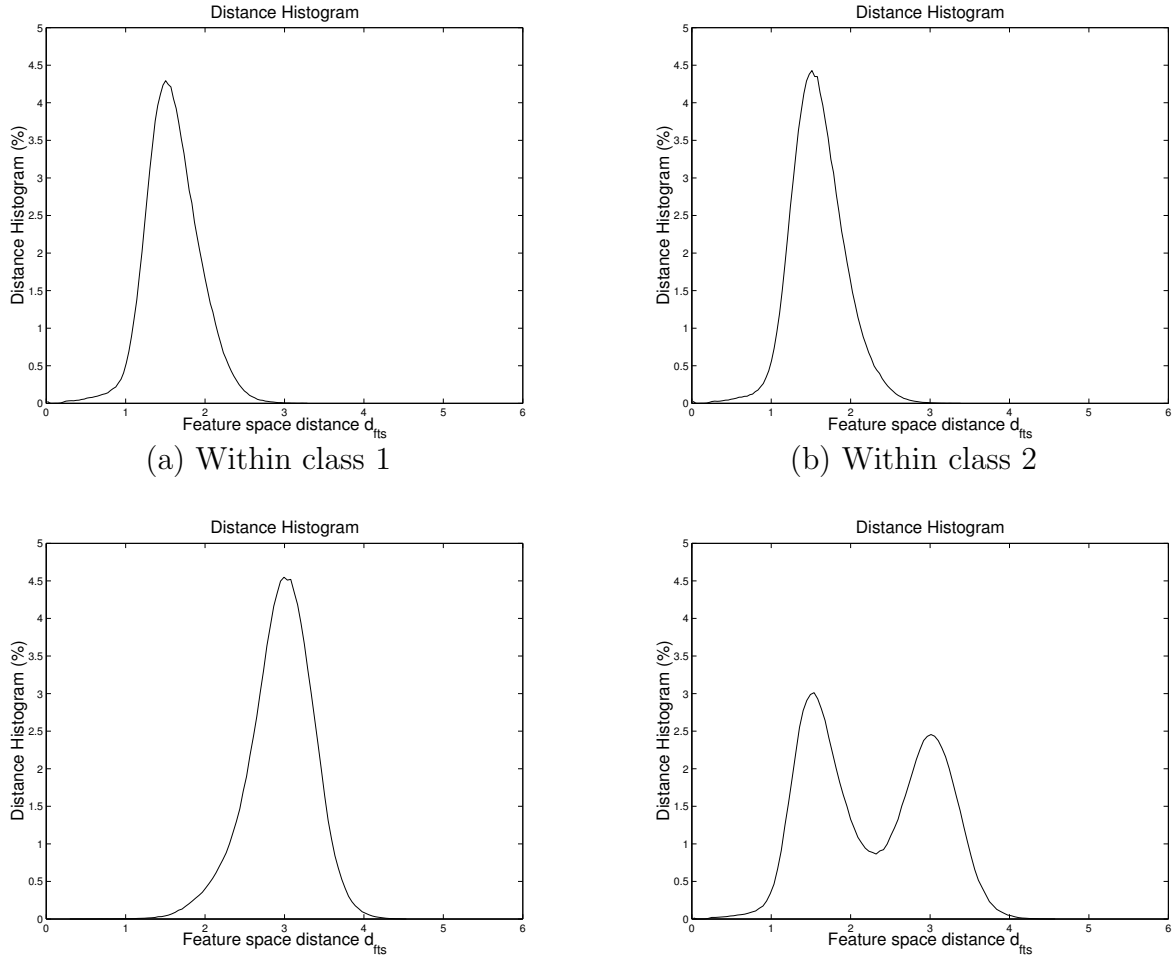


Figure 5.2. Distance histograms for two-texture mosaic D4-D84 illustrating the relevance of different peaks. (a) distance histogram for feature pairs within class 1; (b) within class 2; (c) different classes and (d) overall histogram. The peak for inter-class distance is clearly greater than intra-class distances for both class 1 and 2. Parameters used for feature extraction: DT-CxWT, depth 3, Kaiser window, $\beta = 0$.

show the distance histograms obtained. These histograms are computed from 10^6 pairs of feature vectors, selected randomly from extracted feature sets. This is approximately 3% of all possible pairs for these input images. It has been found from experiments that an exhaustive computation of all combinations does not provide extra information on the shape of the distance histogram. Therefore, a small subset of these pairs is chosen to reduce the computation requirements.

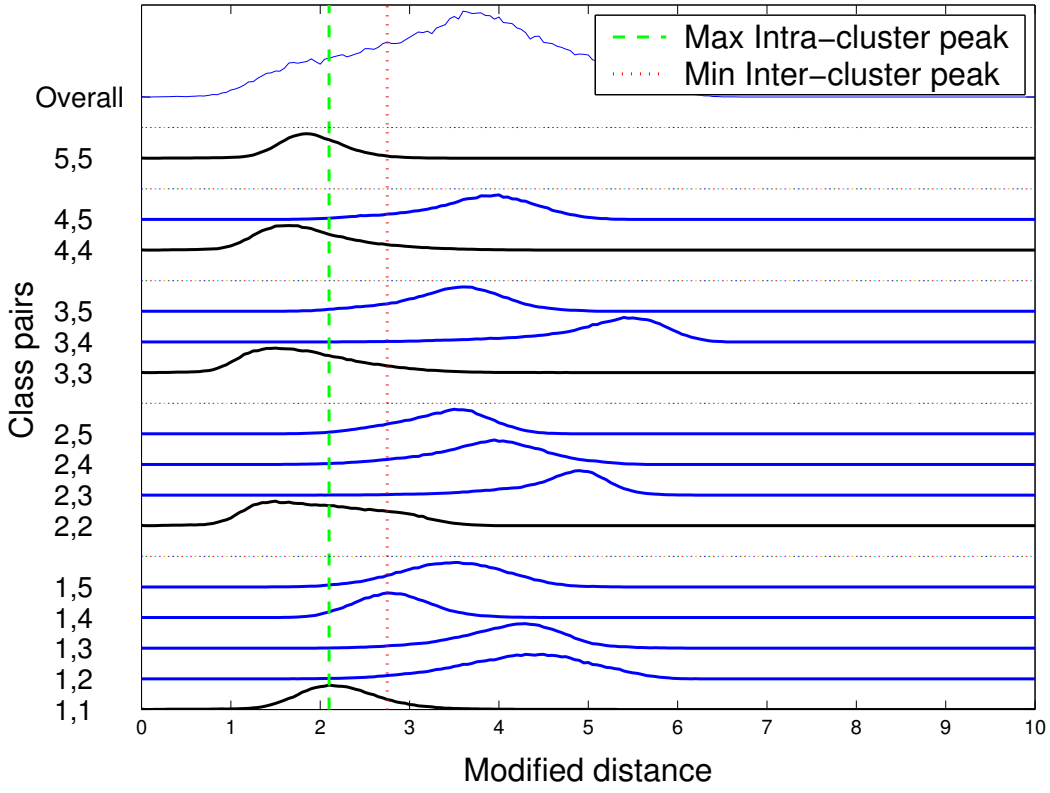
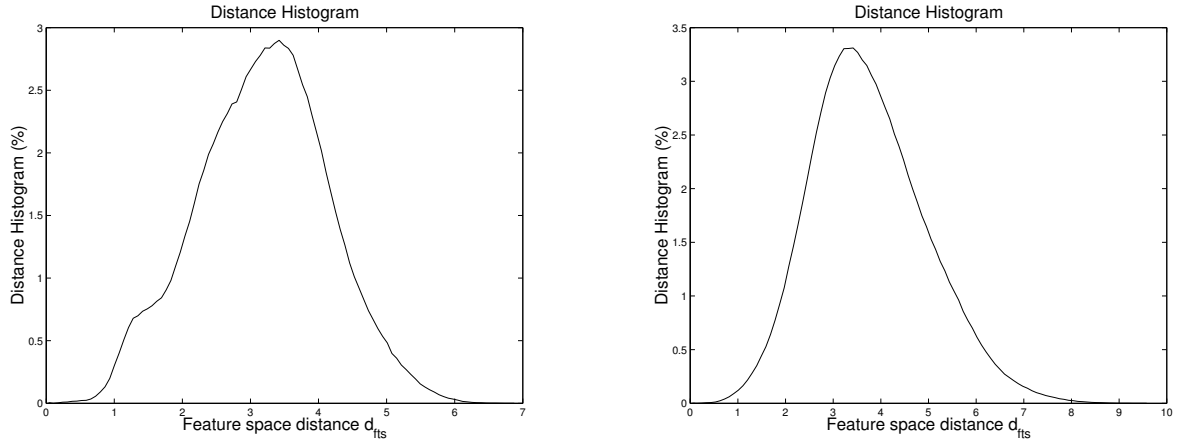


Figure 5.3. Distance histogram for a five-texture mosaic (Nat 5c). The inter-cluster distance histograms for all distinct pairs of clusters are shown in blue, while the intra-cluster histograms are shown in black. At the top is the overall distance histogram for all clusters combined. The green line indicates the position of the furthest right intra-cluster peak, while the red line shows the position of the furthest left inter-cluster peak. Feature extraction parameters: 3-level DT-CxWT transform, Kingsbury filter pairs, Kaiser smoothing window with $\beta = 3$, sigmoidal non-linearity.

From figures 5.5 and 5.6, it can be seen that the DT-CxWT features exhibit a classic two-class distance histogram for three of the four input images. There are clearly two distinct peaks, well-separated from each other by a valley. For the DWT features, the histograms are not as good. Input images D4-D84 (figure 5.5(a)) and D5-D92 (figure 5.6(a)) did not produce the desired "twin peaks" histograms, but there exist noticeable curvature changes, or "knees", on those plots. A clear knee also exists for DT-CxWT features extracted from image D5-D92 (figure 5.6(b)). These knees exist when the peaks corresponding to average intra- and inter-distances are close to each other. It was found in [24] that feature sets with a noticeable curvature change in the distance histogram have comparable segmentation performance as those with two distinct peaks. Thus, it can be concluded from the distance histograms that the features extracted in our experiments



(a) Nat 5b (b) Nat 16b

Figure 5.4. Distance histogram for the five- and sixteen-texture mosaics (Nat 5b and Nat 16b). Feature extraction parameters: 3-level transforms, Kaiser smoothing window with $\beta = 3$, sigmoidal non-linearity.

show good separability, which in turn suggests that good segmentation performance can be expected. Comparing between the transforms, the DT-CxWT features exhibit better distance histogram characteristics than the DWT features in general, simply because their histograms have more easily distinguishable peaks.

A comparison between different values of β for the Kaiser smoothing windows is presented in figure 5.7. For simplicity, a single two-texture mosaic (input image D4-D84) has features extracted using a 3-level DT-CxWT with different values of β for the smoothing window. For $\beta = 0$, when the Kaiser window is simply a rectangular window, the distance histogram exhibits excellent peak characteristics (figure 5.7(a)). The same applies when $\beta = 3$, which is found in figure 5.5(b). However, it has been noticed that as β increases, the intra- and inter-distance peaks converge towards each other until the valley between them disappears entirely (figures 5.7(c) and 5.7(d)). This behaviour is not surprising, given that greater values of β decrease the spatial width of the smoothing window. As a result, larger values of β for the smoothing window reduce the uniformity of features within the same cluster, because the windows would have a smaller effective size (figure 5.1). Indeed, the raw features are highly oscillatory, so an unsmoothed (corresponding to $\beta \to \infty$) feature set would have a unimodal distance histogram. Thus, the benefits of a smoothing process is apparent from the distance histograms. However, the optimal value of β varies from case to case, and hence a large range of values is used in the experiments.

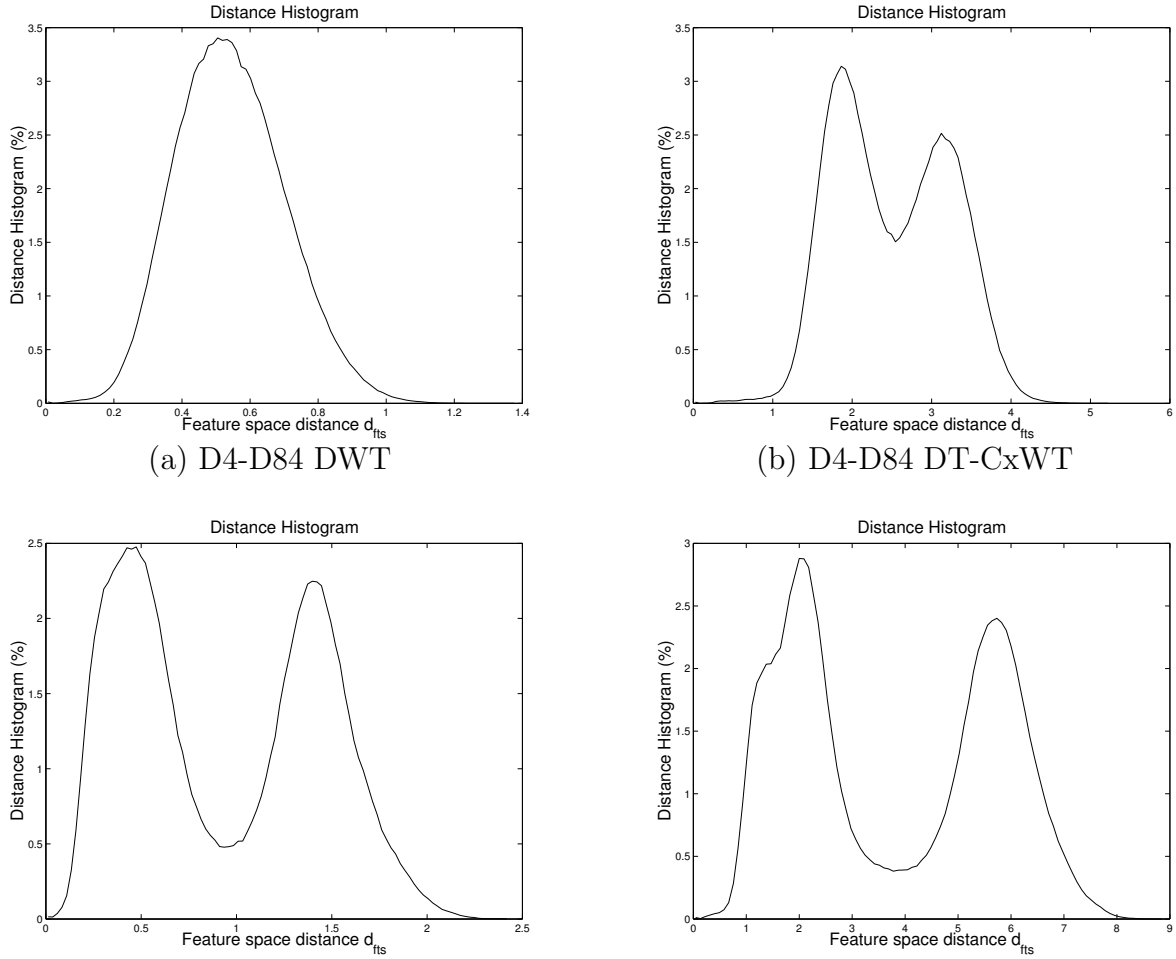


Figure 5.5. Distance histogram for two-texture mosaics D4-D84 and D8-D84. On the left-hand side are the histograms for DWT features, while the right-hand side shows the histograms for DT-CxWT features. There are clear differences in the histograms, with the DT-CxWT features exhibiting the classic twin peak distribution characteristic of separable feature sets. Feature extraction parameters: 3-level transforms, Daubechies 9-7 and Kingsbury filter sets, Kaiser smoothing window with $\beta = 3$, sigmoidal non-linearity.

5.2.4 Summary of Feature Separability

Combining the results of the three separability measures, it can be concluded that the feature extraction process described in Chapter 3 is capable of producing feature sets with good separability. However, all of these measures only focus on particular aspects of a feature set, and their results are not conclusive proof of separability. Ultimately, the

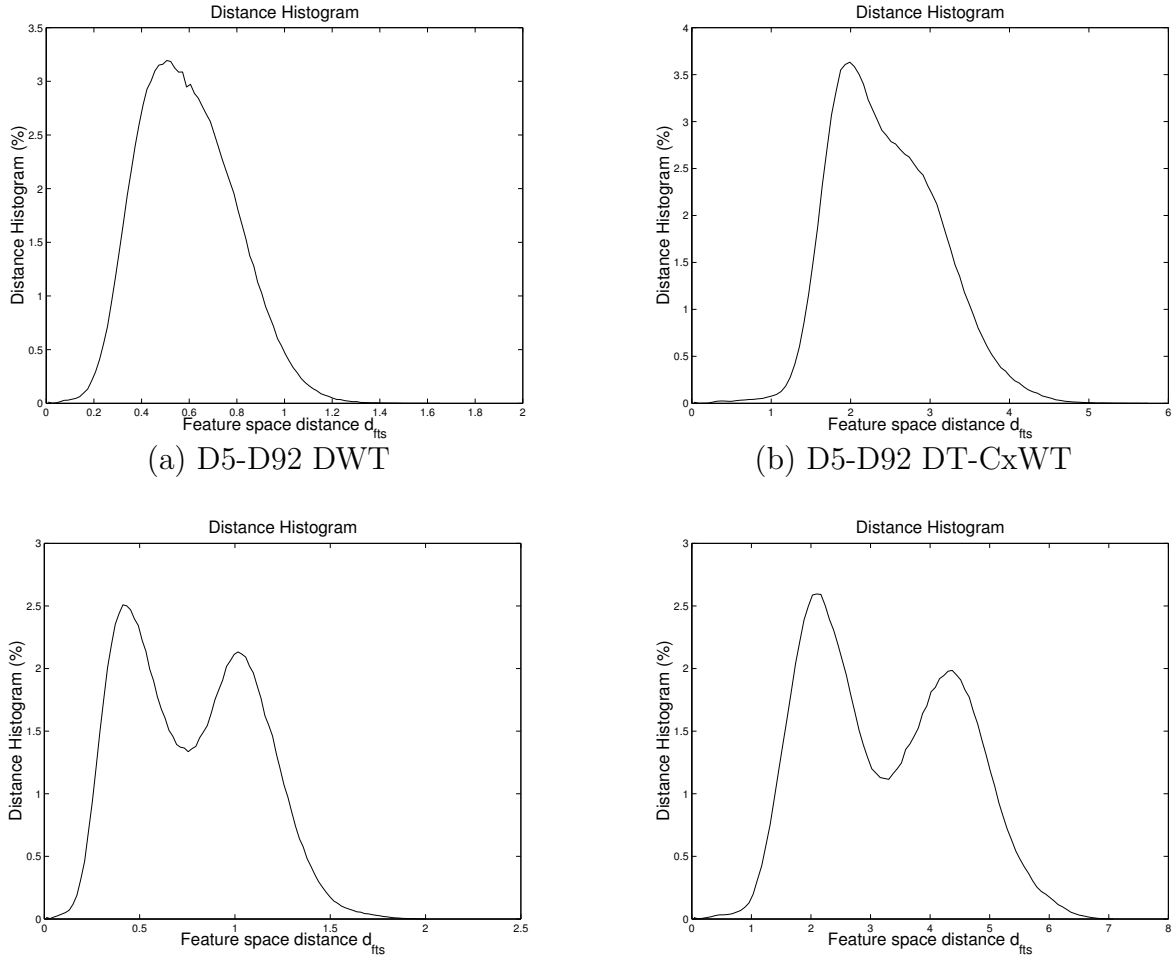
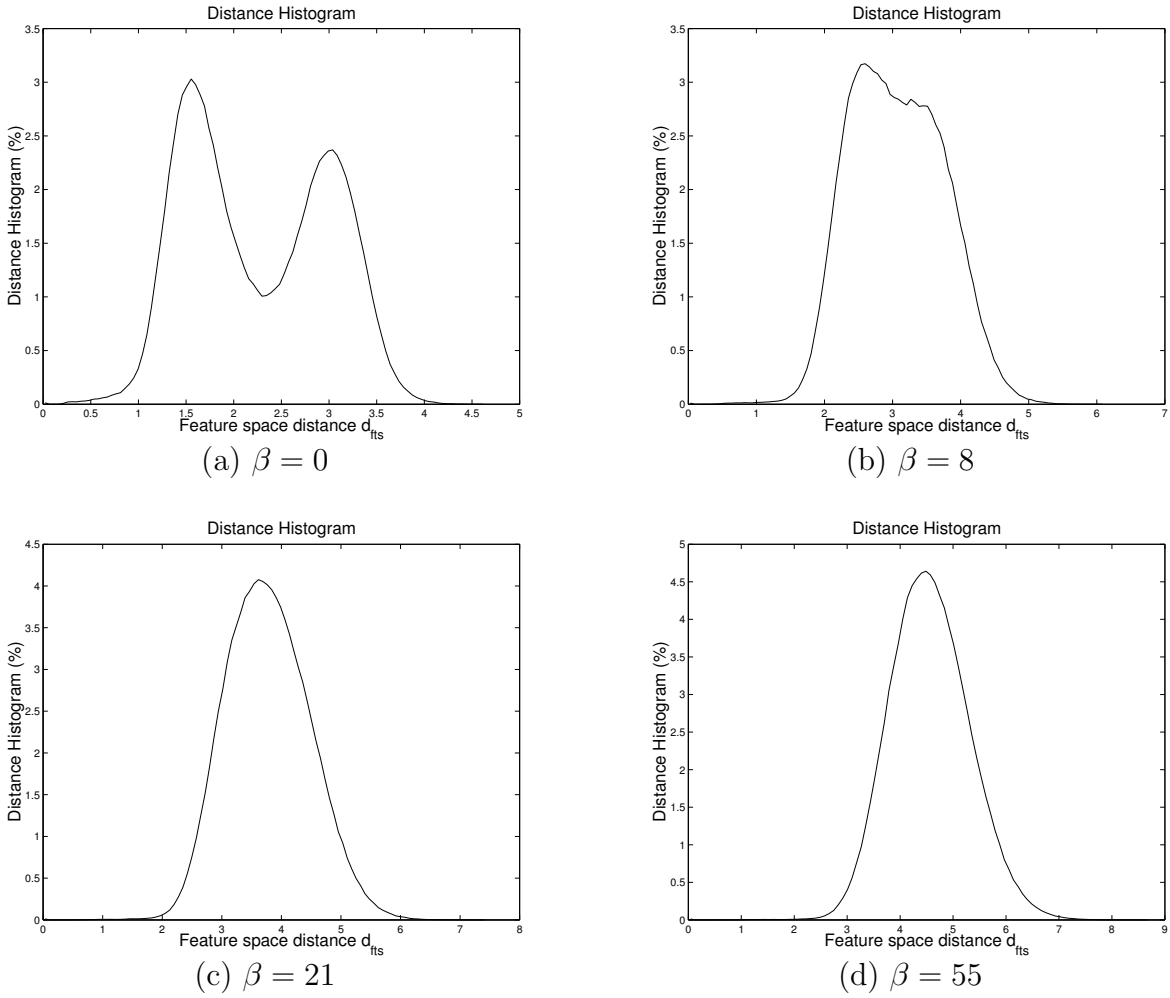


Figure 5.6. Distance histogram for two-texture mosaics (continued). The images are D5-D92 and D12-D17; left-hand side are DWT histograms, and right-hand side are from DT-CxWT. Again, the DT-CxWT features show better separability characteristics than the corresponding DWT features. Feature extraction parameters are the same as for figure 5.5.

quality of the extracted features is determined by the final segmentation results. These will be discussed in the following sections.



(a) $\beta = 0$ (b) $\beta = 8$

(c) $\beta = 21$ (d) $\beta = 55$

Figure 5.7. Distance histograms for different β. The input image used is D4-D84, 3-level DT-CxWT using Kingsbury filter pair.

5.3 Experiments with K-means

This section presents the results obtained from segmentation experiments obtained with K-means clustering method. While the previous section presented the separability of extracted feature set, it is necessary to examine the segmentation results in order to verify the true quality of the features.

Table 5.5 shows the segmentation results obtained from different wavelet features, for the entire suite of 115 input images. These tables are condensed as much as possible for brevity; a full collection of result tables can be found in Appendix B. These results provide direct comparisons between different transform types, depths and smoothing methods. The table 5.5 is organised into a number of columns. The left half of the table spanning the first four columns contains the segmentation error rates for median-filter smoothing. These are further distinguished by the type ("C" for DT-CxWT, "D" for DWT) and the depth (2 or 3) of transform used. The right half of the table shows the results from Kaiser window smoothing. This is also separated into four columns showing the different transforms and depths used.

Table 5.5. Condensed results of K-Means experiments. The best error rates (%) are shown for each input image. This condensed table of results only shows the best case; a full set of results for all values of β is found in Appendix B.

	C2M	D2M	C3M	D3M	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
					(%)	β	(%)	β	(%)	β	(%)	β
D4-D84	1.59	6.53	4.43	7.89	0.95	2	2.23	2	0.78	3	3.61	2
D5-D92	2.33	5.96	13.40	7.06	2.15	3	2.09	1	2.39	1	2.50	2
D8-D84	0.35	0.85	0.72	1.15	0.66	21	0.72	13	0.39	34	0.84	8
D12-D17	2.22	2.66	2.47	2.37	2.43	5	2.80	5	1.73	5	1.83	3
My 5a	14.15	13.44	15.29	18.66	7.35	5	9.85	5	8.73	13	11.39	8
My 5b	24.77	17.90	19.39	23.67	6.26	2	5.76	2	7.06	5	8.37	1
Nat 5b	12.45	44.10	38.02	43.29	22.34	13	16.35	3	28.58	8	9.54	0
Nat 5c	29.22	35.92	16.32	30.62	7.97	1	13.89	1	4.72	3	13.51	1
Nat 5m	13.26	35.51	38.02	36.94	7.61	2	10.52	1	6.63	0	10.85	1
Nat 5v2	41.11	50.07	38.02	52.29	26.40	0	28.61	1	12.27	1	31.09	2
Nat 5v3	26.48	42.05	38.02	44.46	14.28	0	15.97	0	11.86	0	16.56	0
Nat 5v	34.24	28.70	38.02	28.97	24.36	0	17.96	0	14.17	0	15.20	0
bonn 00	24.11	24.95	21.80	27.15	19.90	5	19.14	3	19.75	3	19.42	3
bonn 01	21.38	31.74	26.25	39.26	23.98	21	25.10	13	21.50	21	24.41	1
bonn 02	32.28	34.45	33.54	36.32	34.39	3	36.35	8	28.30	5	34.27	55
bonn 03	10.83	14.54	36.29	15.42	5.18	2	6.95	0	4.78	2	6.90	0
bonn 04	30.28	35.33	29.53	33.98	26.20	3	29.82	5	29.41	13	31.18	1
bonn 05	12.13	42.20	39.19	42.76	11.72	2	11.81	3	10.42	2	10.19	3
bonn 06	5.60	26.82	24.29	23.24	4.06	2	4.52	2	4.03	2	5.70	5
bonn 07	31.71	32.96	33.08	31.25	7.30	1	5.87	13	26.20	3	5.51	13
bonn 08	35.63	47.35	36.35	58.11	29.79	0	30.82	0	27.78	0	28.46	0
bonn 09	28.91	29.78	31.66	35.85	23.29	1	22.95	1	21.99	2	22.00	1
bonn 10	23.78	25.19	29.27	31.35	18.38	1	20.07	1	17.77	1	18.96	1
bonn 11	33.89	35.22	35.24	35.29	5.05	21	11.10	34	23.54	0	24.81	1

	C2M	D2M	C3M	D3M	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
					(%)	β	(%)	β	(%)	β	(%)	β
bonn 12	35.89	22.24	12.56	21.12	7.61	2	7.98	2	7.74	3	8.33	1
bonn 13	29.02	40.57	7.04	35.96	12.45	13	23.28	2	3.93	2	16.38	8
bonn 14	35.32	35.01	34.11	35.99	7.56	5	8.87	0	7.64	1	12.38	13
bonn 15	34.58	10.33	25.83	27.00	6.07	2	6.60	2	25.54	5	5.58	3
bonn 16	14.75	25.26	7.57	12.26	8.84	1	15.50	5	5.06	1	6.62	0
bonn 17	35.11	20.84	32.89	45.77	6.26	5	16.13	5	33.87	21	28.09	0
bonn 18	22.09	28.11	35.94	37.75	17.83	3	20.81	0	25.18	2	20.76	1
bonn 19	13.06	35.71	24.59	32.18	13.09	8	18.62	13	6.20	8	27.87	1
bonn 20	42.18	35.52	41.57	33.97	28.55	5	28.71	2	28.03	34	36.87	34
bonn 21	21.40	23.36	23.11	26.64	15.32	0	11.32	1	16.20	0	11.37	0
bonn 22	24.63	47.57	33.59	42.33	14.80	8	18.52	2	16.33	8	15.70	3
bonn 23	6.69	11.30	27.68	30.10	11.16	21	10.37	13	25.39	21	13.43	21
bonn 24	39.95	30.54	23.79	43.08	24.96	2	18.57	3	29.91	0	11.78	2
bonn 25	24.80	28.23	24.70	28.75	20.07	0	22.13	0	19.57	1	7.71	1
bonn 26	28.08	28.48	28.88	24.55	7.65	13	13.81	21	8.89	21	13.01	21
bonn 27	29.82	15.81	27.44	35.77	9.61	2	10.27	2	26.84	1	28.92	2
bonn 28	31.21	41.48	28.96	40.26	22.46	2	17.62	3	19.46	2	24.55	1
bonn 29	5.28	10.28	25.31	11.05	4.71	5	6.73	3	3.08	5	5.50	3
bonn 30	34.13	35.77	32.78	37.82	24.13	5	24.38	5	23.99	8	24.15	2
bonn 31	6.59	38.28	28.96	25.94	28.59	1	4.89	5	4.44	3	28.14	1
bonn 32	37.11	20.76	33.47	35.96	35.61	1	13.96	2	32.37	0	33.04	0
bonn 33	23.02	25.43	23.28	25.83	5.59	3	5.65	0	14.94	1	15.73	2
bonn 34	35.74	31.98	17.55	35.95	23.85	1	21.36	3	17.60	8	14.14	2
bonn 35	23.56	26.36	24.74	27.03	4.24	8	5.66	1	4.94	8	5.49	2
bonn 36	31.71	35.75	26.89	36.38	9.59	2	14.92	0	13.43	3	12.23	2
bonn 37	4.39	7.20	3.80	6.64	3.34	3	4.64	3	2.77	5	3.86	3
bonn 38	10.57	26.79	37.26	27.48	8.58	3	9.41	1	33.23	1	9.06	2
bonn 39	2.28	4.69	30.77	5.49	1.83	5	2.39	5	2.01	8	2.38	3
bonn 40	2.72	4.90	30.60	7.65	4.08	3	4.05	1	5.14	2	28.09	1
bonn 41	36.87	38.73	37.16	39.45	35.14	2	14.00	0	35.20	3	34.43	2
bonn 42	25.68	28.13	20.27	26.73	18.99	1	19.70	3	18.17	1	18.93	3
bonn 43	36.06	48.74	36.60	48.90	9.67	2	15.89	3	6.83	3	17.41	3
bonn 44	23.09	31.42	38.71	56.87	8.17	3	10.60	3	28.10	34	9.55	2
bonn 45	14.15	21.61	15.72	22.01	14.65	5	14.77	3	14.67	3	14.04	2
bonn 46	32.04	27.54	36.93	25.33	28.78	2	35.21	0	25.98	0	28.62	0
bonn 47	9.80	34.11	33.94	35.51	5.78	1	8.07	1	6.03	0	11.33	1
bonn 48	24.34	26.53	25.64	28.73	19.28	2	7.21	3	6.35	0	25.62	0
bonn 49	15.21	44.43	19.34	45.92	8.86	2	12.00	1	25.76	0	28.42	0
bonn 50	12.04	19.79	28.52	24.72	27.42	5	14.54	1	13.62	0	11.68	1
bonn 51	31.16	5.70	29.89	33.98	22.58	5	21.17	0	4.92	21	4.51	3
bonn 52	33.35	6.65	27.62	28.50	5.07	8	4.77	3	7.23	13	21.68	2
bonn 53	35.38	35.24	36.58	34.06	28.60	2	25.30	2	29.57	0	28.23	2
bonn 54	14.47	28.12	12.51	15.94	7.61	3	8.90	2	9.11	1	9.66	1
bonn 55	7.78	17.63	22.75	21.92	6.27	3	10.41	1	6.78	1	7.51	1
bonn 56	42.90	41.94	31.37	39.37	28.86	3	37.70	3	22.45	1	25.30	1
bonn 57	10.93	21.03	23.18	21.90	6.13	2	8.40	0	5.58	2	6.54	0
bonn 58	16.00	18.29	5.46	11.69	7.65	3	18.76	5	5.40	8	7.97	0
bonn 59	11.46	16.00	9.58	14.69	10.58	3	12.38	3	9.69	3	10.20	3
bonn 60	26.69	14.68	25.67	29.39	20.65	2	6.13	1	26.18	34	21.57	0
bonn 61	24.29	9.03	21.29	21.65	5.02	5	17.76	2	7.01	2	20.22	8
bonn 62	29.24	34.63	24.40	36.85	7.10	3	9.55	1	7.60	2	23.13	2
bonn 63	11.18	14.50	35.93	38.60	24.18	5	9.47	1	25.60	8	15.85	13
bonn 64	11.30	32.58	31.47	34.19	11.46	2	23.32	1	13.97	1	14.94	0
bonn 65	4.34	25.10	43.22	25.10	5.63	0	9.92	5	19.37	21	21.18	1
bonn 66	3.32	6.04	3.74	7.54	2.54	2	3.34	2	2.76	2	3.14	3
bonn 67	24.25	26.46	23.76	24.86	19.73	0	21.17	1	17.46	0	19.52	0
bonn 68	32.52	49.01	28.31	28.99	12.34	1	28.94	8	21.02	0	21.64	0
bonn 69	12.87	22.96	11.13	27.15	6.38	1	8.56	1	7.42	0	7.89	0
bonn 70	10.81	12.84	37.01	15.67	13.71	0	25.00	5	31.20	1	15.93	1
bonn 71	17.60	14.75	17.33	26.32	8.40	2	8.33	3	5.95	5	8.25	3
bonn 72	16.75	9.62	15.80	27.37	10.94	34	5.86	1	17.21	2	4.43	5
bonn 73	30.38	34.20	22.18	40.22	15.59	0	28.41	1	15.89	0	20.14	0
bonn 74	5.98	14.28	5.57	19.48	4.46	3	4.89	2	4.36	2	5.88	2
bonn 75	4.43	35.66	18.56	28.37	3.93	2	7.78	3	28.22	0	21.28	5

	C2M	D2M	C3M	D3M	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
					(%)	β	(%)	β	(%)	β	(%)	β
bonn 76	29.39	36.18	29.56	35.02	7.28	1	9.19	1	6.92	2	10.99	0
bonn 77	49.17	35.71	49.53	49.44	39.94	1	27.76	1	40.22	2	36.16	21
bonn 78	23.24	25.51	25.81	31.17	16.86	5	18.69	3	16.75	5	19.00	3
bonn 79	32.55	33.78	46.54	33.25	12.99	0	15.48	0	11.60	0	24.35	2
bonn 80	5.69	17.97	19.75	12.21	5.87	5	8.18	2	4.83	8	6.68	2
bonn 81	12.54	37.02	26.93	37.10	8.39	3	23.77	1	12.41	13	25.16	1
bonn 82	48.65	43.74	49.07	49.27	40.78	21	30.64	1	37.83	0	37.08	2
bonn 83	6.04	24.58	22.70	28.03	5.85	3	7.98	1	4.59	5	6.37	5
bonn 84	33.15	12.22	5.43	36.35	7.04	3	7.79	3	6.15	3	6.22	5
bonn 85	21.56	28.52	26.60	34.86	11.95	0	15.83	0	11.87	1	16.05	0
bonn 86	24.34	29.38	45.12	27.89	18.78	2	16.46	5	18.44	1	17.41	0
bonn 87	33.31	37.16	37.58	40.84	31.04	3	12.50	3	31.10	3	31.60	3
bonn 88	7.88	14.12	21.80	23.02	7.68	5	7.38	5	7.68	3	8.03	5
bonn 89	4.50	7.82	32.85	8.27	5.15	13	6.37	2	4.29	13	6.51	1
bonn 90	5.58	11.33	5.23	26.23	4.46	5	5.07	3	5.06	1	4.93	3
bonn 91	42.60	22.92	37.68	30.18	16.95	2	15.40	3	9.88	2	23.75	8
bonn 92	3.88	26.52	4.72	25.62	4.16	3	4.98	1	27.12	3	8.59	21
bonn 93	29.73	11.04	27.12	14.02	23.52	8	24.88	3	7.61	0	8.98	0
bonn 94	7.48	13.84	30.58	31.49	5.69	3	7.01	2	6.30	2	8.04	0
bonn 95	13.96	19.74	17.76	26.04	13.01	0	15.64	0	14.75	1	15.42	0
bonn 96	24.65	29.22	24.96	53.05	17.29	2	19.67	0	15.69	2	16.79	0
bonn 97	31.79	29.64	32.11	9.52	5.02	21	8.28	21	4.71	8	5.60	5
bonn 98	35.52	29.60	32.57	36.59	7.71	2	9.81	0	7.52	2	16.33	1
bonn 99	9.80	28.38	4.52	17.33	4.08	5	16.26	3	3.67	5	4.72	2
Nat 10	30.34	54.82	51.58	54.87	17.83	3	29.76	0	13.07	2	23.10	0
Nat 10v	43.63	55.98	62.26	58.70	36.05	3	38.44	3	34.20	0	32.68	0
Nat 16b	53.61	62.04	65.66	60.95	37.94	1	41.00	0	30.72	0	37.54	0

An executive summary of results is presented in table 5.6. This smaller table provides a direct comparison between different transform types, depths and smoothing methods. The quantities calculated are the overall mean, median and the 95% confidence intervals on the means. It is assumed that the error rates of different images form a normal distribution. The confidence intervals are thus estimated using the t-distribution [40]. Two observations can be drawn from table 5.6. Firstly, it is clear that the Kaiser windowing method gives superior results when compared with the median filtering method. On average, the Kaiser filtering method gives error rates approximately 10% lower than the median filter smoothing method, which is a very significant difference. This can be explained by the fact that the Kaiser windows are more flexible than median filters, since they have a freely adjustable window shape parameter in β. This added flexibility allows the Kaiser windows to better adapt to different textures. Secondly, it is found that a depth 2 DT-CxWT gave the best results with K-means clustering, followed by a depth 2 DWT, then depth 3 DT-CxWT and lastly, depth 3 DWT. This trend remains true for both Kaiser and median filter smoothing methods. However, the differences in performance between the different transform types and depths are not very large, with the gap being a mere 2.2%. It is evident that the added complexity of depth 3 transforms gave the

simple K-means algorithm problems; the extra information in the additional subbands invoked the infamous "curse of dimensionality". However, the depth 2 DT-CxWT features, with 28 dimensions, managed to slightly out-perform the 10-dimensional, depth 3 DWT features. This is evidence that the DT-CxWT is more suited to texture feature extraction than the DWT, which was expected, since the DT-CxWT has superior directionality and shift-invariance over the conventional DWT.

Table 5.6. K-means results executive summary. The overall mean and median error rates for the different transform/smoothing combinations are listed in this table. In addition, the 95% confidence interval for the means are shown.

Transform type/depth	Median Filtering			Kaiser Filtering		
	Mean	Median	95% C.I.	Mean	Median	95% C.I.
DT-CxWT2	22.10	24.11	(19.74, 24.46)	13.96	10.58	(12.13, 15.80)
DT-CxWT3	26.85	27.44	(24.58, 29.11)	15.01	13.07	(13.12, 16.90)
DWT2	26.60	27.54	(24.24, 28.97)	14.79	13.81	(13.12, 16.46)
DWT3	29.94	29.39	(27.61, 32.28)	16.11	15.20	(14.35, 17.88)

Overall, the K-means segmentation results for 2-texture mosaics are good, with the algorithm achieving error rates of less than 3% for the Kaiser smoothing cases. These are simple input images, with a single straight boundary separating the two textured regions. However, the results show that the texture features are very capable of distinguishing between two different textures. The results for the 2-texture mosaics compare favourably to those in the literature. For example, the D12-D17 mosaic was segmented by Randen and Husøy in their experiments [64]. They reported segmentation error rates between 2.1% and 30.0% for their various optimised texture feature extractors. In contrast, the results in table 5.5 shows the error rates to be between 1.73% and 2.80% for the same mosaic. In order to further explore the performance of the algorithm for 2-texture segmentations, several qualitative experiments are performed using different texture mosaics. The left-hand column of figure 5.8 shows a set of 2-texture mosaic consisting of Brodatz textures. These are constructed from the same textures as those used in the K-means experiments, but with a different set of separating boundaries. These images consist of a uniform background texture, on which exists a cross-shaped region in the middle with another texture. The boundary between the two textures is much more complicated than in the previous 2-texture mosaics. These additional experiments on more demanding cases are aimed at increasing the confidence in the ability of wavelet features to distinguish between a given pair of textures. Table 5.7 shows the results of the 2-texture cross mosaics

segmentations, and the right-hand column of figure 5.8 shows the visual segmentation results. The results follow a similar pattern to the simpler 2-texture mosaics, namely, that all cases except D5-D92 have excellent error rates. While the results are not as good as the simpler mosaics', the error rates are still below 3% for most cases. From figure 5.8(c), it is seen that the two constituent textures of D5-D92 are visually very similar, and the cross boundary is difficult to locate even with the human eye, which accounts for the unusually poor segmentation performance for this case. Compare this situation with the other three mosaics, where the differences between the textures are more obvious to human interpretation, it is reassuring to see this set of segmentation results aligning with human intuition. In summary, there is strong experimental evidence in the combined results to suggest that the texture features described here are capable of distinguishing pairs of textures.

Table 5.7. Segmentation results for 2-texture cross mosaics. The best error rates (%) for the four images are shown here.

Image	DT-CxWT2	DWT2	DT-CxWT3	DWT3
D4-D84 cross	1.65	3.22	1.83	4.07
D5-D92 cross	8.06	9.17	7.69	9.85
D8-D84 cross	2.37	2.17	2.36	2.15
D12-D17 cross	2.30	2.68	2.34	2.91

The K-means results are much more complicated for the input mosaics with more textures. The increase in complexity from having more constituent textures make the 5-, 10- and 16-texture segmentation problems far more difficult than for the 2-texture cases. The error rates for these mosaics range from 3% to 41% for the Kaiser filtering cases. For the median filter smoothing method, the results are even worse, with the highest error rate being 65%. From these experiments, it can be concluded that Kaiser window filtering is superior to the median filtering method. However, the performance for the multi-class cases is relatively poor when compared with the 2-texture cases. This fact has already been indicated by the distance histograms of those multi-class cases; the peaks among and between classes overlap to make a pure distance-based clustering algorithm ineffective. This problem will be addressed in section 5.5.

The important parameter in the Kaiser smoothing cases is β, which controls the effective width of the window (section 3.4.3). An appropriate choice of β would improve the segmentation accuracy tremendously, a fact borne out from the experimental results.

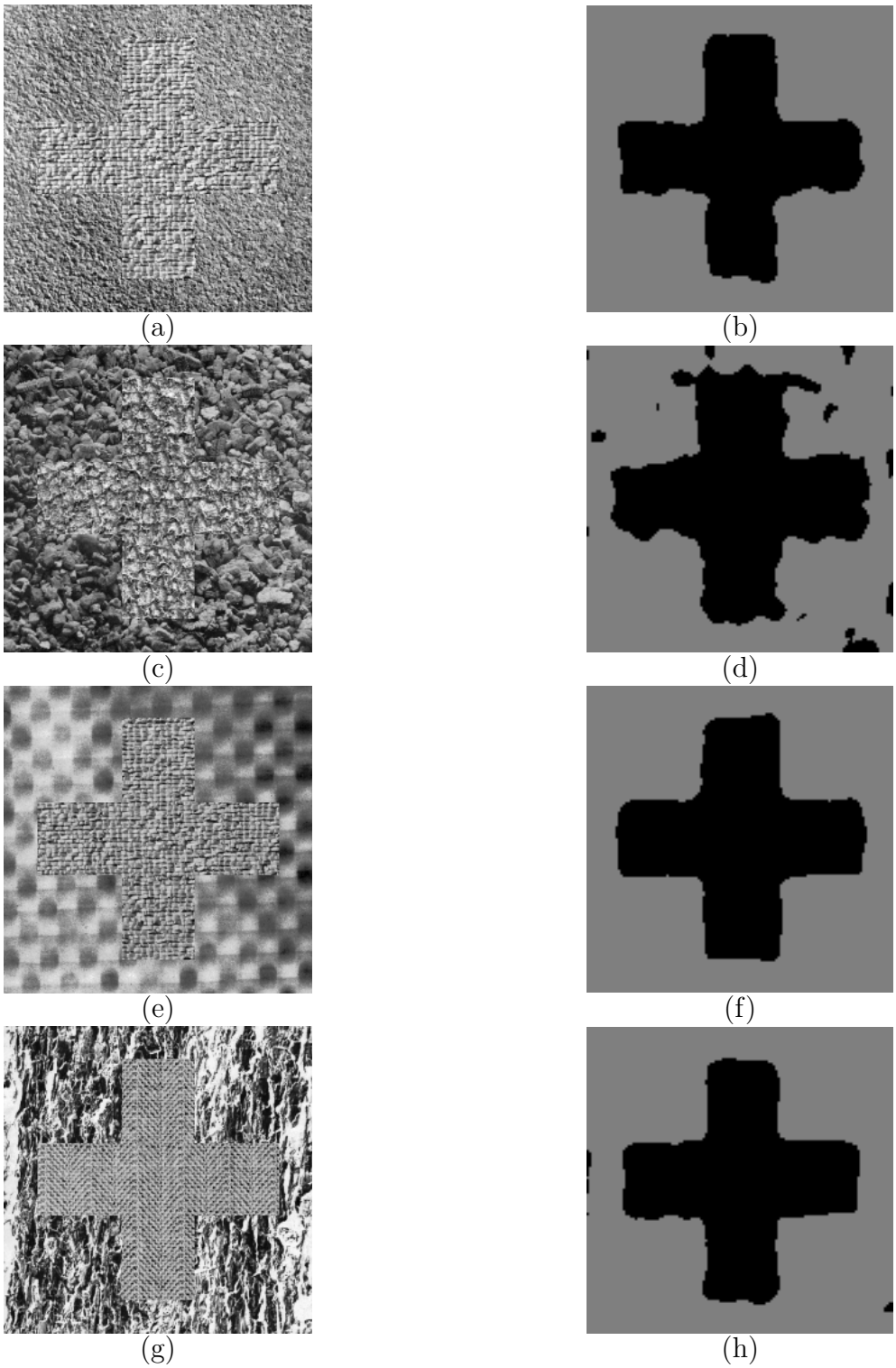


Figure 5.8. Visual segmentation results for 2-texture cross mosaics. (a) D4-D84 cross, (b) D4-D84 cross segmentation, (c) D5-D92 cross, (d) D5-D92 cross segmentation, (e) D8-D84 cross, (f) D8-D84 cross segmentation, (g) D12-D17 cross, (h) D12-D17 cross segmentation.

In fact, different values of β may yield extremely different segmentations; this can be seen in the detailed result tables in Appendix B. This wild variations in segmentation accuracy can be explained by the differences in texture primitive sizes in some cases. The original motivation to use window smoothing on features was to average wavelet features over the size of primitives present in the textures. For example, consider the input image D5-D92 shown in figure A.9. For $\beta = 1$, the smoothing window's effective size is sufficiently large to adequately smooth the texture features; the windowed area captures the information of the constituent textures. The result is a good segmentation of the regions (see figure 5.9(a)), with the errors mainly occurring at the boundaries. However, for $\beta = 34$, the smoothing window becomes too narrow, and the windowed area is too small to capture the characteristics of the textures. This results in a poor segmentation, with many small residual regions, as shown in figure 5.9(b).

(a) $\beta = 1$ (b) $\beta = 34$

Figure 5.9. Effect of β on segmentation for Kaiser smoothing windows. Segmentation results for image D5-D92, using: (a) $\beta = 1$, (b) $\beta = 34$ as the feature smoothing parameter. Other feature extraction parameters: DT-CxWT, Kingsbury filter pair, transform depth 3.

In light of the importance of the β parameter, it is interesting to examine the segmentation results for different values of β across the suite of input images. Table 5.8 shows the frequency of optimal β's for all the cases. While there is a fairly uniform distribution for low values of β, values of optimal β greater than 8 occur far less often. In fact, in only one case does the optimal β equal to 55, the largest value used in the experiments. Therefore, a general recommendation is for small values of β when extracting features for Brodatz-like textures.

This section has presented results from the extensive texture segmentation experiments performed on the suit of input images. The wavelet texture features have been shown to be effective in distinguishing between two textures. Good segmentation performance was achieved for such cases. However, the performance for multi-class cases is less satisfactory. In some cases, the K-means algorithm produces very good results, but fails

Table 5.8. K-means results executive summary - part 2. The optimal Kaiser window β parameter values for all images, transform types and depths are tallied from the K-means experiments to produce the distribution of optimal β.

Transform type/depth	β									
	0	1	2	3	5	8	13	21	34	55
DT-CxWT 2	13	15	28	24	19	5	4	6	1	0
DT-CxWT 3	24	19	21	15	10	11	5	6	4	0
DWT 2	20	28	18	26	13	2	5	2	1	0
DWT 3	30	25	22	17	7	5	3	4	1	1

in some others. It is suspected that the primary problem is due to the added complexity of having multiple classes, each with its own feature space distance distribution. Overall, the experiments showed the Kaiser filtering method to be far superior to the median filtering method for feature smoothing. It was also discovered that the DT-CxWT features produced consistently better segmentation results than the DWT features, while a transform depth of 3 is less successful than a depth of 2.

5.4 Experiments with Fuzzy K-means

In this section, the results from experiments using fuzzy K-means clustering is presented. Table 5.9 shows the segmentation results obtained for the entire suite of 115 input images. Only the Kaiser window smoothing method is considered in these experiments, as it has been shown in the previous section that this is superior to median filtering for wavelet texture features. The result tables are organised into four main columns, each corresponding to the four different wavelet transform type and depth combinations used in the experiments.

Table 5.9. Condensed results of Fuzzy K-Means experiments. The best error rates (%) are shown for each input image. This condensed table of results only shows the best case; a full set of results for all values of β is found in Appendix B.

	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
	(%)	β	(%)	β	(%)	β	(%)	β
D4-D84	2.27	2	0.78	2	4.08	2	1.01	5
D5-D92	2.38	1	2.61	2	2.54	0	2.43	0
D8-D84	0.66	8	0.61	8	0.81	5	0.37	55
D12-D17	2.21	3	1.95	13	1.62	3	1.46	8
My 5a	13.34	5	12.05	2	11.71	1	15.58	1
My 5b	6.68	1	6.71	3	9.09	1	6.68	5
Nat 5b	14.67	0	7.25	1	10.47	0	8.07	0

	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
	(%)	β	(%)	β	(%)	β	(%)	β
Nat 5c	13.28	1	7.51	2	15.90	0	5.99	0
Nat 5m	10.66	1	7.73	0	10.82	0	7.59	0
Nat 5v2	32.53	0	32.89	0	40.08	0	31.28	1
Nat 5v3	13.99	0	12.09	0	22.66	0	30.65	0
Nat 5v	14.71	0	12.59	0	13.59	0	41.41	0
bonn 00	21.84	13	18.84	21	19.56	8	16.94	5
bonn 01	17.50	3	18.09	3	32.17	1	16.85	0
bonn 02	14.81	0	12.09	0	15.38	1	20.39	0
bonn 03	7.21	0	4.58	0	7.45	0	4.03	2
bonn 04	31.08	2	27.56	55	35.47	55	31.67	34
bonn 05	10.55	3	8.94	3	9.03	1	8.14	3
bonn 06	4.60	1	4.33	2	4.79	3	4.21	3
bonn 07	5.78	5	4.68	21	6.38	13	5.88	13
bonn 08	21.25	0	23.75	1	41.46	0	37.23	2
bonn 09	16.82	1	16.34	2	28.01	0	20.40	2
bonn 10	27.64	0	26.15	0	25.98	0	28.47	0
bonn 11	2.95	2	3.54	13	5.38	0	26.65	13
bonn 12	7.77	2	7.45	3	9.01	0	8.07	2
bonn 13	10.00	1	5.35	2	12.95	0	32.05	0
bonn 14	5.97	3	5.34	3	6.34	3	5.64	3
bonn 15	5.91	2	4.48	3	6.11	1	5.49	13
bonn 16	7.56	2	6.82	0	5.90	0	4.52	0
bonn 17	11.74	2	6.17	3	15.90	2	9.03	1
bonn 18	21.92	0	21.31	0	29.13	0	24.97	0
bonn 19	18.24	13	10.14	8	31.24	13	8.12	8
bonn 20	29.25	1	28.68	2	31.08	0	28.42	1
bonn 21	9.45	3	10.53	3	10.93	2	11.80	2
bonn 22	13.15	1	15.88	1	25.58	0	23.74	1
bonn 23	8.11	5	7.93	5	9.92	3	8.83	3
bonn 24	16.58	2	13.42	2	22.11	55	8.81	0
bonn 25	21.61	5	15.03	0	10.99	0	14.15	5
bonn 26	6.49	3	5.78	3	6.18	2	18.53	3
bonn 27	10.36	2	9.00	3	10.64	5	8.02	5
bonn 28	32.03	0	28.85	0	33.03	0	26.78	2
bonn 29	5.50	3	3.83	5	6.09	3	3.90	5
bonn 30	17.77	21	25.85	0	32.18	0	24.88	5
bonn 31	4.77	3	3.36	3	5.87	3	4.92	3
bonn 32	13.12	1	12.26	1	12.35	1	11.09	0
bonn 33	5.79	3	5.79	3	8.02	0	15.34	34
bonn 34	12.19	1	9.19	2	13.02	1	13.76	0
bonn 35	5.12	3	4.26	8	6.54	2	4.91	8
bonn 36	12.51	1	6.95	1	12.46	1	11.78	1
bonn 37	3.85	2	3.15	3	4.00	3	3.17	3
bonn 38	9.01	0	7.86	0	9.78	1	7.69	2
bonn 39	2.40	5	1.79	8	2.96	3	2.36	3
bonn 40	3.30	5	3.44	5	4.95	5	3.97	5
bonn 41	14.29	0	14.45	0	19.60	0	18.93	0
bonn 42	21.48	0	22.05	21	22.07	13	12.54	8
bonn 43	13.45	0	8.21	2	20.75	0	28.00	3
bonn 44	11.54	1	9.91	0	11.71	0	8.92	0
bonn 45	13.81	3	13.41	2	16.19	1	13.26	0
bonn 46	18.85	0	27.63	0	26.90	0	25.79	0
bonn 47	8.54	2	6.48	1	11.33	1	7.52	0
bonn 48	23.97	13	17.36	34	7.22	0	5.91	0
bonn 49	11.84	1	8.29	1	19.62	5	7.59	0
bonn 50	10.01	2	8.81	2	13.24	0	9.83	0
bonn 51	4.52	3	4.17	3	4.30	1	3.32	1
bonn 52	4.20	3	4.38	3	5.91	3	28.88	1
bonn 53	34.02	3	34.59	2	34.53	3	32.75	8
bonn 54	9.34	1	7.65	2	10.49	0	8.48	0
bonn 55	10.18	2	5.85	3	7.97	2	6.83	1
bonn 56	33.97	0	37.90	2	26.06	0	24.47	3
bonn 57	7.12	1	5.82	0	6.85	0	6.54	0
bonn 58	19.13	2	8.58	1	8.36	0	6.12	3
bonn 59	12.29	2	10.55	1	10.82	1	9.83	2
bonn 60	5.87	2	22.22	13	22.20	1	16.80	13

	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
	(%)	β	(%)	β	(%)	β	(%)	β
bonn 61	5.24	2	5.91	2	9.57	0	9.19	0
bonn 62	9.10	1	7.48	1	22.64	3	19.07	0
bonn 63	6.98	1	6.24	2	7.66	0	10.22	2
bonn 64	10.11	1	11.79	2	14.62	0	14.64	0
bonn 65	9.89	1	3.73	8	4.03	2	17.45	55
bonn 66	2.94	2	2.26	5	3.32	1	3.25	3
bonn 67	9.20	1	10.07	1	18.35	1	15.72	1
bonn 68	12.10	2	10.59	2	10.02	1	21.79	5
bonn 69	8.22	1	7.56	0	9.80	0	9.42	2
bonn 70	8.02	1	7.91	0	11.29	0	11.89	0
bonn 71	7.82	3	7.31	5	9.14	3	7.56	3
bonn 72	5.52	2	4.17	3	4.71	2	5.14	1
bonn 73	14.84	0	15.28	0	49.95	1	35.90	0
bonn 74	5.15	2	4.66	3	7.09	1	4.59	0
bonn 75	8.10	3	4.24	2	7.12	1	4.42	0
bonn 76	9.12	1	7.06	0	11.54	0	7.22	0
bonn 77	26.71	0	32.40	1	34.51	1	32.98	13
bonn 78	5.74	3	5.33	5	7.71	2	15.80	8
bonn 79	14.54	0	12.02	0	13.67	0	11.35	0
bonn 80	5.55	2	5.01	3	5.38	2	5.13	3
bonn 81	30.23	0	7.29	2	26.42	2	24.48	8
bonn 82	32.54	0	21.33	5	39.01	0	34.60	3
bonn 83	7.40	3	5.25	3	13.33	21	9.25	13
bonn 84	7.57	3	5.91	5	6.10	3	6.69	2
bonn 85	12.47	1	9.45	1	29.19	0	15.62	0
bonn 86	7.22	2	16.93	21	23.99	0	14.49	2
bonn 87	12.82	3	12.11	5	33.03	1	12.16	0
bonn 88	6.46	5	6.40	3	8.40	2	7.72	3
bonn 89	5.45	3	4.34	5	5.38	3	4.80	5
bonn 90	4.89	3	4.35	5	5.40	1	4.38	3
bonn 91	10.60	0	12.32	0	10.95	0	11.98	0
bonn 92	4.59	2	4.33	3	5.87	2	5.60	1
bonn 93	8.70	0	7.61	0	10.22	0	43.10	1
bonn 94	6.87	2	5.84	3	9.28	0	7.90	0
bonn 95	12.91	0	12.86	0	18.21	0	20.81	0
bonn 96	6.54	2	16.44	3	20.56	5	15.19	21
bonn 97	3.96	3	3.41	5	5.91	2	4.61	0
bonn 98	8.84	1	7.35	2	14.97	0	9.13	0
bonn 99	4.02	0	3.48	5	5.16	3	4.53	2
Nat 10	35.84	0	38.83	0	51.77	0	64.46	0
Nat 10v	41.08	0	37.74	0	55.08	0	53.03	1
Nat 16b	43.55	0	50.34	1	57.15	0	69.15	0

Overall, the average segmentation error rate achieved varies from 11.4% to 15.6%, depending on the type and depth of transform used. Thus, the segmentation performance of fuzzy K-means is comparable to conventional K-means. This fact is clearly illustrated in the direct comparison with conventional K-means in table 5.10. The average error rate is only slightly lower for fuzzy K-means, with the difference being approximately 1% to 2%. However, the median error rate with fuzzy K-means is much lower, with the difference from conventional K-means being approximately 3% to 4%. This suggests that the fuzzy K-means is more likely to produce better segmentation result than conventional K-means, but it can also have greater variations between results. The performance order

of the various wavelet transforms is exactly the same as for conventional K-means; 2-level DT-CxWT gave the best average segmentation performance, followed by 2-level DWT, 3-level DT-CxWT and 3-level DWT. For 2-texture mosaics, the results favour conventional K-means over fuzzy K-means, with the former producing slightly lower error rates. However, the reverse is true for 5-texture mosaics, with the fuzzy K-means algorithm having lower error rates, by as much as 4% on average.

A disadvantage of the fuzzy K-means algorithm is the extra computational requirements over conventional K-means. In these experiments, a value of 10^{-5} is used for the termination constant ϵ (equation (4.11)). This value was found to be sufficient to produce decent clusterings without requiring too many iterations. However, an average of 120 iterations are necessary for 5-texture mosaics when 3-level DT-CxWT features are used. In comparison, only 30 iterations are needed by conventional K-means on average. On top of this, each iteration of the fuzzy K-means algorithm is slightly more expensive computationally than conventional K-means. Therefore, there is a significant computational penalty associated with using the fuzzy K-means algorithm to cluster the texture features.

Table 5.10. Fuzzy K-means results summary and comparison. The overall mean and median error rates (%) from the fuzzy K-means experiments are shown for different transform types and depths. The 95% confidence intervals for the mean error rates are also listed. Corresponding quantities for conventional K-means are repeated from table 5.6 to facilitate direct comparison.

Transform type/depth	Conventional			Fuzzy		
	Mean	Median	95% C.I.	Mean	Median	95% C.I.
DT-CxWT2	13.90	10.58	(12.13, 15.80)	11.36	7.73	(9.63, 13.10)
DT-CxWT3	15.01	13.07	(13.12, 16.90)	14.93	9.83	(12.60, 17.26)
DWT2	14.79	13.81	(13.12, 16.46)	12.42	9.89	(10.76, 14.09)
DWT3	16.11	15.20	(14.35, 17.88)	15.59	10.95	(13.36, 17.83)

Table 5.11 shows the variation of the optimal Kaiser smoothing parameter value for the fuzzy K-means experiments. As in section 5.3, the optimal β values are found from experiments on the test suite with known ground truths. A similar trend to that from the K-means experiments is observed. There is a heavy concentration of small β values, with optimal β greater than 8 occurring infrequently. There is only a sprinkling of genuinely high values of optimal β. This indicates that the fuzzy K-means algorithm also relies on heavy smoothing on the features to be able to effectively segment the texture mosaics.

Table 5.11. Fuzzy K-means results summary - part 2. The optimal Kaiser window β parameter values for all images, transform types and depths are tallied from the fuzzy K-means experiments to produce the distribution of optimal β.

Transform type/depth	β									
	0	1	2	3	5	8	13	21	34	55
DT-CxWT 2	28	26	26	23	7	1	3	1	0	0
DT-CxWT 3	27	14	23	24	13	5	3	4	1	1
DWT 2	50	24	14	15	5	1	3	1	0	2
DWT 3	43	14	13	17	10	7	6	1	2	2

In summary, the texture segmentation results from the fuzzy K-means algorithm on wavelet-based features supports the findings from the earlier experiments with conventional K-means clustering. That is, the algorithm can segment between two-texture mosaics very well, but has difficulty with the complexities of multi-texture mosaics. In terms of the wavelet-based features, the same conclusions can also be reached as from the K-means experiments, namely that the DT-CxWT is demonstrably better than the DWT, while a lower transform depth of 2 gives better segmentation performance than a depth of 3.

5.5 A Modified K-means Clustering

By their nature, textures are spatially extensive phenomena; it is impossible to define the texture in terms of a single pixel, because that would carry no meaning. When a pixel is said to belong to one texture, it implicitly means the surrounding area around the pixel resembles that particular texture. This fundamental property can be exploited in the clustering algorithm, by considering spatial location information when making the clustering decisions. The rationale is that, if a pixel belongs to one particular texture, then it is highly likely that its immediately neighbouring pixels also belong to the same texture.

The classic K-means clustering algorithm used in the experiments presented in section 5.3, and the closely related fuzzy K-means algorithm used in section 5.4, do not take into account spatial information. The algorithm considers each feature vector exclusively in the feature space, without ever being aware of the spatial location that vector corresponds to. While this is fine for general pattern recognition problems, this leads to relatively poor performance for texture segmentation, where spatial localisation is an inherent part of the problem. From table 5.5, it is evident that some cases have very high error rates when the K-means algorithm is used. The poor performance principally occurred in multi-class input images which contain more than 2 different textures. These are precisely the cases where a pure feature space distance-based algorithm has problems dealing with the extra complexity resulting from having multiple textures in the same image. In contrast, the K-means algorithm proved to be adequate in segmenting 2-texture images. Section 5.2.3 has already predicted the likely difficulties with multiple textures from the feature space distance histogram technique. In light of these difficulties, it is proposed that a possible "locality rule" be built into the existing texture segmentation algorithm in order to successfully segment multi-texture images.

5.5.1 Modification to Existing K-means

For the case of K-means clustering algorithm, a locality rule can be achieved by using the following modified distance in the cluster update step:

$$d(\mathbf{x}_i, \mathbf{x}_j) = (1-\lambda)\sum_k |\mathbf{x}_i(k) - \mathbf{x}_j(k)| + \lambda(|x_i - x_j| + |y_i - y_j|), \quad 0 \leq \lambda < 1 \qquad (5.1)$$

where (x_i, y_i) and (x_j, y_j) are the coordinates of the pixel in the image. These coordinates are normalised with respect to the image dimensions. The first term of equation (5.1) is the Manhattan distance in feature space; this measures the degree of dissimilarity between two feature vectors. The second term is the distance in geometric space, which measures the spatial separation between the locations of the two feature vectors. The parameter λ is freely adjustable to shift the relative emphasis between the two distances. It is constrained to lie between 0 and 1, obviously to keep both distance contributions non-negative.

In general, a good segmentation should consist of a collection of homogeneous regions devoid of small holes in the interior. In this spirit, the motivation behind the use of the modified expression in equation (5.1) becomes clear: its purpose is to provide balance between the twin objectives of high intra-cluster homogeneity and cluster integrity. Occasionally, these two objectives conflict each other, and the magnitude of the free parameter λ can be adjusted to emphasise a particular objective, be it feature similarity or spatial compactness. For this reason, the free parameter λ will be called the *spatial proximity factor*. A low value of λ would emphasise the feature space similarity within the clusters, while a high value would discourage the formation of small, disjointed clusters. This trade-off built into the modified distance measure provides a flexibility that proves to be effective for multi-class segmentation problems. In contrast, the classic K-means simply seeks to minimise the feature space error objective, which amounts to promoting high intra-cluster similarity only. This often produces segmentations with many small residual regions within large homogeneous clusters. These can be due to small irregularities within a texture pattern, but they can also be artifacts from an inappropriate initialisation scheme. Unfortunately, these residual regions can have a devastating effect on the K-means algorithm as it re-iterates the cluster updating process. If there are many erroneous, disjoint regions in one segmentation, then the cluster statistics (most notably class mean) will be skewed for subsequent iterations. In extreme cases, the algorithm is incapable of recovering from incorrect class means, and it converges to a poor segmentation. The introduction of a spatial distance term adds a level of robustness to the algorithm, as it becomes more likely that small disjointed regions will be eliminated as the algorithm iterates.

5.5.2 Expected Behaviour of Modified K-means

Before examining the effects on experimental results with modified K-means clustering, it is important to first consider the behaviour of the clustering algorithm as λ is varied. In particular, it is interesting to examine the likely effects of different values of λ on the segmentation error rate. In the following discussion, it is perhaps simplest to consider the error rate as a function of λ, and examine its variations as λ is increased from 0 to 1. Firstly, when $\lambda = 0$, the modified K-means algorithm reduces to the classic K-means with the Manhattan distance metric. This performs a feature space-only segmentation, and can often produce poor results due to the presence of many residual regions and/or incorrectly merged clusters. Next, consider the effects on error rates for small values of λ. With the increase in λ (from 0), the influence of the spatial distance term increases, and that has the effect of eliminating some of the residual regions, which usually reduces the segmentation error rate. If λ is increased further, it is expected that the size and number of residual regions will gradually decrease, resulting in a corresponding decrease in the error rate.

In addition, as λ is increased further, a second effect is expected to takes place. It has been noticed that one of the reasons for the poor performance of conventional K-means segmentations for multi-class cases is that two or more distinct clusters are incorrectly grouped together, often in the early iterations of the algorithm. The conventional K-means algorithm would be unlikely to recover from the mixing of means that results from such a situation, resulting in highly erroneous clusterings. This phenomenon is usually caused by poor initialisation, and is a difficult problem to avoid. However, with the introduction of a spatial distance term, two incorrectly merged clusters can be split during the modified K-means process if the value of λ is sufficiently large to discourage the inappropriate combining of two or more separate clusters. In a sense, the spatial term has the ability to force the algorithm to scrutinise all the clusters and check whether there are actually incorrectly merged clusters, due to the penalty imposed by a large spatial distance term. When the value of λ is increased to a point where this effect occurs, the clustering result is dramatically improved, as evidenced by a large decrease in the segmentation error rate. The value of λ that produces this sudden improvement is called a *lock-on* point, in lack of a better name. It is as though, beginning with this value of λ, the modified K-means algorithm is locked onto the correct clustering. As λ is increased

beyond the lock-on value, neither the segmentation results nor the feature space distortion measures change greatly. However, if λ is increased indefinitely, it will eventually reach a point when the spatial term dominates the feature distance term, and the segmentation algorithm fails. In the extreme case, the segmentation process becomes entirely dependent on the spatial coordinates of features, that is, the locations of the pixels. The result of this is a simple partition of the rectangular image with maximal geometric separation among the regions, thus, ceasing to be a segmentation based on texture information. Obviously, that will produce meaningless results in the current context, and it presents a genuine danger in applying the modified K-means method. The value of λ for which the spatial term begins to dominate the clustering process is called the *breakdown* point. At this point, the error rate plot would exhibit a sudden, large increase that reflects the failure of the algorithm. In the general case of an unsupervised segmentation, where the ground truth is not known, the error rate information would not be available for this monitoring method to apply. A possible method to handle this difficulty will be presented in section 5.5.7.

It was observed in experiments that the error rate plots against values of λ usually adhere to the expected trend as described above. Figure 5.10 illustrates this for the 5-texture mosaic "Nat 5c" from the suite of images used in the experiments. It is noticed the segmentation error rates are sometimes quite sensitive to changes in λ near certain values. That is, the error rate, as a function of λ, is not smooth. In practice, the limit of useful values of λ (i.e. before breakdown occurs) are usually much less than 0.5. Obviously, the exact value varies from case to case, and methods to estimate the useful values of λ are very desirable in the application of modified K-means. This topic is further discussed in section 5.5.7.

5.5.3 Distance Histograms for Modified K-means

The distance histogram technique has been found to be a useful indicator in estimating the separability of a feature set in section 5.2.3. It was particularly effective for two-texture images. It would be interesting to examine the effects of the spatial proximity factor on the separability of the feature set for the modified K-means clustering. To accomplish this, it only requires applying equation (5.1) while calculating the modified distance between a pair of feature vectors. Since the modified K-means algorithm was

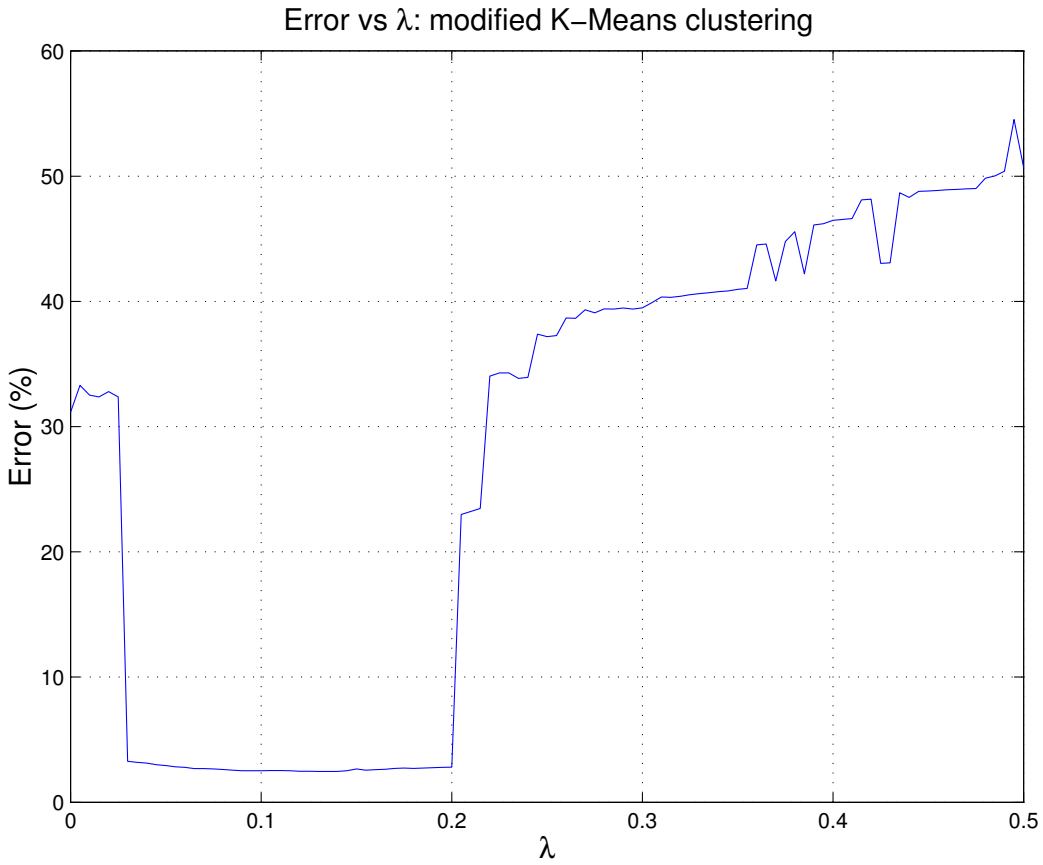
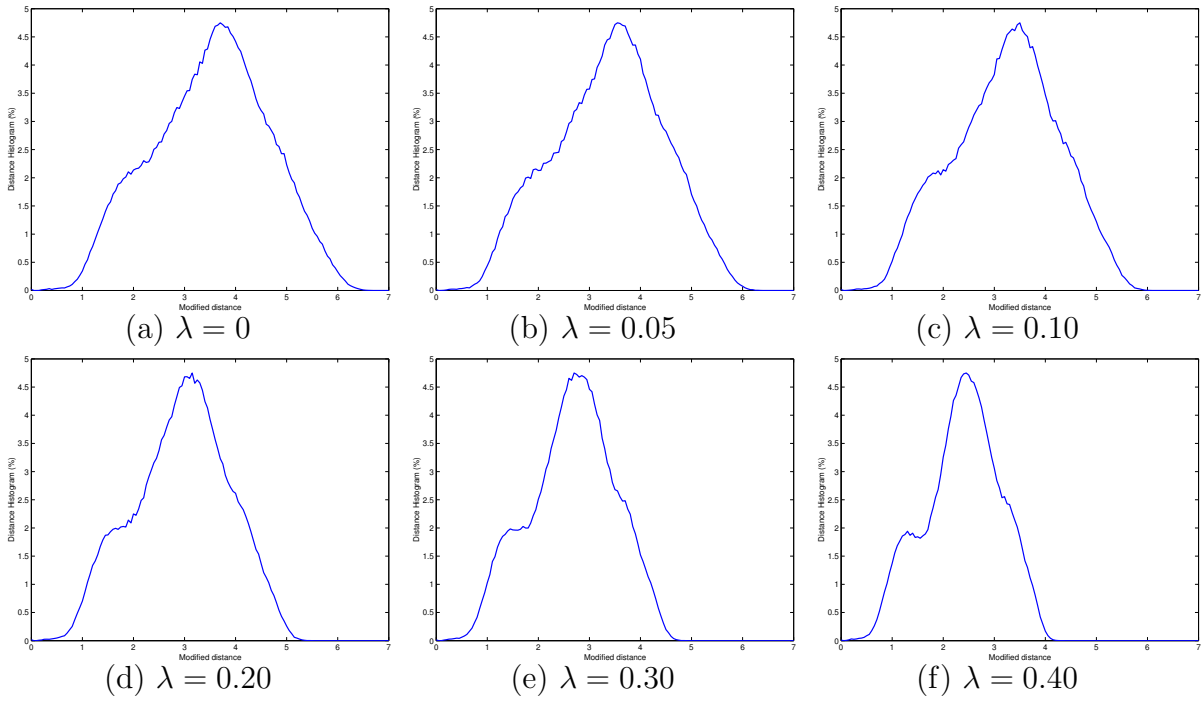


Figure 5.10. Error rate vs λ for modified K-means clustering. Input image used is "Nat 5c", feature extraction parameters: depth 3 DT-CxWT transform using Kingsbury filter pair, Kaiser smoothing window parameter $\beta = 3$, random cluster initialisation.

designed to assist multi-class clustering, there is little purpose in computing the modified distance histograms for 2-texture mosaics. On the other hand, mosaics with many textures are very complex, and their distance histograms get quite complicated, making it difficult to ascertain the value in examining their histograms. Therefore, 5-texture mosaics are considered. They are multi-class without being overly complicated, and their histograms still exhibit several clear characteristics.

Figures 5.11(a)-(f) illustrate the modified distance histograms of the input image Nat 5c for a range of different λ parameter values. It is clear that the modified distance does not significantly alter the general shape of the curve for small values of λ. In this case, there is no noticeable difference in the histograms for $\lambda < 0.1$. For larger values of λ, the intra- and inter-cluster peaks, or in this case, the knee (intra-cluster) and the main peak (inter-cluster), are brought closer to each other. Despite the clear existence of a



(a) $\lambda = 0$ (b) $\lambda = 0.05$ (c) $\lambda = 0.10$

(d) $\lambda = 0.20$ (e) $\lambda = 0.30$ (f) $\lambda = 0.40$

Figure 5.11. Modified distance histogram for five-texture mosaic, Nat 5c, for different values of λ. The style of these plots are the same as in figure 5.3. Feature extraction parameters: 3-level DT-CxWT transform, Kingsbury filter pair, Kaiser smoothing window with $\beta = 3$, sigmoidal non-linearity.

"twin-peaks" characteristics when $\lambda = 0.4$, the fact that the peaks are closer together makes distinguishing between intra- and inter-cluster pairs more difficult. The modified K-means clustering algorithm will have more problems distinguishing between intra- and inter-cluster pairs, causing the poor performance at large values of λ. Figures 5.12(a)-(f) provides further illustration of this phenomenon. It is evident that, as λ increases, the histograms for inter-cluster pairs (drawn as solid lines) shift leftwards. While the intra-cluster histograms also shift left, the net differences between them, in modified distance space, are reduced. Even more revealing are the distance histograms involving each pair of clusters. These are separated into groups by horizontal dotted lines in the plots of figure 5.12: all five histograms involving cluster 1 (clusters are randomly numbered) are grouped together, the other four histograms involving cluster 2 are shown above those involving cluster 1, and the pattern is repeated. The class pair labels on the vertical axes identify each histogram. Studying these histograms explains the shape of the histograms in figure 5.11. All individual cluster-pair histograms are compressed in width, and shifted left. The compression in the widths of the peaks lead to a seeming improvement in the

overall histogram, since narrower cluster-pair histograms superimpose to produce a more clear-cut, "twin-peaks" characteristic. However, the left shifting of the individual peaks also means that the overall intra- and inter-cluster peaks are much closer together, and this has an adverse effect on the segmentation performance. It has been noticed that when the intra- and inter-cluster peaks get significantly close to each other, the overall segmentation performance is reduced. That is, this takes place despite the clear bi-modal appearance of the overall histogram. In the case illustrated by figure 5.12, this degradation in performance takes place for λ greater than 0.15.

5.5.4 The Distortion Measure

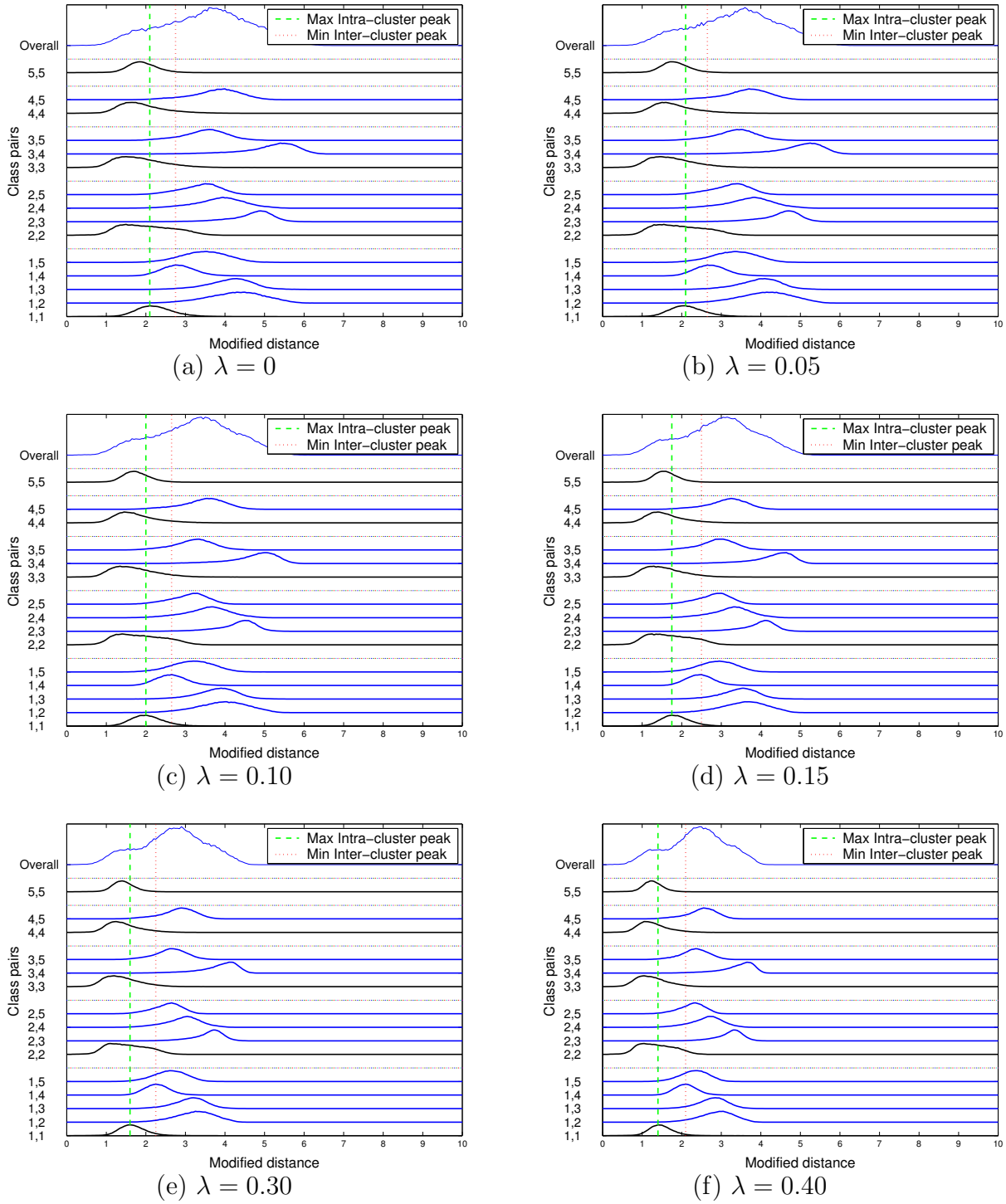
In studying the behaviour of the modified K-means algorithm for variations in λ, it is useful to monitor the average distance of a pixel from its cluster centre, or *distortion*, in both the feature and geometric spaces. These quantities are defined as follows:

$$d_{fts} = \frac{1}{N}\sum_{i,k} |\mathbf{x}_i(k) - \boldsymbol{\mu}_i(k)| \tag{5.2}$$

$$d_{geo} = \frac{1}{N}\sum_{i} (|x_i - \mu_{xi}| + |y_i - \mu_{yi}|) \tag{5.3}$$

where $\mathbf{x}_i$ is a feature vector, x_i, y_i are the spatial coordinates of the vector; $\boldsymbol{\mu}_i$ is the mean feature vector of the cluster to which $\mathbf{x}_i$ belongs, μ_{xi}, μ_{yi} are the mean spatial coordinates of the same cluster. The sums are computed over all N pixels in the feature image. These quantities are computed after the convergence of the modified K-means algorithm, once the final cluster centres are known.

It was stated in the previous section that as λ is increased from 0, clusters produced by the algorithm tend to become more compact, due to the greater effects of the spatial distance term. Hence, it is reasonable to expect the average geometric distortion, d_{geo}, to fall monotonically when considered as a function of λ. As λ is increased from zero, d_{geo} experiences a rapid decrease in all the cases, which is a result of the increased emphasis on geometric space clustering. The decrease slows and stabilises as the algorithm produces the best result, until breakdown is reached, at which point d_{geo} decreases sharply as the algorithm degenerates to being completely spatial. The feature space distortion, d_{fts} rises with increase in λ. Initially, for small values of λ, d_{fts} rises slowly and steadily. When λ reaches breakdown, d_{fts} experiences a sudden increase.

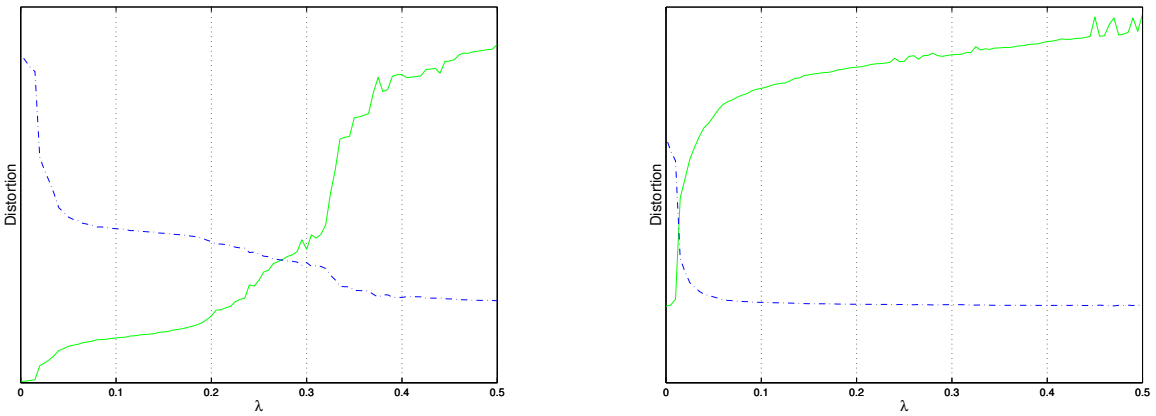


(a) $\lambda = 0$ (b) $\lambda = 0.05$

(c) $\lambda = 0.10$ (d) $\lambda = 0.15$

(e) $\lambda = 0.30$ (f) $\lambda = 0.40$

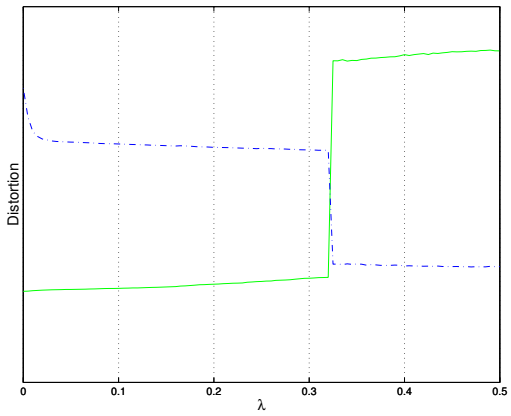
Figure 5.12. Modified distance histograms for five-texture mosaic, Nat 5c, for different values of λ. Histograms for all pairs of clusters are illustrated; the plots are in the same style as figure 5.3. Feature extraction parameters: 3-level DT-CxWT transform, Kingsbury filter pair, Kaiser smoothing window with $\beta = 3$, sigmoidal non-linearity.



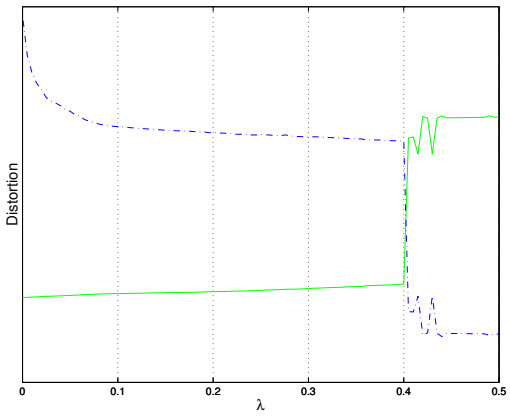
(a) My 5a (b) My 5b

Figure 5.13. Distortion plots. The feature and geometric distortions are plotted against λ. d_{fts} are plotted as green solid lines, while d_{geo} are plotted as blue dotted lines. Feature extracted with depth 3 DT-CxWT, Kingsbury filters, Kaiser window smoothing. Distortions are calculated from the best result among 25 different runs of modified K-means with random initialisations.

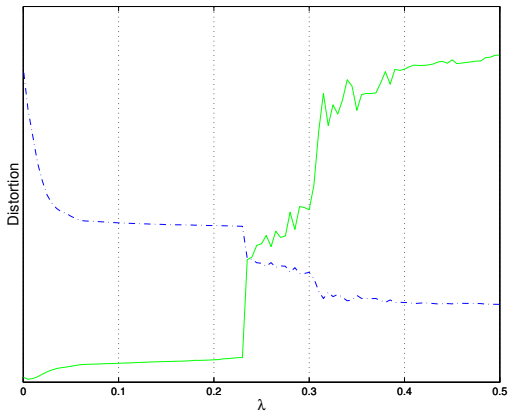
The general characteristics described above can be observed for multi-texture mosaics. For the simple 2-texture mosaic images, the phenomena mentioned above are not readily apparent, since the lock-on and breakdown phenomena do not occur. On the other hand, modified K-means is not necessary for these simple images, for conventional K-means has been shown to be adequate to deal with these cases. For selected 5-texture mosaics, the feature and geometric space distortions are computed and plotted against λ in figures 5.13 and 5.14. These figures clearly show the existence of the breakdown points for the "Nat" series of mosaics; these are points where the feature space distortion greatly increases. In contrast, the lock-on points are much harder to locate, if unique ones even exist at all. Instead, there is usually a continuous decrease of spatial distortion with a steady feature space distortion. The lock-on and breakdown points are not so apparent for the images "My 5a" and "My 5b". For some complicated cases with more than 5 textures, there appear several distinct lock-on and breakdown points on the distortion plots. This can be explained using the same reasoning as described above; the multiple lock-on and breakdown points are concerned with different pairs of clusters separating or merging together. With many different pairs of texture clusters, the merging of any particular cluster pair may well take place at a value of λ below the value where separation of another cluster takes place. This is a result of the greatly increased complexity when there are many different combinations of texture pairs in an image.



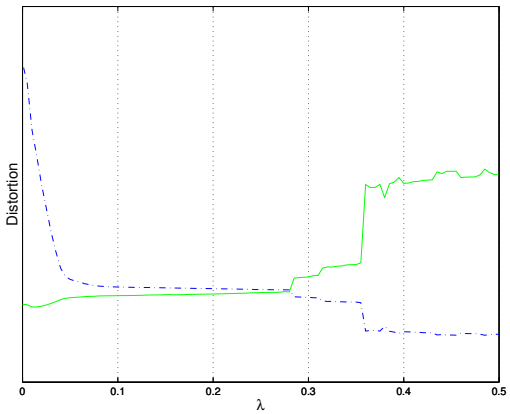
(a) Nat 5b



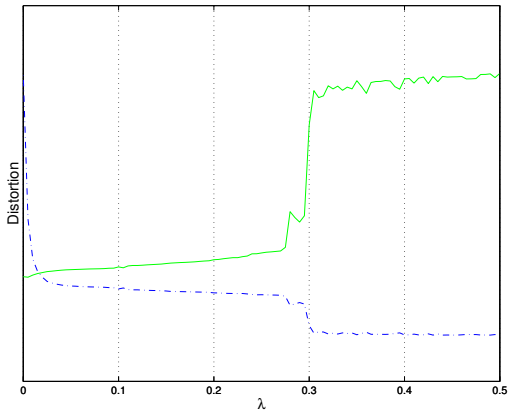
(b) Nat 5c



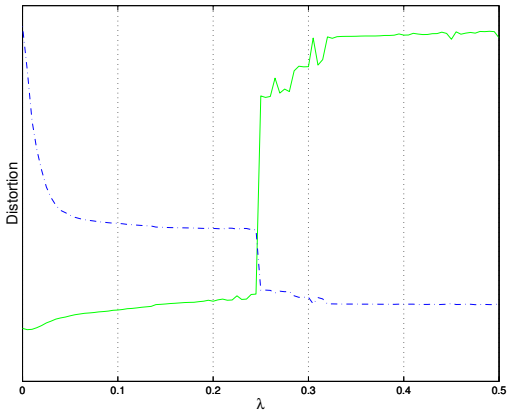
(c) Nat 5m



(d) Nat 5v



(e) Nat 5v2



(f) Nat 5v3

Figure 5.14. Distortion plots (continued). d_{fts} are plotted as green solid lines, while d_{geo} are plotted as blue dotted lines. Feature extraction parameters are the same as for figure 5.13.

5.5.5 Results from Experiments

This section presents the results obtained from segmentation experiments obtained with the modified K-means clustering method. Table 5.12 shows the segmentation results obtained from different wavelet features, for the entire suite of 115 input images. These tables are condensed as much as possible for brevity; a full collection of result tables can be found in Appendix B. These results provide direct comparisons between different transform types, depths and smoothing methods. The format of presentation is identical to table 5.5, so as to allow easy comparison between the conventional and modified K-means results.

Table 5.12. Condensed results of Modified K-Means experiments. The best error rates (%) are shown for each input image. This condensed table of results only shows the best case; a full set of results for all values of β is found in Appendix B.

	C2M	D2M	C3M	D3M	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
					(%)	β	(%)	β	(%)	β	(%)	β
D4-D84	0.09	0.15	0.11	0.11	0.00	2	0.01	2	0.04	3	0.06	3
D5-D92	0.34	0.41	0.34	0.33	0.43	55	0.51	55	0.35	13	0.38	0
D8-D84	0.16	0.33	0.06	0.37	0.28	55	0.31	55	0.13	55	0.24	55
D12-D17	0.34	0.41	0.16	0.26	0.17	55	0.35	0	0.15	0	0.28	3
My 5a	4.49	7.37	6.49	12.71	3.90	8	4.97	0	4.66	13	4.38	2
My 5b	2.33	2.67	2.28	2.74	2.03	34	2.43	13	2.14	55	2.70	55
Nat 5b	2.45	4.06	2.17	5.26	2.76	5	3.65	3	2.12	3	4.14	2
Nat 5c	2.00	3.62	2.01	3.35	3.14	13	4.71	8	2.46	8	4.09	3
Nat 5m	2.47	4.47	2.44	4.48	3.28	5	4.28	8	2.97	8	4.06	2
Nat 5v2	5.20	8.51	5.30	10.19	4.71	2	6.20	0	4.25	3	6.45	2
Nat 5v3	5.34	6.83	5.62	9.05	5.18	3	5.08	3	5.11	5	4.75	2
Nat 5v	4.93	4.88	7.83	5.79	3.38	8	3.39	8	3.44	5	3.24	2
bonn 00	2.03	2.29	2.59	3.01	2.42	5	2.47	13	2.25	34	2.27	8
bonn 01	3.03	4.41	4.12	4.68	2.55	8	3.13	2	2.99	8	3.06	5
bonn 02	2.31	3.16	2.40	4.32	2.47	13	2.80	13	2.75	21	3.20	21
bonn 03	2.21	3.09	2.41	3.96	1.78	5	2.56	3	2.06	34	2.80	8
bonn 04	1.73	2.84	2.00	21.46	2.02	55	2.45	34	2.59	55	3.04	34
bonn 05	3.72	3.63	2.47	3.38	2.56	13	4.05	3	3.24	8	3.36	13
bonn 06	2.28	3.51	2.17	3.64	2.09	3	2.30	5	2.29	8	2.95	8
bonn 07	0.90	1.82	1.43	2.74	1.93	8	2.40	21	1.28	55	2.17	34
bonn 08	21.89	23.62	4.29	26.41	4.24	2	4.69	0	6.62	8	4.74	2
bonn 09	11.24	4.03	21.01	4.50	3.92	3	4.00	13	3.82	3	5.23	8
bonn 10	2.59	2.96	22.55	2.80	2.95	34	3.51	21	3.20	55	3.48	0
bonn 11	1.75	2.09	2.31	2.74	2.00	13	2.07	8	2.35	21	2.64	13
bonn 12	2.39	2.44	2.36	4.27	2.42	8	2.37	8	2.72	8	2.78	3
bonn 13	2.62	5.67	2.40	3.62	1.97	3	2.44	3	1.80	5	2.25	2
bonn 14	2.53	3.17	2.08	3.24	2.23	13	2.64	13	1.62	13	2.14	3
bonn 15	1.36	2.13	1.84	2.75	1.35	21	1.89	5	2.06	5	1.52	3
bonn 16	1.18	1.64	1.21	1.87	1.70	34	1.72	21	1.59	55	1.50	34
bonn 17	1.64	2.11	1.90	2.78	1.98	55	2.20	13	2.01	34	2.33	21
bonn 18	3.96	5.77	4.15	6.12	3.86	5	5.10	0	4.08	21	4.19	3
bonn 19	3.06	9.67	2.55	4.88	2.47	3	3.07	3	2.29	8	2.34	8
bonn 20	3.84	3.58	5.98	4.47	3.84	34	3.69	5	3.90	21	4.14	3
bonn 21	3.63	3.90	4.21	4.25	3.60	13	3.93	13	4.02	13	4.14	8
bonn 22	9.17	12.65	9.67	13.58	6.21	2	8.01	1	5.87	0	6.84	0
bonn 23	1.40	2.12	2.67	3.63	1.81	34	1.73	8	2.64	21	2.42	8
bonn 24	2.40	2.98	2.80	3.06	2.15	21	2.18	21	2.35	21	2.26	21
bonn 25	2.76	2.94	2.81	3.34	2.17	13	2.64	8	2.62	5	3.27	5

	C2M	D2M	C3M	D3M	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
					(%)	β	(%)	β	(%)	β	(%)	β
bonn 26	1.76	2.78	21.06	4.54	1.82	13	1.79	13	2.36	21	2.27	13
bonn 27	3.89	4.04	4.25	3.43	4.08	5	4.38	5	4.14	21	3.91	55
bonn 28	7.17	4.97	4.53	5.32	5.00	1	4.68	13	3.30	13	3.99	5
bonn 29	1.64	2.52	1.66	3.27	1.65	34	2.00	21	1.31	13	2.16	21
bonn 30	2.14	3.52	2.91	3.43	2.86	21	3.12	21	2.58	21	2.69	21
bonn 31	3.14	24.39	3.58	4.79	2.49	5	2.95	5	2.93	5	3.66	1
bonn 32	4.45	4.27	4.67	3.96	4.24	34	3.96	13	4.32	13	3.88	13
bonn 33	2.57	2.45	22.50	3.15	2.15	34	2.40	34	2.23	8	2.21	34
bonn 34	3.25	5.15	2.97	6.05	3.49	5	5.49	8	3.94	21	5.22	8
bonn 35	2.06	2.48	1.86	3.02	1.96	13	2.13	8	2.19	21	2.12	8
bonn 36	3.56	3.64	2.89	26.48	2.40	3	2.93	8	2.26	5	1.99	5
bonn 37	1.48	2.38	1.55	2.27	1.50	21	2.04	13	1.57	8	1.59	8
bonn 38	2.48	2.56	2.51	21.15	2.76	21	2.95	21	2.12	8	2.81	13
bonn 39	1.31	2.23	1.45	2.39	1.33	34	1.69	8	1.56	34	1.71	8
bonn 40	1.31	1.81	1.53	2.15	1.81	55	1.87	34	1.60	34	1.85	13
bonn 41	3.85	5.82	3.95	5.68	2.83	5	3.49	3	2.89	8	3.86	2
bonn 42	1.36	2.61	1.50	3.92	1.56	13	2.24	3	1.18	21	1.90	5
bonn 43	3.65	5.48	3.42	4.60	3.33	5	3.91	13	2.62	3	3.08	2
bonn 44	2.60	3.97	2.52	4.88	2.17	21	3.02	21	1.95	8	2.48	8
bonn 45	6.28	9.20	6.27	8.73	5.87	5	6.92	3	5.46	8	6.71	8
bonn 46	3.31	3.69	3.38	4.91	4.68	5	3.99	21	3.45	13	4.08	21
bonn 47	1.86	3.90	2.47	3.41	2.20	21	3.73	13	1.75	21	2.62	8
bonn 48	22.31	3.11	2.72	3.27	1.71	13	1.97	5	1.76	0	1.61	5
bonn 49	2.39	2.48	2.56	3.22	2.55	8	3.11	2	2.44	8	3.11	2
bonn 50	2.20	1.86	2.67	2.69	2.57	55	2.47	34	3.53	55	3.58	34
bonn 51	1.51	2.01	1.70	1.87	1.75	34	1.92	8	1.53	21	1.59	8
bonn 52	2.25	2.91	2.63	3.18	1.84	13	2.11	8	2.29	21	2.23	5
bonn 53	3.49	2.93	3.47	3.53	2.61	13	2.76	13	2.95	34	3.23	13
bonn 54	2.28	3.02	2.40	3.41	2.63	21	3.03	8	2.28	34	2.93	8
bonn 55	1.61	2.00	1.84	2.25	2.23	34	2.42	21	2.12	34	1.97	21
bonn 56	5.79	11.83	4.95	6.14	4.48	5	4.49	8	3.57	13	5.00	13
bonn 57	3.45	5.63	3.81	6.04	3.14	2	5.16	2	3.50	3	4.35	1
bonn 58	2.26	3.19	2.08	3.47	2.72	8	2.98	13	2.11	34	2.64	13
bonn 59	1.99	2.79	1.93	2.85	2.38	8	2.84	8	1.81	13	1.98	5
bonn 60	11.74	3.27	21.28	4.36	2.67	2	2.95	3	3.12	2	2.95	2
bonn 61	2.08	2.56	20.10	3.26	2.10	8	2.30	8	2.37	34	2.76	8
bonn 62	6.02	3.86	21.84	8.92	2.93	5	3.19	2	3.04	5	2.93	2
bonn 63	5.06	10.84	2.28	6.10	1.78	21	2.49	8	2.71	1	2.09	13
bonn 64	3.50	4.91	4.70	4.93	4.66	34	5.55	13	4.70	13	5.55	3
bonn 65	1.91	3.94	2.19	22.83	1.62	21	3.99	21	2.48	34	2.98	5
bonn 66	1.67	2.50	1.99	3.22	1.58	8	1.84	8	1.57	21	1.65	8
bonn 67	2.84	2.95	2.59	19.27	2.90	21	3.13	13	2.18	21	2.63	13
bonn 68	2.00	21.79	2.29	3.10	2.78	34	3.09	13	2.17	8	2.65	55
bonn 69	3.13	4.53	2.48	4.75	2.79	5	3.70	3	2.50	5	2.91	3
bonn 70	1.75	1.89	2.30	2.67	1.73	21	1.64	8	2.45	13	1.99	13
bonn 71	2.38	3.52	2.65	3.75	2.63	21	2.97	5	2.09	13	2.93	5
bonn 72	2.20	3.05	2.11	2.97	2.75	13	3.34	5	2.73	34	2.86	13
bonn 73	3.86	4.43	5.03	5.07	2.75	13	3.66	2	2.81	1	3.62	8
bonn 74	2.51	3.66	2.51	3.88	2.04	1	2.16	1	2.09	5	2.35	3
bonn 75	1.93	2.75	2.37	3.28	2.00	13	2.82	8	1.84	34	2.26	8
bonn 76	3.75	3.16	2.78	3.53	2.48	5	2.91	5	2.36	8	2.95	5
bonn 77	5.16	6.73	16.96	10.21	4.10	8	4.25	1	3.86	8	5.12	8
bonn 78	1.97	2.83	2.21	3.03	2.53	5	3.07	5	2.27	21	2.97	13
bonn 79	2.97	4.35	3.00	3.60	2.80	13	3.24	5	3.08	8	3.33	8
bonn 80	1.77	2.87	2.02	2.96	1.74	21	1.87	5	2.22	13	2.29	13
bonn 81	2.97	23.19	2.49	4.49	1.88	5	2.40	13	1.84	5	2.27	8
bonn 82	22.97	2.77	2.40	2.97	1.95	21	2.16	21	4.57	55	2.25	5
bonn 83	2.07	3.16	2.55	3.62	2.24	55	2.80	34	2.15	13	2.80	34
bonn 84	2.03	2.92	2.15	2.92	2.09	21	2.69	21	1.73	13	2.26	13
bonn 85	2.58	3.56	2.76	4.61	2.76	8	3.00	3	3.04	5	3.05	0
bonn 86	2.34	20.78	2.88	4.26	1.88	21	2.45	3	2.61	21	2.94	21
bonn 87	5.03	6.76	4.55	5.94	4.39	8	4.83	8	4.67	34	5.31	21
bonn 88	3.25	3.47	3.53	20.27	3.17	34	2.87	13	3.16	34	3.56	8
bonn 89	2.22	3.27	2.05	3.13	2.47	13	2.89	8	2.48	8	2.78	13
bonn 90	2.36	2.81	2.34	3.08	2.21	8	2.75	13	1.80	13	2.65	5

	C2M	D2M	C3M	D3M	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
					(%)	β	(%)	β	(%)	β	(%)	β
bonn 91	2.55	20.46	3.21	4.71	2.67	8	2.83	5	3.42	8	3.08	8
bonn 92	1.85	3.22	1.80	2.84	1.82	13	2.19	5	1.67	34	2.33	21
bonn 93	3.28	4.38	2.67	4.16	3.85	21	4.95	34	2.76	8	3.87	13
bonn 94	2.07	2.98	2.32	4.26	2.12	5	2.45	3	1.96	13	2.87	5
bonn 95	2.75	4.15	3.55	4.35	3.68	21	4.27	21	4.32	34	4.01	21
bonn 96	2.44	3.31	2.70	21.26	2.39	5	2.61	5	2.61	21	2.95	2
bonn 97	1.39	2.15	1.57	2.33	1.37	13	1.74	21	1.13	13	1.50	3
bonn 98	2.08	2.88	2.06	2.69	2.01	13	2.03	8	1.97	8	1.84	5
bonn 99	2.19	21.39	2.26	7.61	2.40	5	2.58	8	1.85	21	2.39	21
Nat 10	2.41	2.94	1.56	2.61	2.09	55	2.82	34	1.45	34	2.20	34
Nat 10v	1.15	1.91	1.33	1.82	1.04	55	1.42	13	1.10	55	1.61	13
Nat 16b	5.88	7.06	5.16	6.33	5.44	3	5.96	3	5.11	5	5.56	2

An executive summary of results is presented in table 5.13, in identical layout as table 5.6. This smaller table provides a direct comparison between different transform types, depths and smoothing methods. The first observation is that the modified K-means method produced much better segmentation error rates than the conventional K-means method. As with the conventional K-means case, the Kaiser window smoothing method again demonstrates its superiority over the median filter smoothing method. However, the differences between these two methods are greatly reduced, primarily due to the absolute improvements in all error rates with the modified K-means method. In a reversal to the conventional K-means case, table 5.13 shows that a DT-CxWT depth 3 transform performs better, on average, than the DWT depth 2 transform. The order of different transform types and depths for modified K-means clustering, in decreasing order of performance, is: DT-CxWT depth 2, DT-CxWT depth 3, DWT depth 2 and DWT depth 3. This shows that, when the clustering algorithm is modified especially for multi-texture segmentation, the DT-CxWT transforms produce features that are better for distinguishing between different textures. The superior directionality and shift-invariance of the DT-CxWT is favoured by the modified K-means algorithm.

Table 5.13. Modified K-means results executive summary. The overall mean and median error rates (%) for the different transform types, depths and smoothing techniques are listed in this table. In addition, the 95% confidence interval for the means are shown.

Transform type/depth	Median Filtering			Kaiser Filtering		
	Mean	Median	95% C.I.	Mean	Median	95% C.I.
DT-CxWT2	3.41	2.41	(2.75, 4.08)	2.61	2.42	(2.40, 2.81)
DT-CxWT3	4.10	2.52	(3.20, 4.99)	2.63	2.37	(2.42, 2.84)
DWT2	4.86	3.22	(3.96, 5.77)	3.06	2.84	(2.82, 3.30)
DWT3	5.39	3.63	(4.43, 6.34)	2.98	2.81	(2.75, 3.21)

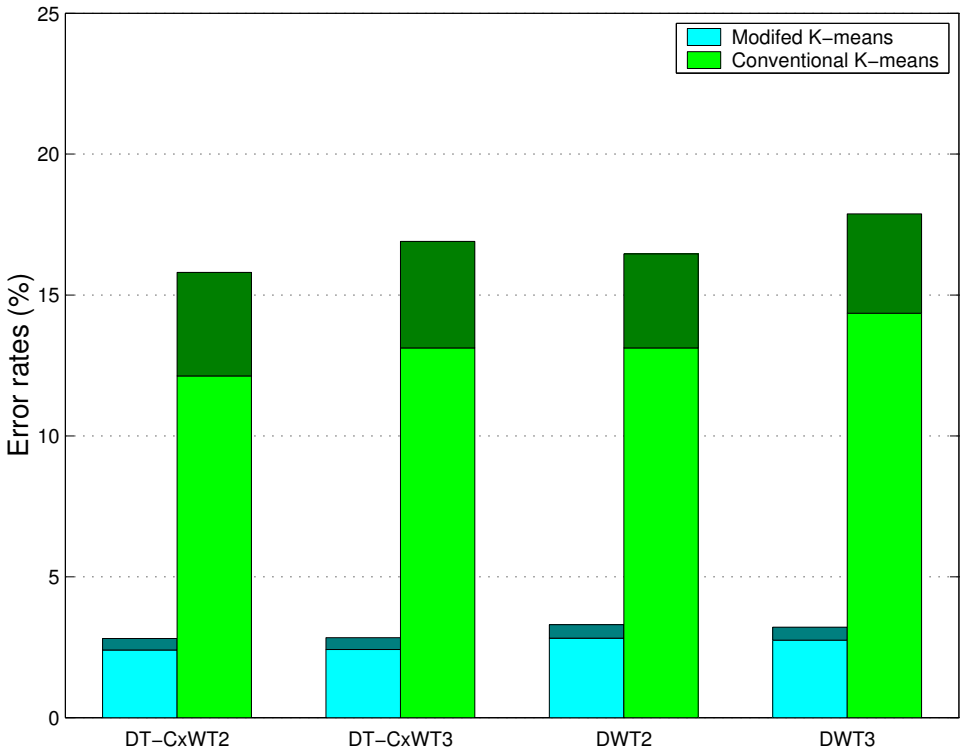


Figure 5.15. Performance comparison between modified and conventional K-means. The height of each bar reflects the average error rates; the 95% confidence intervals are coloured in dark. From this chart, the superiority of modified over conventional K-means in texture mosaic segmentation performance is clearly apparent.

While the segmentation performance of conventional K-means for 2-texture mosaics is quite good, the modified K-means method shows even better performance. In particular, for image D4-D84, perfect segmentation was achieved with a DT-CxWT depth 2 transform and Kaiser window smoothing, while maintaining near-perfect segmentations for other transform types and depths. The error rates are, without exception, below 1%, for all 2-texture mosaics, using any combination of transform type, depth and smoothing method. It should also be mentioned that the error rates for the 2-texture cases do not fluctuate greatly between different β values (in the Kaiser smoothing case). In general, the performance of the modified K-means is nearly perfect for the 2-texture mosaics - the overall error rate across the set of 2-texture mosaics is 0.17% for a level 3 DT-CxWT transform. However, this result must be treated with care. The individual regions in the 2-texture mosaics are simple rectangles separated by a straight boundary. This is precisely one of the optimal solutions for a degenerate, spatial-dominated modified K-means.

That is, a high value of λ would artificially improve the segmentation error rates. However, the main concern with modified K-means is not with segmenting 2-texture mosaics; conventional K-means was found to be capable of that task.

The emphasis of modified K-means is on the segmentation of multi-texture mosaics. These have been difficult for the conventional K-means algorithm, as shown in section 5.3. Unsurprisingly, the overall segmentation performance of the modified K-means algorithm shows vast improvements over conventional K-means for multi-texture mosaics. The error rates are quite stable among the many multi-texture cases, despite the wide range of constituent textures present in these mosaics. With a handful of exceptions, the segmentation error rate is below 5% for multi-texture mosaics. The maximum error rate for the 5-texture mosaics never exceeds 7% with Kaiser smoothing windows. Overall, when averaged over the set of all 5-texture mosaics, the error rate is 2.69% for 2-level DT-CxWT features, and 2.72% for 3-level DT-CxWT. These rates are marginally higher for DWT features. With median filter smoothing, error rates are also improved when modified K-means is used, but they remain over 20% for some cases. Overall, the average error rate with median filtering is 3.54% for 2-level DT-CxWT features and 4.28% for 3-level DT-CxWT. Interestingly, the performance of modified K-means is better for the two 10-texture mosaics than for most of the 5-texture ones. This is suspected to be due to the rather regular texture region boundaries present in these mosaics. The fact that only simple regions exist has made the clustering much easier for the modified K-means. However, with only two such cases, it is difficult to ascertain the performance of modified K-means for mosaics with 10 textures. The single 16-texture mosaic in the experiments yielded satisfactory segmentation accuracy, with an error rate of approximately 5%. However, this is significantly more than the average error rate for 5-texture segmentations.

Table 5.14. Modified K-means results executive summary - part 2. The optimal Kaiser window β parameter values for all images, transform types and depths are tallied from the modified K-means experiments to produce the distribution of optimal β.

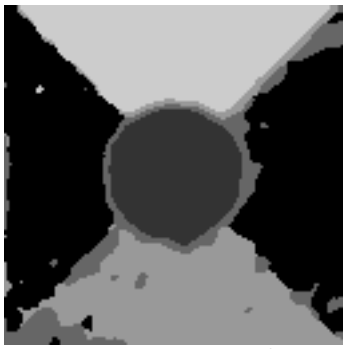
Transform type/depth	β									
	0	1	2	3	5	8	13	21	34	55
DT-CxWT2	0	2	6	7	21	15	21	19	14	10
DT-CxWT3	3	2	1	6	13	23	19	21	18	9
DWT2	5	3	6	15	15	25	22	15	7	2
DWT3	4	2	15	12	15	25	19	12	7	4

Table 5.14 examines the segmentation performance for various values of β with modified K-means. It shows a significant change from the results obtained with conventional K-means. Recall that small values of β performed best with the conventional K-means experiments. This implied that relatively large smoothing windows are necessary to achieve sufficient smoothing to work with conventional K-means. However, the situation is different with modified K-means clustering. The introduction of a spatial distance term discourages the formation of small spatial regions. An effect of this is to reduce the need for heavy smoothing of the features with large windows. The smoothing windows can be smaller as a result; this is achieved with choosing larger values of β in the experiments. There is an important consequence: recall that one of the primary roles of a smoothing window is matching the texture primitive size in the textures, to effectively average the subband energies over the entire primitive. With conventional K-means, this objective is sometimes overshadowed by the need to use large smoothing windows to achieve sufficient spatial smoothness for the clustering to be effective. However, with modified K-means, the inherent ability to form significant spatial clusters more effectively means that the smoothing windows can be chosen more freely, to adapt to the description of the textures by matching texture primitive size. There is a lesser need to use large windows purely to assist the clustering algorithm. This is a fundamental advantage of the modified K-means.

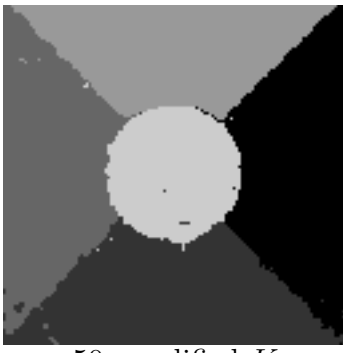
Table 5.14 justifies the aforementioned trend, with quite a wide spread of values for the optimal β across the suites of input images. Compared to table 5.8, medium values of β are more dominant, as well as a good proportion being high values. Examples of this effect are the input cases "bonn 34" and "bonn 50". These mosaics are typified by the fine-grained nature of their constituent textures. In other words, the texture primitives are small, and are more favourably smoothed using a narrower Kaiser window. From the conventional K-means experiments, the segmentation performance is mediocre at best, with error rates between 10% and 20%. With Kaiser window smoothing, the optimal values of β are low, typically between 0 and 5 (see table 5.8). The results are much better with modified K-means, with the error rates are reduced to between 2% to 5%, achieved with optimal β values between 8 and 34. This is an exemplary illustration of one of the advantages of modified K-means over conventional K-means. The best segmentations for these cases, with both conventional and modified K-means, are illustrated in figure 5.16. It is clearly evident that the higher β values used for the modified K-means lead directly to much better segmentations.

(a) bonn 34, conventional K-means, $\beta = 1$

(b) bonn 34, modified K-means, $\beta = 5$



(c) bonn 50, conventional K-means, $\beta = 5$



(d) bonn 50, modified K-means, $\beta = 55$

Figure 5.16. Comparison between conventional and modified K-means for fine-grained texture segmentation. For comparison purposes, a common DT-CxWT depth 2 transform with Kaiser smoothing is used for all segmentation cases shown here. The relevant error rates can be found in tables 5.5 and 5.12.

The modified K-means algorithm introduces an extra term in the distance metric, which would nominally increase the computational load of the algorithm. However, a main reason for introduction of the spatial distance term is to speed up the convergence process. Therefore, two counter-acting factors affect the computational speed of modified K-means compared with conventional K-means: the longer per-iteration load of modified K-means against (ideally) fewer iterations. It would be meaningful to compare the run times of the two different algorithms to segment the same data. Table 5.15 lists the run times and number of iterations from the experiments. Both K-means algorithms share the same code base, and built using the same compiler settings. The experiments were ran on the same machine. For direct comparison, only data taken from the 5-texture mosaic experiments are used in calculating the computation costs. By far the most number of experiments were performed on 5-texture mosaics, and so the measurements obtained

from those are more reliable. For example, the fifth column of table 5.15 contain averages from all the DT-CxWT2 depth 2, Kaiser windows experiments. This has a total of $108 \times 10 = 1080$ experiments, each ran over 100 different values of λ.

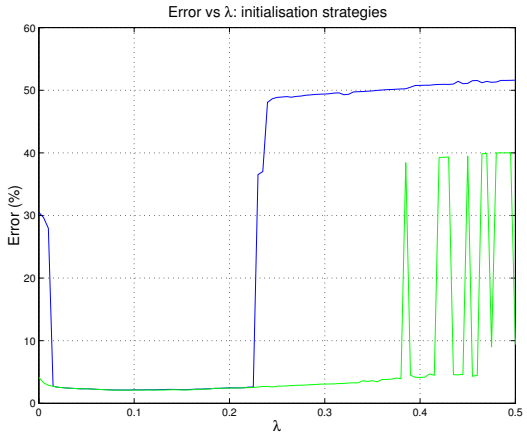
Table 5.15. Run times ('Time' columns) and number of iterations before convergence ('Itns' columns) of modified and conventional K-means. Data are collected from all the 5-mosaic segmentations. The run times are in seconds. The experiments are run on a 1.2GHz PC with 1GB RAM.

Transform type/depth	Conventional				Modified			
	Kaiser		Median		Kaiser		Median	
	Time	Itns	Time	Itns	Time	Itns	Time	Itns
DT-CxWT2	0.921	30.1	1.15	35.0	0.625	20.4	0.707	21.2
DT-CxWT3	1.72	31.2	1.74	40.0	1.15	20.7	1.02	22.0
DWT2	0.410	31.8	0.507	35.9	0.265	20.6	0.308	21.6
DWT3	0.625	34.6	0.558	36.8	0.379	21.1	0.344	22.8

Table 5.15 clearly shows that the modified K-means algorithm was faster than the conventional algorithm in the experiments. Although the modified K-means algorithm incurs an extra cost in computing the spatial distance metric, it requires approximately $\frac{1}{3}$ fewer iterations to converge to a solution. This vast difference between the number of iterations easily overcomes the per iteration penalty so the net result is a faster algorithm. In addition, the data for the modified K-means include run times with *all* values of the spatial proximity parameter, λ. Inappropriate, especially very small, values of λ would lead to unusually high number of iterations. This is because modified K-means approach conventional K-means as λ approaches zero. Thus, the modified K-means figures presented above are likely to be over-estimates of the true computational requirements. On the other hand, an important aspect of successfully using modified K-means is the estimation of an appropriate value for λ. This is likely to require significant computations, which would reduce the comparative advantage of modified K-means in terms of speed.

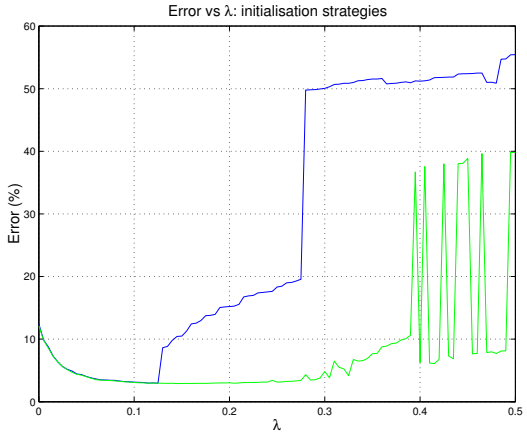
A key difficulty has been observed with the modified K-means when performing the experiments. It has been noticed that, in some instances, the segmentation performance varies greatly with small differences in λ when the cyclic initialisation scheme is used. The K-means algorithm is sensitive to initial conditions, and this property has been inherited by the modified version. This is not surprising, since the modified K-means algorithm is deterministic once the initial conditions are determined. The complication here is that the modified K-means algorithm is also sensitive to changes in λ. This means that, while

cyclic initialisation produces good segmentation results for a particular range of λ, it may be inappropriate for values of λ outside that range. Worse yet, it only takes a small change in λ for big performance differences to occur. The only real method to overcome the problem is to run the modified K-means algorithm multiple times with different, random initialisations. It will be possible to obtain a better estimate of the optimal clustering for *every* value of λ. This method adds to the computational burden tremendously, as the entire algorithm needs to be run multiple times. Figures 5.17 to 5.18 illustrate the problem with cyclic initialisation. For each example, modified K-means clustering is repeated 25 times with different random initialisations, and the best result (lowest error rate) is chosen to be the truly representative segmentation for each value of λ. It is plainly evident that, by repeating modified K-means with random initialisations, the error rate curve is greatly smoothed, at least before the breakdown point. It is interesting to see that, with random initialisation, it is possible to push the breakdown point further to the right. In other words, it is possible to obtain good segmentations even with high values of λ, provided the appropriate initialisation is used. However, the number of suitable initialisations decreases as λ is increased. With smaller values of λ, a large number of different initialisations converge to produce good segmentations. For large values of λ, however, only a small number of initialisations converge to a good clustering. In some cases, there may be none at all in the entire set of 25 different random initialisations. This explains the wild fluctuations observed for values of λ much higher than the optimum. If one can take the "envelope" of minima in these regions of the curves, one can see that a smooth degradation of segmentation accuracy as λ is increased. This points out that the breakdown phenomenon is more a manifestation of the initial state of the algorithm than an intrinsic behaviour of the modified K-means clustering. This claim has some direct support from the random initialisation error plots for the cases of "My 5b" and "Nat 5c", which lack obvious breakdown points (figures 5.18(b) and 5.17(b)). Rather, these error curves have a smooth, bowl-shape characteristic, instead of the familiar valley-between-ridge characteristics typical of cyclic initialisations. Overall, it is desirable to use different random initialisations in modified K-means clustering, for it will lower the sensitivity on the λ parameter. However, this benefit comes at a significant cost in computation.

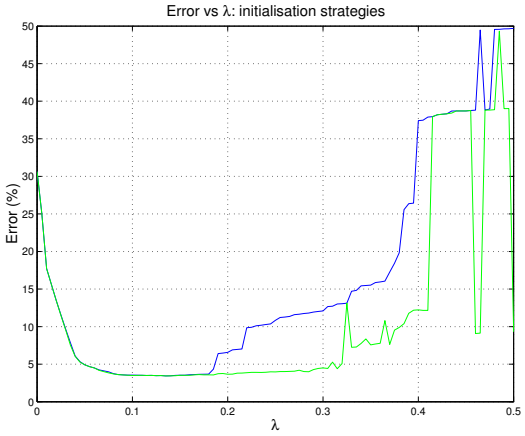


(a) Nat 5b

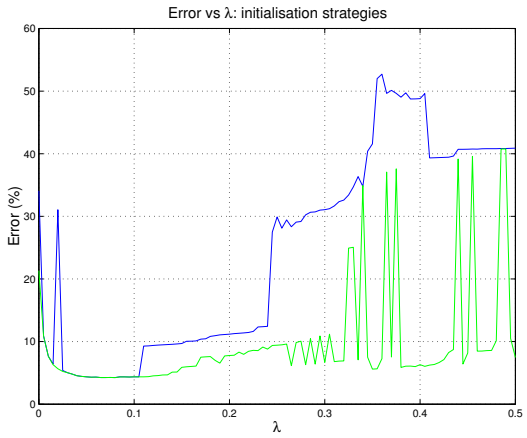
(b) Nat 5c



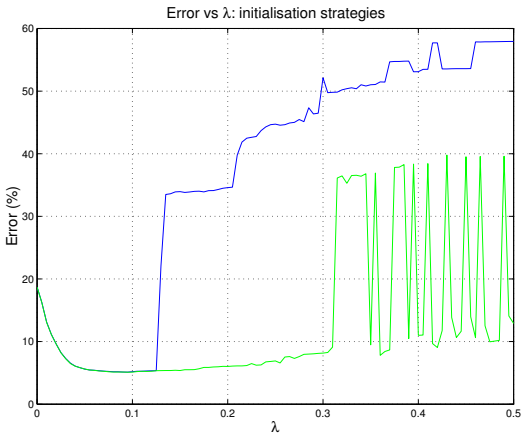
(c) Nat 5m



(d) Nat 5v

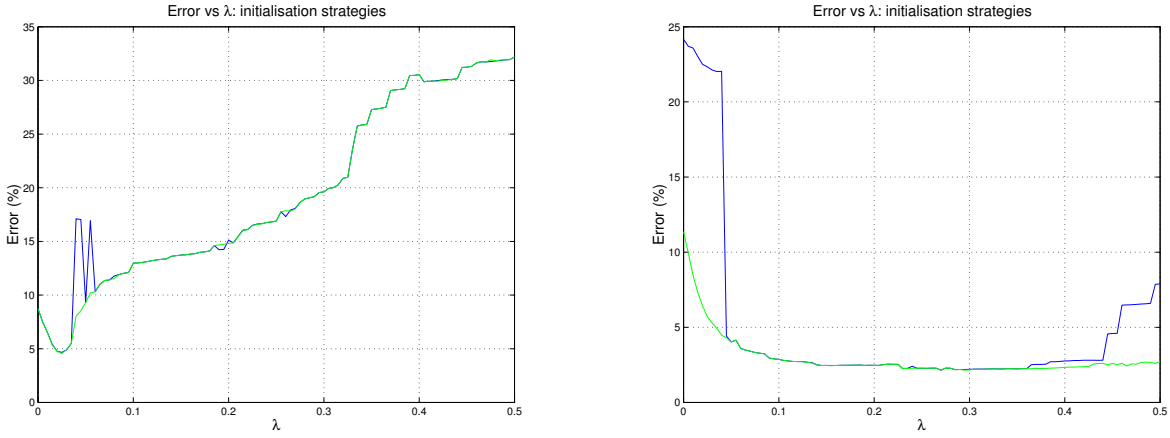


(e) Nat v2



(f) Nat v3

Figure 5.17. Cyclic and random initialisations in modified K-means. Average error rates from random initialisations are plotted in green lines, while the cyclic initialisation error rates are plotted in blue lines. The smoothing effect of random initialisation is plainly evident in this case. All the features are extracted from 3-level DT-CxWT with Kingsbury filters, and optimal Kaiser smoothing parameters are used for each image.



(a) My 5a (b) My 5b

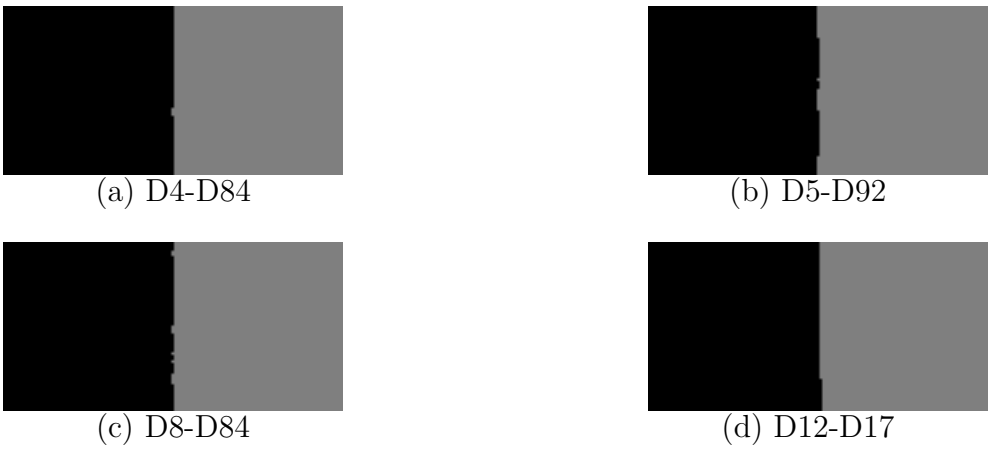
Figure 5.18. Cyclic and random initialisations in modified K-means. Average error rates from random initialisations are plotted in green lines, while the cyclic initialisation error rates are plotted in blue lines. All conditions are identical as in figure 5.17.

5.5.6 Segmented Images

This section shows segmented images from the modified K-means experiments. It is obviously infeasible to present the results from all the cases, so only selected samples of the results are shown. Appendix A has a full listing of all the input images used in the experiment, as well as the ground truths. The segmented images have individual regions shaded with different gray levels for simple visual interpretation. These gray levels are randomly assigned when the images are generated, and do not correspond to any textural properties in particular. All the segmented images are direct outputs of the modified K-means clustering algorithm, without any form of post-processing.

2-texture Segmentations

Figure 5.19 shows the segmentations for the 2-texture mosaics used in the experiments. The particularly simple nature of the separating boundaries make all these images simple for modified K-means. Nearly perfect segmentations are achieved in all these cases, with all errors restricted to be on or near the boundary.



(a) D4-D84 (b) D5-D92

(c) D8-D84 (d) D12-D17

Figure 5.19. 2-texture mosaic segmentation results. (a) D4-D84, (b) D5-D92, (c) D8-D84 and (d) D12-D17.

5-texture Segmentations

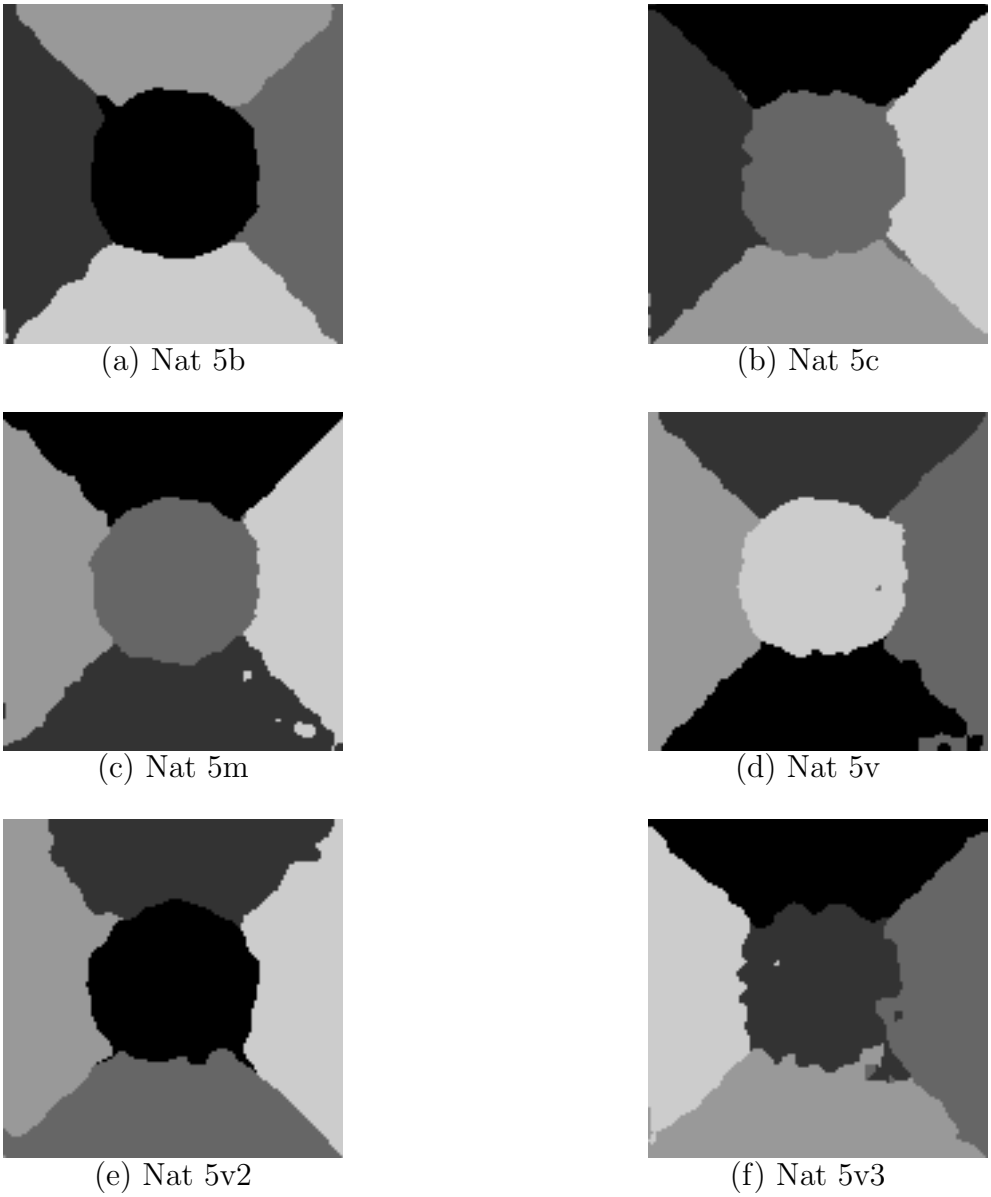
Figure 5.20 shows the segmentations for a selected subset of 5-texture mosaics used in the experiments. For the images "My 5a" and "My 5b", the segmentation results are vastly different, despite the same constituent textures in both. This is due to the more complicated boundaries present in the former. In particular, the clustering algorithm has the most problems dealing with the top-left boundary between two very similar textures. This has resulted in a significant number of misclassified pixels around this boundary. In contrast, "My 5b" has a very good segmentation, with clear boundaries separating the texture regions. It should be noted that there are small number of residual regions present in the segmentation of "My 5a". In practice, these segmentations can be further enhanced by post-processing operators, which can remove small residual regions, before the segmented information is used.

(a) My 5a (b) My 5b

Figure 5.20. 5-texture mosaic segmentation results. (a) My 5a and (b) My 5b.

Segmentations of the "Nat" series of 5-texture mosaics are presented in figure 5.21. The segmentation performance for "Nat 5b" and "Nat 5c" are particularly pleasing, with clean, accurate boundaries identified; all the segmentation errors are concentrated on or near the boundaries. There are slight problems with the edges of the actual images, such as in the top corners of the segmentation of "Nat 5b", where the smoothed features contain artificial data. Performance for "Nat 5v", "Nat 5v2" and "Nat 5v3" are expected to be the worst, since the constituent textures are very difficult to distinguish even with human interpretation. Due to the sheer volume of images in the "bonn" series of images, figure 5.22 can only showcase a small set of segmentation results. These 6 images are randomly chosen from the set of all segmentations, and are quite typical of the modified K-means segmentation performance on the "bonn" set. Apart from some boundary effects,

the segmentations are of very high quality, with accurate identification of the straight and curved boundaries.

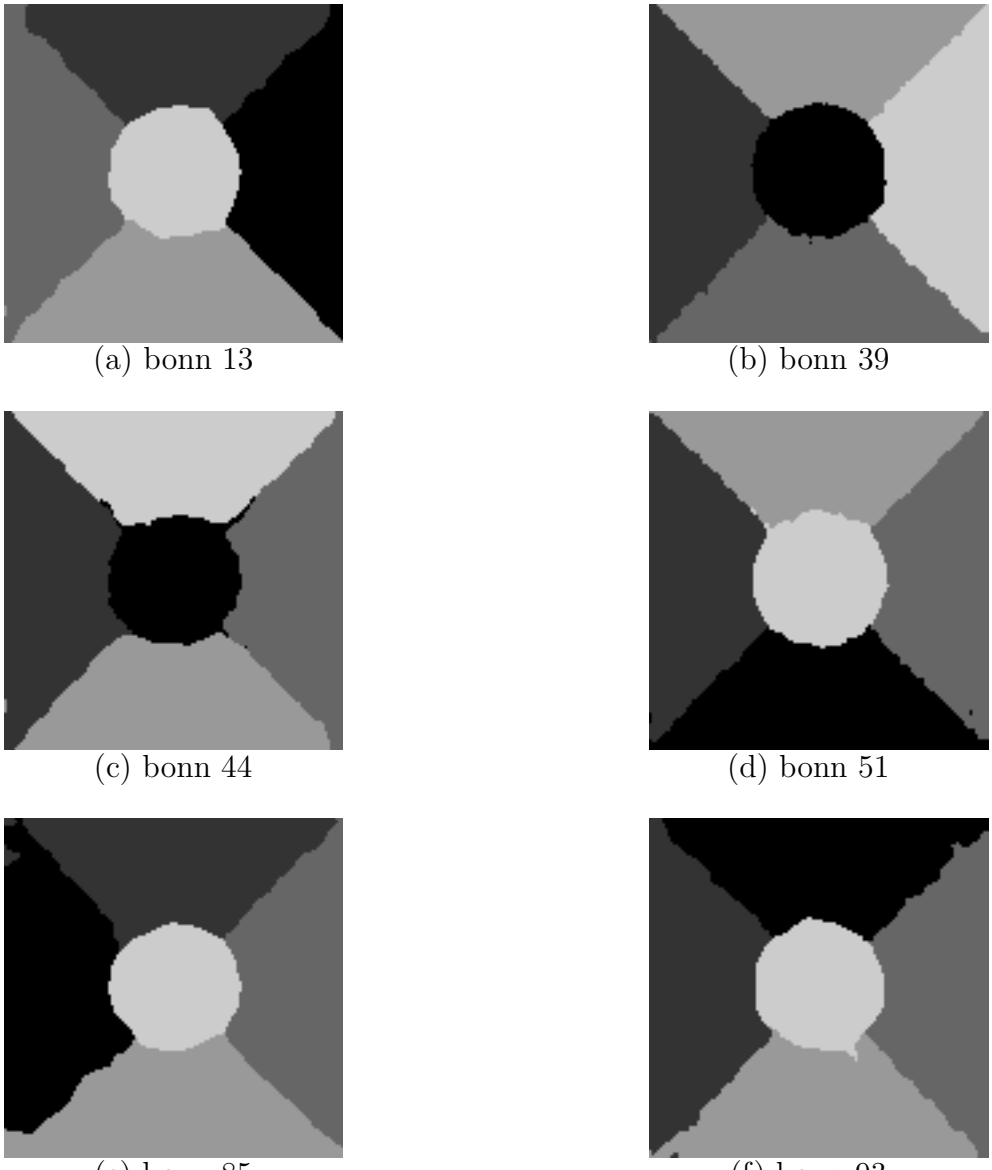


(a) Nat 5b (b) Nat 5c

(c) Nat 5m (d) Nat 5v

(e) Nat 5v2 (f) Nat 5v3

Figure 5.21. 5-texture mosaic segmentation results. (a) Nat 5b, (b) Nat 5c, (c) Nat 5m, (d) Nat 5v, (e) Nat 5v2 and (f) Nat 5v3.



(a) bonn 13 (b) bonn 39

(c) bonn 44 (d) bonn 51

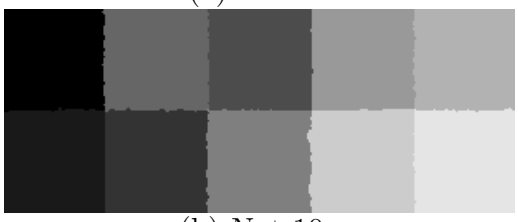
(e) bonn 85 (f) bonn 93

Figure 5.22. 5-texture mosaic segmentation results. (a) bonn 01, (b) bonn 44, (c) bonn 61, (d) bonn 82, (e) bonn 85 and (f) bonn 93. Due to space limitations, only a sample of the results for the Bonn series is shown here.

Other Segmentations

Figures 5.23 and 5.24 show the segmentation results from the three 10 and 16-texture mosaics in the input suite. The segmentation result for the 10-texture mosaics are excellent visually, with boundaries located accurately. For the 16-texture mosaic, the result is not as good; there are significant residual regions in several places, and the curved separating boundaries are inaccurate. However, this is already a vast improvement upon the conventional K-means segmentation of the same image.

(a) Nat 10



(b) Nat 10v

Figure 5.23. 10-texture mosaic segmentation results. (a) Nat 10, (b) Nat 10v.

Nat 16b

Figure 5.24. 16-texture mosaic segmentation result.

5.5.7 A Potential Indicator

In the experiments of texture mosaics, the modified K-means has been shown to be an effective clustering tool for segmentation. For this algorithm to be generally useful, it is clearly desirable to have a method to find the optimal values of λ. This is a difficult problem in general, but a method to find indicators of optimality is more achievable. By narrowing the search range for the optimal λ, it is possible to greatly speed up the search for the optimal segmentation. Knowledge of potential end markers for the search range would be essential to this search process. Section 5.5.4 has already discussed the characteristic behaviour of feature and geometric distortions when λ is varied. It is plausible that the characteristic traits of the distortions can be exploited to provide informative indicators for optimisation. For this purpose, the following functional is proposed:

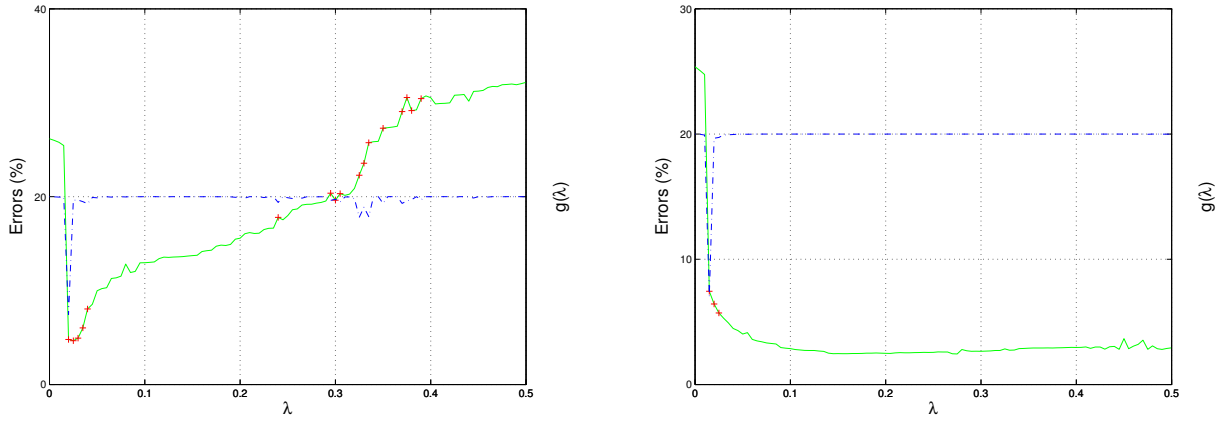
$$g(\lambda) = (\Delta d_{fts}) \times (\Delta d_{geo}) \tag{5.4}$$

where $\Delta d_{fts} = d_{fts}(\lambda_{k+1}) - d_{fts}(\lambda_k), k \in \mathbf{Z}$. d_{fts} is regarded as a function of λ, being the feature space distortion of the clustering produced from λ, and k is an index to the values of λ. Δd_{geo} is defined similarly. From its definition, this functional is expected to capture the upward and downward fluctuations of the feature and geometric space distortions. Since the feature and geometric space distortions are expected to experience large changes of opposite signs near breakdown, a large and negative value of g is likely to indicate a breakdown. While it will not directly provide an optimal value of λ, monitoring the value of g will at least give an indication of the upper bound of the useful range of λ values to use in modified K-means.

To illustrate the applicability of $g(\lambda)$ as an indicator, it is useful to plot both the error rates and $g(\lambda)$ as functions of λ on the same set of axes. These are shown in figures 5.25 and 5.26. In order to provide a true indication of the value of $g(\lambda)$, the plots are obtained from repeated runs with random initialisation, as described in section 5.5.5. The red crosses in the figures 5.25 and 5.26 highlight the values of λ where the magnitude of the function $g(\lambda)$ exceeds a given threshold. In these cases, the threshold is taken as the average magnitude of g over all values of λ. For the "Nat" series of input images, the indicator function performs well, with large negative spikes clearly showing the breakdown point(s) (see figure 5.26). In these plots, the red crosses clearly coincide with the rapid increases in the error rates. In particular, the function g accurately located the *first* breakdown points in almost all the cases. The "Nat" series of images do not seem to

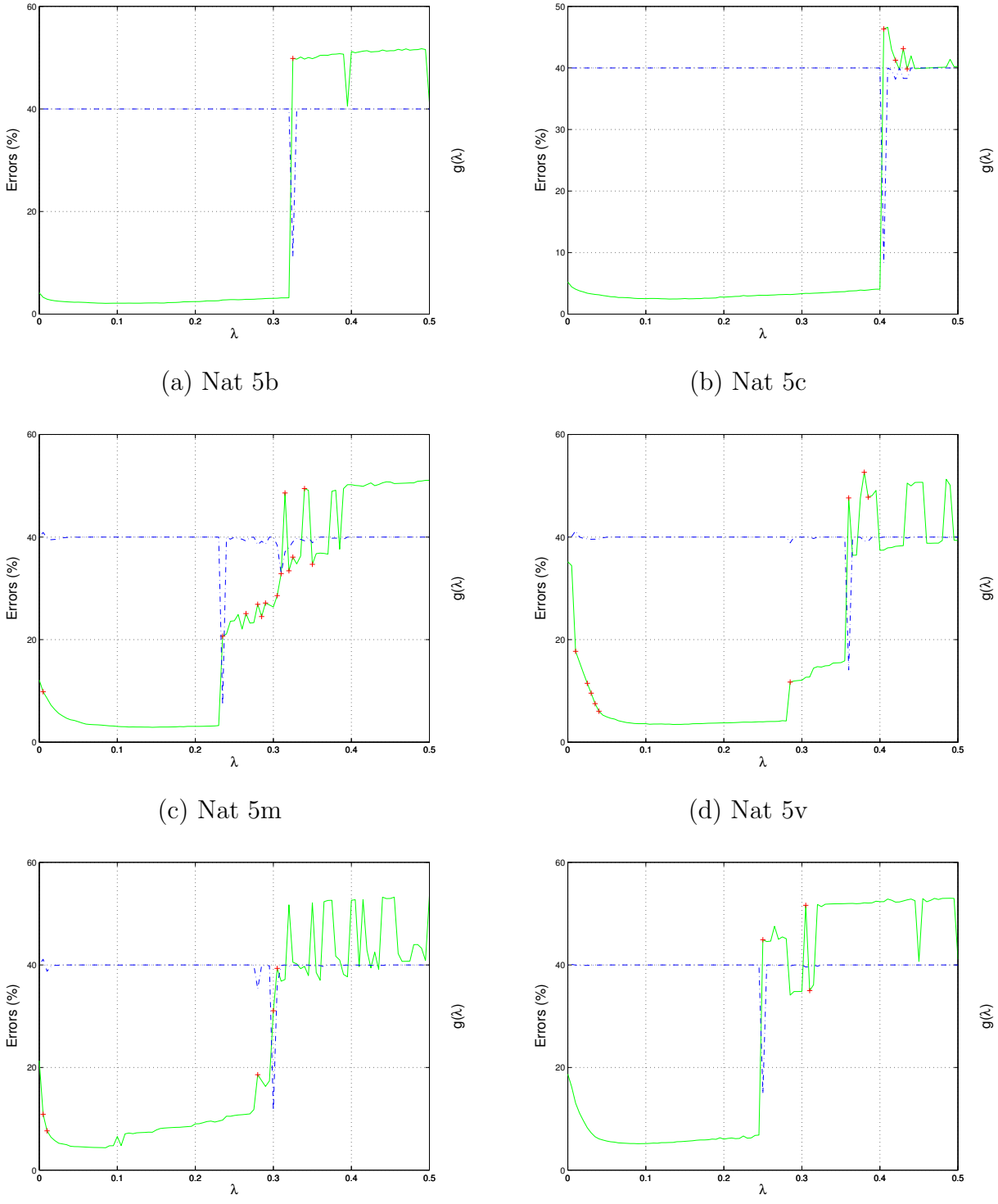
have clear lock-on points; rather, the error rates gradually improve as λ increases from zero. As a result, $g(\lambda)$ does not detect any lock-on points, as evidenced by a lack of red crosses for low λ values in the plots. However, it has been noticed for images "Nat 5m" and "Nat 5v", there are small fluctuations about zero for g at low values of λ, which leads to some erroneous detections of lock-on/breakdown points. For the images "My 5a" and "My 5b", $g(\lambda)$ completely fails to provide any meaningful indication for lock-on. In fact, the function indicates the presence of a breakdown (large negative spike) precisely where a lock-on occurs. The problems with the image "My 5b" is understandable, because the boundaries between the texture regions are very regular and convex, making it artificially suitable to large λ values. As a result, there are no breakdown points for this image, and the function g fails to act as an indicator.

Despite the difficulties with some cases, the function g provides useful indications on the location of breakdown points in general. For example, assume that the location of largest negative spike for g is used as a simple indicator for the breakdown point. This test successfully provides an upper bound on the useful range of λ in 75% of all the experiments with modified K-means. In these success cases, the indicated breakdown values are greater than the optimal value of λ. In other words, the behaviour of g provided a valid termination point for these segmentation experiments.



(a) My 5a (b) My 5b

Figure 5.25. Plot of $g(\lambda)$ and error rates for "My 5a" and "My 5b" images. g are plotted as blue dash-dot lines, while error rates are plotted as green solid line. Values of λ that gives rise to significant peaks in g are marked with a plus sign at the corresponding points on the error rate plots. Note that the scales for both error rates and $g(\lambda)$ are different between plots.



(a) Nat 5b (b) Nat 5c

(c) Nat 5m (d) Nat 5v

(e) Nat 5v2 (f) Nat 5v3

Figure 5.26. Plot of $g(\lambda)$ and error rates for the "Nat" series of 5-texture mosaics. The style of these plots are the same as in figure 5.25.

5.6 Experiments with Modified Fuzzy K-means

In this section, the results from experiments using modified fuzzy K-means clustering is presented. Table 5.16 shows the segmentation results obtained for the entire suite of 115 input images. As in section 5.4, only the Kaiser window smoothing method is considered in these experiments, as it has been shown to produce more effective texture features from wavelet transforms.

Table 5.16. Condensed results of Modified Fuzzy K-Means experiments. The best error rates (%) are shown for each input image. This condensed table of results only shows the best case; a full set of results for all values of β is found in Appendix B.

	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
	(%)	β	(%)	β	(%)	β	(%)	β
D4-D84	0.00	0	0.06	0	0.00	0	0.00	0
D5-D92	0.00	0	0.21	1	0.00	0	0.22	0
D8-D84	0.00	2	0.09	0	0.13	2	0.10	55
D12-D17	0.29	0	0.81	34	0.26	55	0.67	55
My 5a	13.34	5	9.72	1	11.71	1	12.41	21
My 5b	2.17	8	1.43	55	1.95	55	1.26	55
Nat 10	2.54	55	2.49	8	2.22	55	2.21	13
Nat 10v	1.36	34	1.19	34	1.49	55	1.39	5
Nat 16b	7.73	1	6.07	1	6.84	0	5.52	1
Nat 5b	3.72	3	2.84	1	4.10	3	2.36	5
Nat 5c	4.75	5	3.25	5	4.55	3	2.62	3
Nat 5m	4.33	2	3.43	3	4.56	2	3.11	0
Nat 5v2	10.35	8	6.45	3	8.26	0	5.92	0
Nat 5v3	5.62	1	5.85	0	7.05	0	6.13	0
Nat 5v	3.92	5	3.06	2	3.44	2	3.12	2
bonn 00	2.33	34	2.21	34	2.70	34	2.45	21
bonn 01	3.11	5	2.75	5	4.42	3	4.45	2
bonn 02	2.76	13	2.25	21	2.96	8	2.64	8
bonn 03	3.09	3	2.08	8	3.75	5	2.61	5
bonn 04	1.97	13	1.69	13	2.38	13	2.00	8
bonn 05	3.91	5	3.31	3	2.98	5	2.68	3
bonn 06	2.22	5	2.03	5	2.84	2	2.39	5
bonn 07	2.73	8	2.34	34	3.72	13	2.26	21
bonn 08	4.93	3	4.47	1	6.38	0	5.28	0
bonn 09	3.93	1	3.75	5	4.55	1	4.25	0
bonn 10	4.05	5	3.48	8	8.88	8	8.15	5
bonn 11	1.90	5	2.15	5	3.09	5	3.12	5
bonn 12	4.03	3	4.05	5	5.45	5	4.92	3
bonn 13	2.19	2	1.75	3	2.88	2	2.08	1
bonn 14	2.89	8	2.23	13	3.22	13	2.25	13
bonn 15	1.74	13	1.32	8	2.29	5	2.14	3
bonn 16	1.67	21	1.81	13	2.28	21	2.30	8
bonn 17	2.76	1	2.76	3	4.74	13	3.72	2
bonn 18	4.13	3	4.13	3	4.36	0	4.72	0
bonn 19	2.15	3	2.19	13	2.80	8	2.19	8
bonn 20	2.26	13	2.56	5	2.19	5	3.04	5
bonn 21	3.17	2	3.00	8	3.78	5	3.34	8
bonn 22	10.12	5	7.93	3	10.28	2	8.40	2
bonn 23	1.89	34	2.03	21	2.36	13	4.00	0
bonn 24	2.19	13	2.13	13	3.00	5	3.05	3
bonn 25	3.06	3	3.34	0	4.05	2	3.80	2
bonn 26	1.66	13	1.75	34	2.13	8	2.37	21
bonn 27	3.94	5	3.63	21	4.94	3	3.92	13
bonn 28	26.19	34	4.88	3	22.91	13	14.74	8

	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
	(%)	β	(%)	β	(%)	β	(%)	β
bonn 29	2.23	13	1.85	13	3.66	8	2.28	21
bonn 30	2.21	3	2.01	5	3.12	2	2.81	21
bonn 31	4.49	3	3.10	3	5.05	3	4.00	5
bonn 32	3.95	13	3.47	21	3.73	8	2.86	8
bonn 33	2.53	21	2.16	13	3.03	13	2.72	21
bonn 34	6.38	2	4.90	2	6.90	2	5.83	3
bonn 35	2.47	8	2.00	21	3.33	8	2.00	21
bonn 36	3.01	8	2.56	13	3.68	8	3.77	13
bonn 37	2.27	5	1.81	8	2.22	8	1.81	21
bonn 38	3.03	8	3.34	5	4.35	5	4.93	3
bonn 39	1.70	8	1.49	13	2.19	5	1.84	34
bonn 40	1.79	34	1.90	21	2.70	13	2.16	21
bonn 41	3.98	3	5.33	1	6.32	2	6.40	0
bonn 42	21.48	0	2.16	5	3.65	3	2.67	3
bonn 43	3.11	2	2.17	3	3.09	2	2.84	1
bonn 44	3.15	13	2.43	21	3.91	5	3.18	3
bonn 45	8.22	8	6.83	8	7.89	0	9.39	3
bonn 46	5.58	2	5.32	2	8.83	3	5.81	8
bonn 47	4.09	8	2.20	3	4.03	5	3.70	5
bonn 48	2.36	5	2.03	5	2.70	8	2.17	8
bonn 49	2.76	3	2.62	5	3.31	5	3.31	3
bonn 50	2.71	13	2.95	8	3.78	8	4.17	8
bonn 51	1.89	8	1.88	13	2.12	5	1.99	3
bonn 52	2.29	8	2.13	8	3.05	8	2.78	13
bonn 53	2.37	5	2.39	13	3.41	13	3.07	8
bonn 54	3.29	21	2.78	21	4.16	34	3.33	8
bonn 55	2.56	13	2.32	34	2.86	8	2.85	21
bonn 56	7.23	0	4.79	3	5.59	0	5.60	2
bonn 57	4.70	2	3.30	3	4.77	2	4.02	3
bonn 58	3.58	5	3.34	2	4.52	2	3.75	3
bonn 59	3.28	5	2.57	5	2.75	3	2.58	8
bonn 60	2.73	2	2.40	5	3.45	5	2.96	5
bonn 61	2.48	13	2.22	21	2.76	13	2.31	34
bonn 62	3.17	2	3.24	2	3.80	2	3.72	0
bonn 63	4.72	1	3.51	2	3.61	2	4.33	2
bonn 64	5.19	21	4.11	55	4.89	34	3.95	55
bonn 65	3.38	8	1.78	13	2.41	8	1.89	13
bonn 66	1.68	13	1.50	13	2.11	5	1.70	34
bonn 67	3.21	13	3.05	8	3.80	13	3.63	5
bonn 68	3.55	13	3.11	13	3.15	8	2.98	5
bonn 69	4.35	1	3.77	3	4.96	0	3.78	3
bonn 70	1.68	8	1.65	8	2.48	8	2.49	8
bonn 71	2.76	8	2.23	13	3.75	21	3.06	13
bonn 72	3.80	5	3.09	8	3.34	21	2.42	34
bonn 73	3.88	5	3.77	5	6.56	2	5.19	0
bonn 74	2.30	3	2.07	5	2.94	1	2.17	0
bonn 75	3.02	5	1.95	5	2.84	3	2.22	5
bonn 76	3.27	3	2.19	5	3.17	3	2.62	3
bonn 77	3.43	0	3.10	0	3.72	0	2.87	13
bonn 78	3.16	8	2.91	8	4.21	5	4.16	8
bonn 79	3.78	3	3.20	5	3.82	3	3.11	5
bonn 80	1.72	8	1.27	34	2.62	5	2.06	13
bonn 81	3.30	2	2.46	3	4.09	1	2.94	2
bonn 82	2.08	13	2.02	13	2.56	3	2.68	5
bonn 83	2.72	13	2.08	13	3.22	8	2.60	8
bonn 84	2.24	13	2.11	8	2.96	5	2.79	5
bonn 85	3.00	21	2.36	21	5.97	5	3.50	3
bonn 86	2.79	8	2.58	8	4.82	5	4.24	13
bonn 87	5.68	5	3.13	8	6.43	1	3.33	5
bonn 88	2.50	13	3.01	13	3.99	5	3.88	8
bonn 89	2.36	8	2.23	13	2.70	5	3.11	5
bonn 90	2.56	3	2.34	3	3.10	2	2.91	0

	DT-CxWT 2		DWT 2		DT-CxWT 3		DWT 3	
	(%)	β	(%)	β	(%)	β	(%)	β
bonn 91	2.94	3	2.87	8	3.15	3	3.39	13
bonn 92	2.40	3	2.00	13	2.93	8	2.26	8
bonn 93	4.06	5	3.27	34	3.81	2	3.06	8
bonn 94	2.75	3	2.59	5	4.50	8	3.79	5
bonn 95	6.21	1	6.03	1	10.60	2	8.28	5
bonn 96	3.04	3	2.74	3	3.25	2	2.77	8
bonn 97	1.61	13	1.41	13	2.29	8	1.85	8
bonn 98	2.29	8	2.31	5	2.92	2	2.81	21
bonn 99	2.37	8	1.84	34	2.66	13	2.36	13

The average error rates for the modified fuzzy K-means algorithm are slightly higher than those obtained from modified K-means. Across the set of input images, the average error rate is 2.84% with level 2 DWT features, which is close to the average error of 2.61% for modified K-means, obtained with level 2 DT-CxWT features. However, it is observed that the error rate fluctuates more than the modified K-means; for the same feature extraction methods, modified fuzzy K-means can have error rates ranging from under 2% to over 15%. For the corresponding set of features, modified K-means show a greater ability to consistently produce good segmentations. Notably, modified fuzzy K-means failed to produce good segmentations for some images in the test suite, while modified K-means did not particularly fail in any of the experiments.

Table 5.17. Modified fuzzy K-means results summary and comparison. The overall mean and median error rates (%) from the modified fuzzy K-means experiments are shown for different transform types and depths. The 95% confidence intervals for the mean error rats are also listed. Corresponding quantities for fuzzy K-means are repeated from table 5.13 to facilitate direct comparison.

Transform type/depth	Modified			Modified Fuzzy		
	Mean	Median	95% C.I.	Mean	Median	95% C.I.
DT-CxWT2	2.61	2.42	(2.40, 2.81)	2.84	2.46	(2.57, 3.11)
DT-CxWT3	2.63	2.37	(2.42, 2.84)	3.45	2.96	(3.07, 3.84)
DWT2	3.06	2.84	(2.82, 3.30)	3.67	3.00	(3.06, 4.28)
DWT3	2.98	2.81	(2.75, 3.21)	4.03	3.44	(3.54, 4.52)

Table 5.18 illustrates the variation of the optimal Kaiser smoothing parameter value for the modified fuzzy K-means experiments. There is a particular preference for moderate values of β. This is in accordance with the trend observed from modified K-means; once again, the effect of introducing the spatial proximity term allows less severe smoothing of the features to effect a good segmentation. Overall, the modified fuzzy K-means

experiment results agree with the results from modified K-means, giving further support for the usefulness of the modified technique.

Table 5.18. Modified fuzzy K-means results summary - part 2. The optimal Kaiser window β parameter values for all images, transform types and depths are tallied from the modified fuzzy K-means experiments to produce the distribution of optimal β.

Transform type/depth	β									
	0	1	2	3	5	8	13	21	34	55
DT-CxWT 2	6	7	11	19	20	21	20	5	5	1
DT-CxWT 3	5	7	6	17	21	17	21	10	9	2
DWT 2	11	5	21	13	23	20	12	3	3	4
DWT 3	14	3	8	17	20	21	12	12	4	4

5.7 Experiment Summary

Following the discussion of wavelet-based texture feature extraction and clustering algorithms in chapters 3 and 4, this chapter presents the experimental details and results. The properties of different extracted texture feature sets are examined. In particular, the separability, believed to be crucial in this work, of the feature sets are measured using several techniques. In the main experiments, it has been shown that wavelet-based texture features and modified K-means form an effective combination for texture segmentation. The algorithm yielded good results over many experiments with artificial mosaics. It is interesting to compare the results of this chapter with similar experiments on artificial mosaics using different texture features.

Randen and Husøy performed an extensive series of texture segmentation experiments using a variety of different wavelet features and learning vector quantisation (LVQ) classification [62, 60, 61, 63, 64]. In this chapter, the modified K-means algorithm managed to segment the "Nat" series of images very effectively. The same set of images have been used in other texture segmentation experiments in the literature. The most directly comparable results are those shown in table 9 on page 308 in [64], which are obtained using different wavelet transforms. The first five columns in that table correspond to the images: Nat 5c, Nat 5v, Nat 5v2, Nat 5v3 and Nat 5m. The eighth and ninth columns are for the images Nat 10 and Nat 10v, respectively. Unfortunately, the sixteen-texture mosaics found in columns 6 and 7 are not the same as Nat 16b. However, the boundaries are exactly the same, and the constituent textures are very similar to the image 11(f)

(sixth column) in [64], and so it is reasonable to assume similar results if it is used in the current experiments. It is immediately obvious that the error rates reported in this chapter are far superior to those found in [64], even the cases where full-rate decompositions are used. While the modified K-means algorithm can routinely produce segmentation error rates of lower than 5%, the algorithms presented in [64] produced a best average of 20%. Even the conventional K-means results in section 5.3 are better; this suggests that the improvements are not only due to the modified K-means algorithm. The introduction of a variable smoothing window shape (or effective size) contributes to the improved segmentation performance. In related work by Randen and Husøy [61, 63] reported segmentation results for the images Nat 5b and Nat 16b using similar methods. These error rates are much closer to the performance of the modified K-means method described in this chapter. Their algorithm managed to segment Nat 5b and Nat 16b with 97.5% and 93.4% accuracy, respectively. In comparison, the modified K-means technique produced best segmentation accuracy of 97.97% and 94.89% for the corresponding images. Various two-texture mosaic segmentation results are reported in [62, 60, 61, 63, 64]. A different technique is used to segment the two-texture mosaics; it involves optimising the filter coefficients used in the image decomposition, with respect to a given objective function. This has the advantage of being adaptive to each input image. The results produced are good, with the reported error rates as low as 0.6%. However, the segmentation performance for the D4-D84 and D5-D92 mosaics are poor, with respect to the excellent results obtained from D12-D17 (table 9 in [64]). It appears that the optimisation technique has difficulties dealing with textures with similar coarseness.

Apart from the "Nat" series of mosaics, the other set of input images is the "bonn" series obtained from the University of Bonn (see Appendix A). In [57, 28], Hofmann, Puzicha *et al* published details of their texture segmentation algorithm. For artificial mosaic experiments, the Bonn suite of images were created. In [57], results of segmentation experiments performed on this suite is presented. Median error rates of approximately 5% to 7% are reported, obtained using a range of methods described in [57]. In comparison, the modified K-means method with depth 3 DT-CxWT wavelet features produced a median error rate of 2.37% over the 100 input images. This is further evidence that the results in this thesis compare very favourably with those obtained from other approaches.

This page is blank

Chapter 6

Application Examples

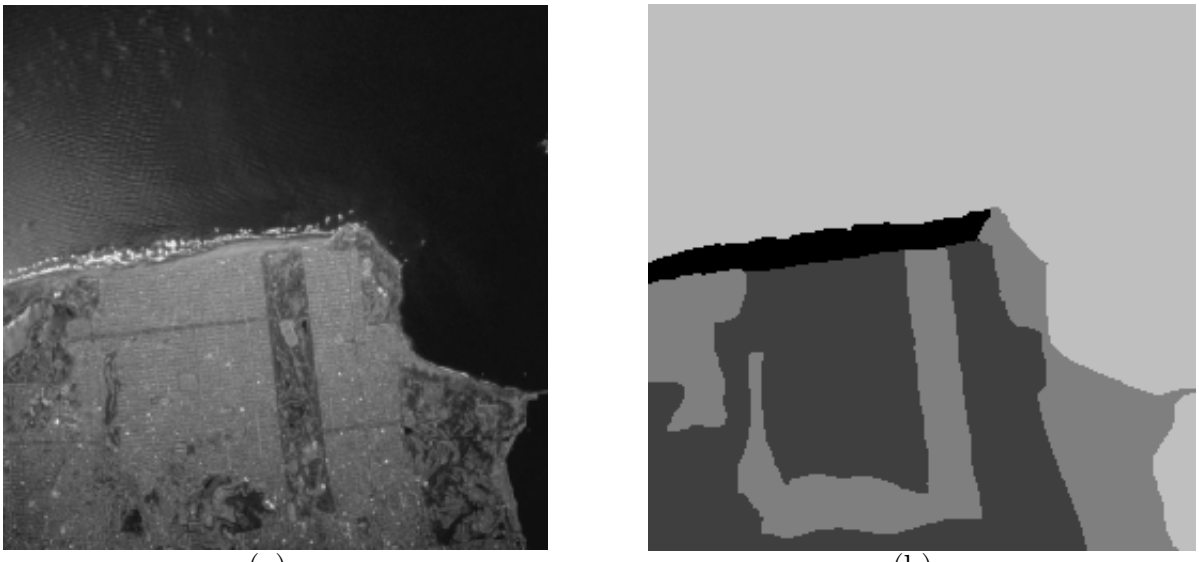
The problem of texture segmentation has numerous applications. Texture is a fundamental cue in human visual information processing. It can be used in a variety of problems, especially in images where texture contains a significant amount of the visual information. In chapter 5, wavelet-based texture features are used to segment artificial mosaics composed from natural textures arranged in regular geometric shapes. In real problems, texture is one visual cue used to understand the composition of images. In this chapter, several examples of real-world applications of texture segmentation are examined. The same wavelet-based texture features are used in these segmentation problems. More precisely, discrete and complex wavelet features are extracted, smoothed using Kaiser windows and clustered with the modified K-means algorithm. This chapter is organised as follows. Section 6.1 presents some applications in surveillance. Section 6.2 tackles the problem of segmentation of objects in a scene using texture information. Section 6.3 shows how texture segmentation can be used for document processing.

6.1 Surveillance

The data used in this example is an aerial photograph of an area of San Francisco, shown in figure 6.1(a). This photograph can be found from the University of Bonn's Computer Vision Group website [14]. This is a commonly used example to test the performance of various segmentation algorithms. The photograph covers the coastal stretch of San Francisco Bay. Four different textures are identified by the author in this image: the sea, beach, residential and parklands. A hand segmentation is shown in figure 6.1(b). It

should be noted that the author is not a trained terrain expert; the segmentation is done purely with common human perception. When performing the hand segmentation, the boundaries separating the regions are chosen to be as smooth as possible, whenever the true boundary cannot be easily identified.

In complete contrast with the image mosaics used in chapter 5, the segmentation has some disconnected clusters which belong to the same class. For example, the parklands region is actually separated into three different spatial clusters (see figure 6.1(b)). Therefore, the interchangeability of the terms class and cluster in the previous chapters no longer holds true. In this case, the term cluster is used to describe a connected set in the image, while class refers to all clusters which contain the same texture. Since the modified K-means algorithm encourages the formation of spatially compact clusters, this aerial photograph should provide a tough challenge for the algorithm. In practice, this means small values of the locality factor λ should be used in order to maintain spatially disconnected clusters. With a sufficiently large value of λ, the separate clusters belonging to the same class are forced to combine together, causing many classification errors. As a consequence, a reduced range of λ is used for the modified K-means experiments, ranging from 0 to 0.2, with steps of 0.002.



(a) (b)

Figure 6.1. Aerial photograph of San Francisco. (a) the original image and (b) a hand segmentation of aerial photograph of San Francisco.

Table 6.1 lists the segmentation results for different transform types and depths, as well as Kaiser window β parameter values. It is evident that the different combinations of feature parameters do not have a great impact on the segmentation performance. The error rates lie between 10.8% and 16.6% among all combinations. This is in complete contrast to the artificial mosaics in chapter 5. It is believed that the difficult nature of the example led to the relative insensitivity to the detailed feature extraction parameters. In comparison, the segmentation of the same image in [14] has an error rate of 27.4%, when compared against the ground truth in figure 6.1(b). In this segmentation, the brighter stretch of water (top left quadrant of the image) is classified as being different from the remainder of the sea, and the beach is assigned to the same class as the parklands. The images are available online at [14].

Table 6.1. Aerial surveillance segmentation results. The error rates (%) are listed for different wavelet transforms and smoothing parameters. Also listed are the 95% confidence intervals for the overall mean error rate.

Transform	0	1	2	3	5	8	13	21	34	55	Mean	95 % C.I.
DT-CxWT2	15.07	14.52	13.29	12.26	**11.57**	11.90	12.21	13.04	14.00	14.97	13.27	(12.39 14.15)
DT-CxWT3	14.07	13.70	13.05	12.15	11.35	**10.81**	11.11	11.66	12.40	13.90	12.41	(11.59 13.22)
DWT2	16.53	15.70	14.62	**13.84**	13.99	14.47	15.30	15.62	16.62	17.17	15.39	(14.61 16.17)
DWT3	15.75	15.20	13.85	12.67	**11.68**	11.99	12.32	13.06	13.79	15.19	13.54	(12.56 14.52)

Figure 6.2 shows the segmented image, and each region from the original image. It is clear, from these images, that the algorithm successfully segmented the beach (figure 6.2(b)) and the sea (figure 6.2(e)). The parklands and residential area (figures 6.2(c)-(d)) are less clearly separated, but significant portions are segmented out from the original. It is noticed that the boundaries between some regions have been erroneously included in some regions; this effect is most notable for the parkland and residential areas. This is mostly attributable to the relatively small spatial parameter values. At the boundaries between textures, there are significant high frequency energies, and these often result in large coefficients in the wavelet transforms. Finer textures with irregular structures are also characterised by high energies in many high frequency subbands, and it is likely that some texture boundaries are mis-classified as these finer textures. For the artificial mosaics in chapter 5, larger values of the spatial locality factor in the modified K-means algorithm discouraged the formation of long, thin regions, thereby eliminating the problems with boundaries. However, as explained earlier, large λ values are not useful in this particular case. In addition to boundary problems, there is a small handful of residual

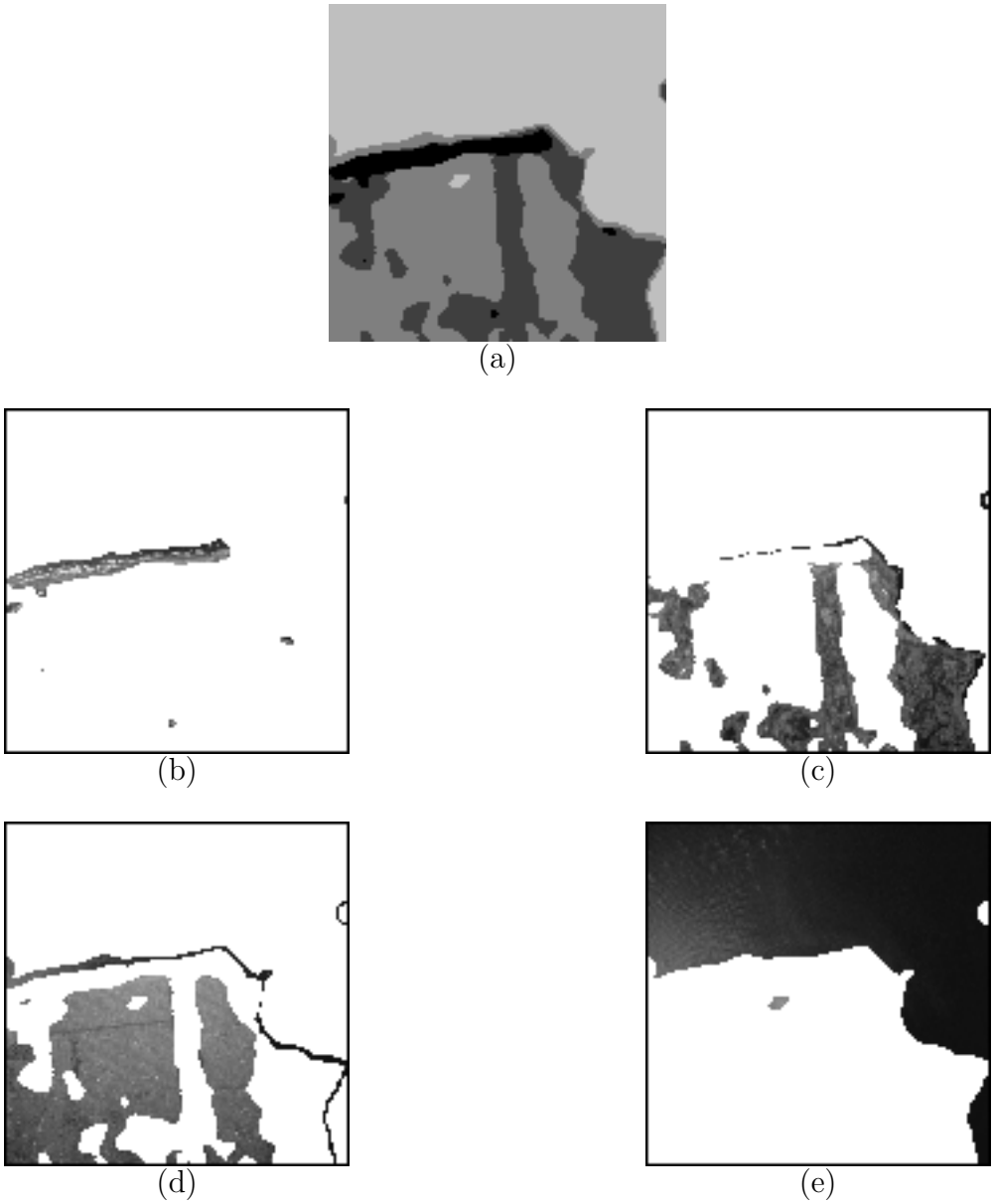
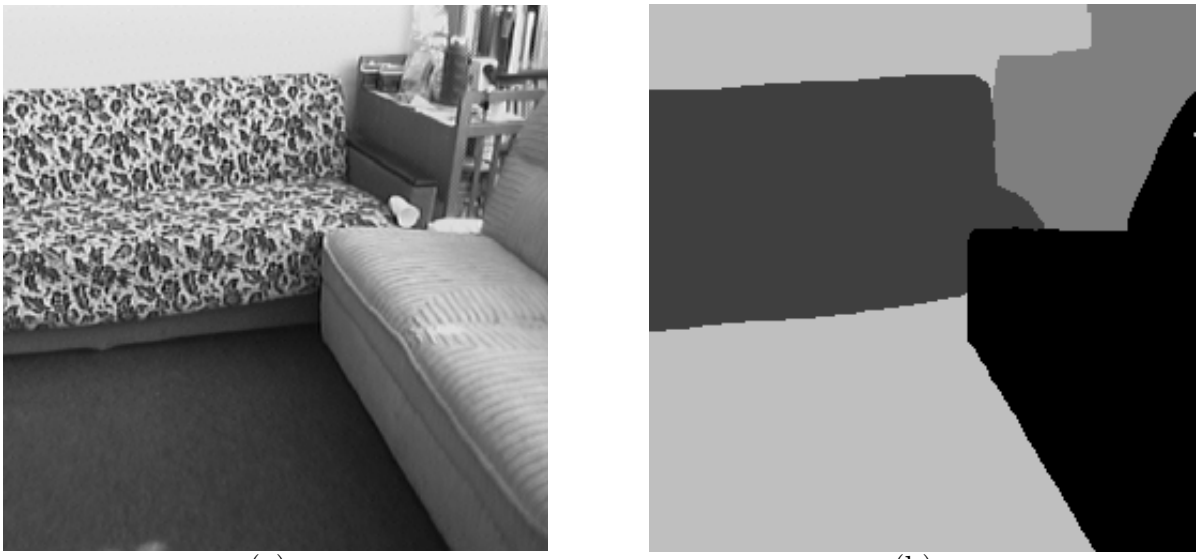


Figure 6.2. Segmentation of aerial photograph. (a) all segmented regions, (b) segmented beach terrain in original image, (c) parklands, (d) residential area and (e) the sea.

regions within some segmented regions, which degrades the accuracy of the segmentation. Fortunately, it is possible to apply simple median filtering in the post-processing to remove residual regions. For the texture boundaries, there are common techniques involving erosion filters to reduce the misclassification rate. However, the purpose of this section is to provide an illustration on the results obtainable from wavelet features and modified K-means clustering, and hence post-processing is not attempted in this thesis.

6.2 Object Segmentation

Figures 6.3(a) and 6.4(a) show the input images used in this section. Both input images consist of common objects in a real world scene; the first image is a picture of a sitting room with two sofas with distinctly different patterns for their covers, while the second image shows a wall with a radiator. The objects in these two scenes have rich textures to distinguish them from the rest of the image, and are therefore interesting test cases for the performance of the segmentation algorithm. Figures 6.3(b) and 6.4(b) show the hand segmentation of the sofa and radiator images, respectively. These segmentations are performed by the author. The sofa image is separated into four distinct regions, corresponding to: sofa 1 (with flowers pattern), sofa 2 (lines pattern), bookcase and floor & wall. The floor and wall region is considered as one because they both have very smooth textures, and thus should be clustered together by a texture segmentation algorithm.

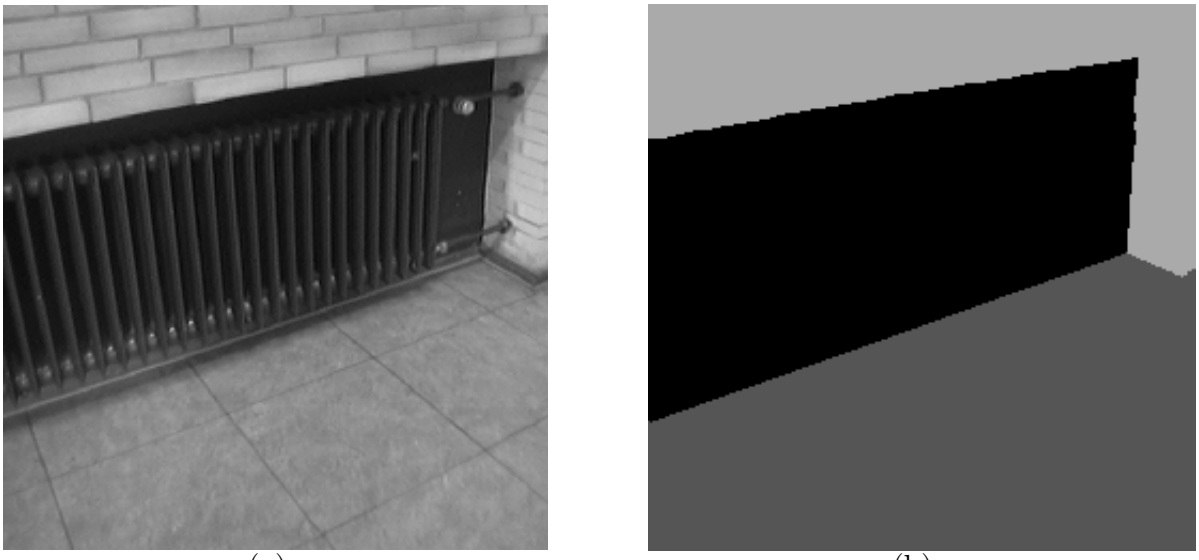


(a) (b)

Figure 6.3. Indoor scenery 1 - two sofas in a lounge room. (a) original image and (b) a hand segmentation into 4 separate regions. The different regions are: sofa 1 (flower), sofa 2 (lines), bookcase and wall & floor.

For the radiator image, three separate textures are identified: brick wall, radiator and tiled floor. In both images, the boundaries between the regions are chosen to properly segment objects in the image based on textures, as interpreted by a human observer. It is noted that the sofa image has a texture class separated into two spatially unconnected clusters, much like the San Francisco image in the previous section, while the radiator

image consists of three single-cluster textures. Therefore, it is expected that segmentation accuracy will be better for the radiator image.



(a) (b)

Figure 6.4. Indoor scenery 2 - a radiator in a lounge room. (a) original image and (b) a hand segmentation into 3 separate regions. The different regions are: brick wall, radiator and tiled floor.

Table 6.2 lists the segmentation results for the two scenes, under a range of feature extraction parameters. As with the case for the aerial photograph, the segmentation performance do not vary greatly among the different feature parameters for the radiator. The error rates lie between 9.66.% and 13.57%, which are slightly lower than for the aerial photograph. However, the error rate varies much greater for the sofa image. In particular, the segmentation performance varies greatly between different Kaiser window parameters; the range of error rates is between 9.88% and 27.22%. For some larger β values, the algorithm performs significantly worse than for lower values; this is indicative of the fact that the image does require adequate smoothing. It is believed that the flower patterns on the left-hand side sofa are the main reasons for needing heavy smoothing. For comparison, the segmentation algorithm of [14] has produced an error rate of 13.02% for the same sofa image. This is slightly higher than the best case performance of 9.88% obtained with our algorithm.

Figure 6.5 shows the segmented sofa image, and each of the separated regions. The result clearly bears out the object shape of the two sofas. However, the algorithm suffers from the same boundary misclassification problem as for the San Francisco Bay image

Table 6.2. Sofa and radiator scene segmentation results. The error rates (%) are listed for different wavelet transforms and smoothing parameters. Also listed are the 95% confidence intervals for the overall mean error rate.

Sofa	β											
Transform	0	1	2	3	5	8	13	21	34	55	Mean	95% C.I.
DT-CxWT2	16.47	15.48	15.20	14.90	14.49	14.35	**14.22**	26.03	14.25	27.22	17.06	(13.64 20.48)
DT-CxWT3	13.19	13.12	12.88	12.61	13.65	13.03	12.16	**10.21**	12.04	12.25	12.54	(11.88 13.19)
DWT2	10.94	10.88	10.52	10.24	**9.88**	13.44	11.77	12.22	26.53	26.67	14.04	(9.55 18.53)
DWT3	11.93	11.68	11.41	11.29	10.91	10.71	10.59	**10.43**	10.75	12.01	11.16	(10.78 11.55)
Radiator	β											
DT-CxWT2	**9.66**	9.82	10.08	11.04	11.23	10.88	11.22	12.85	11.78	11.96	11.06	(10.38 11.74)
DT-CxWT3	**10.52**	10.54	10.57	10.67	10.99	11.18	11.32	11.40	11.57	11.66	11.05	(10.75 11.35)
DWT2	**10.05**	10.14	10.34	10.63	11.74	11.84	11.35	11.77	12.10	13.57	11.37	(10.63 12.11)
DWT3	**10.27**	10.31	10.34	10.67	10.91	11.24	11.68	11.94	12.87	13.25	11.32	(10.59 12.05)

described in section 6.1. The bookshelf region suffers the most in particular; the books on the shelf contains a lot of high-frequency information, which causes this region to be aliased to the boundaries of sofa 1 (flower patterns) and its surrounding regions. This results in a clearly visible outline of this sofa (the flower-patterned portion, which excludes the sofa's base) in the books cluster in figure 6.5(c). The same problem also exists for the sofa 2 (lines) cluster, to a lesser extent. In addition to the boundary misclassification problem, the side of the bookshelf and the handle of sofa 1 is clustered with sofa 2. Both of these objects have smooth textures. Among the immediately neighbouring regions - sofa 1, sofa 2 and bookcase - sofa 2 has the closest matching texture. It should also be noted that the base of sofa 1, having a smooth texture, is clustered with the wall & floor. While this is unsatisfactory for an object recognition system, it is indeed a correct result from a texture segmentation perspective. Indeed, it is interesting to note that, if the relatively smooth texture of sofa 2 is also classified as belonging to the wall and floor texture, then the K-means algorithm's performance increases significantly. That is, the image is classified into 3, instead of 4 regions: smooth, sofa with flower pattern, and bookshelf. From a purely texture segmentation perspective, it is valid to reconsider the ground truth in this way. The texture of sofa 2 is quite similar to the smooth wall and floor, especially when compared to the bookshelf and sofa 1 textures. From an object segmentation point of view, the change in class assignment will produce an erroneous output. However, it is impractical and naive to rely solely on texture information to effect object segmentation. There are many other rich visual cues in a full computer vision system. The revised hand segmentation with three classes is illustrated in figure 6.6(a). The results of these experiments are shown in table 6.3, and the segmented image is shown

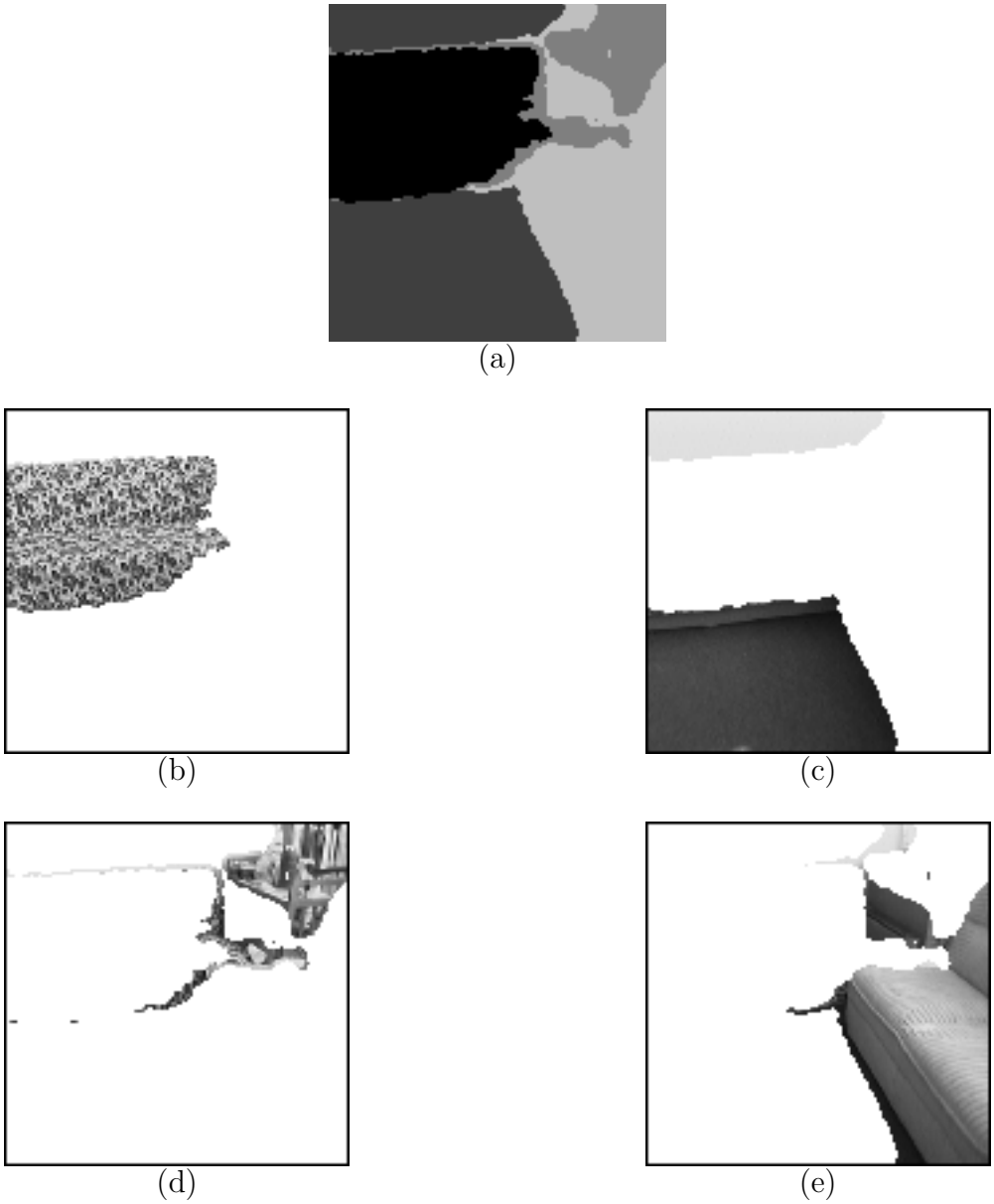


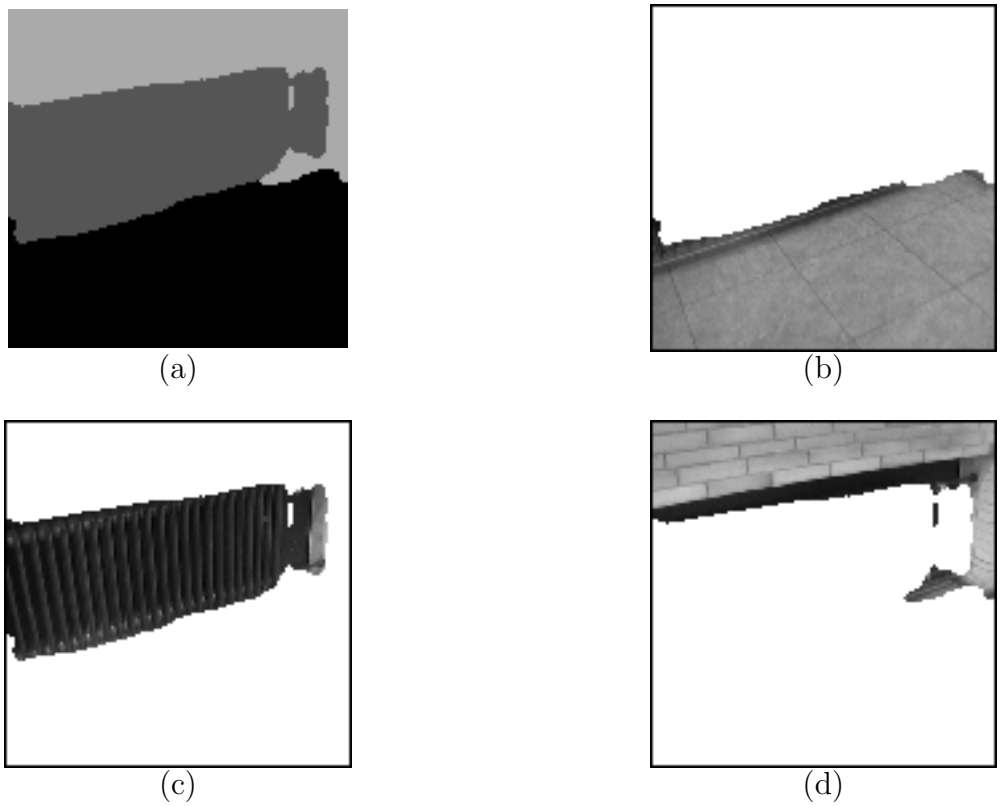
Figure 6.5. Segmentation of sofa scene. (a) segmentation into separate regions, (b) isolated region 1 of original image, (c) region 2, (d) region 3 and (e) region 4.

in figure 6.6(b). It is obvious that the error rates are significantly improved; the upper limit of the 95% confidence intervals are less than the mean error rates obtainable for the four-class segmentation in table 6.2.

Figure 6.7 illustrates the segmentated image of the radiator scene, and each individually separated region. The visual results are much better than what the error rates suggest. The three different textured areas - brick wall, tiled floor and radiator - are

(a) (b)

Figure 6.6. Alternative segmentation of indoor scenery 1. Three classes are used for this segmentation, instead of four. (a) the new ground truth (hand segmentation) and (b) segmentation result.



(a) (b)

(c) (d)

Figure 6.7. Segmentation of radiator scene. (a) segmentation into separate regions, (b) isolated region 1 of original image, (c) region 2 and (d) region 3.

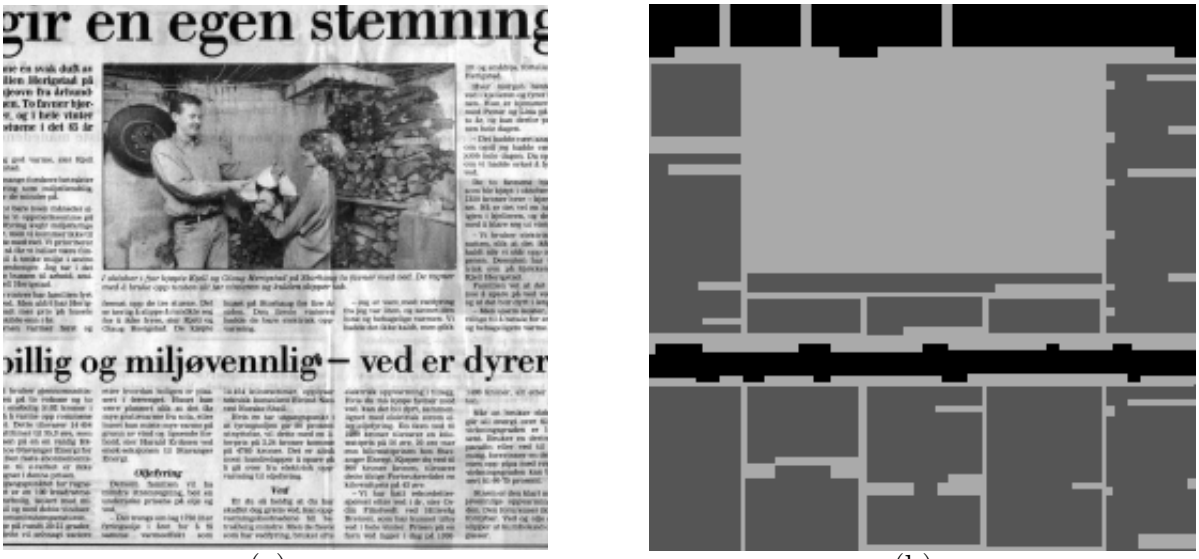
Table 6.3. Alternative sofa segmentation results. The error rates (%) are listed for different wavelet transforms and smoothing parameters. Also listed are the 95% confidence intervals for the overall mean error rate.

Sofa	β											
Transform	0	1	2	3	5	8	13	21	34	55	Mean	95% C.I.
DT-CxWT2	9.03	8.76	8.14	7.42	6.57	6.14	5.73	5.59	5.47	**5.45**	6.83	(5.83 7.83)
DT-CxWT3	9.39	9.03	8.23	7.48	6.65	6.23	5.77	5.49	5.43	**5.38**	6.91	(5.82 8.00)
DWT2	9.45	9.12	8.42	7.62	6.83	6.36	**6.12**	6.12	6.30	6.75	7.31	(6.40 8.22)
DWT3	8.97	8.67	8.09	7.46	6.63	6.28	5.85	5.60	**5.52**	5.58	6.87	(5.91 7.82)

clearly separated. Upon closer examination, the errors are concentrated in two places - above and below the radiator cluster. The pixels just below the brick wall are classified as belonging to the wall region, whereas they are assumed to be part of the radiator cluster when the hand segmentation was performed. However, based on texture information alone, it can be argued that the metal plate edges of the radiator, which are smooth in texture, do not belong to the central part of the radiator, which consists of a set of grills. This merely illustrates the insufficiency of texture as the only visual discrimination cue. Human intuition recognises the grilled part and the metal edges belong to the same object, despite their vastly different textures. Without the luxury of interpretation, a texture segmentation algorithm cannot possibly make the same decision. In this instance, a gray-level thresholding technique would likely assign the metal edges to the radiator cluster, simply due to the dark colour of the metal. Interestingly, the algorithm classified the pixels in the right-side of the radiator correctly, despite that region having the same metal texture. This can be explained by the fact that this region is narrow, and these vertical edges resemble the vertical grills on the radiator. Overall, the segmentation algorithm successfully segmented the main constituent regions in the image.

6.3 Document Processing

The final example in this chapter is the segmentation of a sheet of newspaper. Figure 6.8(a) shows a typical example of a piece of newspaper, containing text, pictures and headlines. In this instance, a segmentation into these three segments is desired, as shown in the hand segmentation in figure 6.8(b). While a specialist application will make use of additional properties of text and pictures (e.g. average brightness, existence of only horizontal and vertical boundaries) to effectively segment the image, a pure texture-based algorithm is likely to perform quite well. As in the case of San Francisco Bay, the hand segmentation shows many disconnected clusters belonging to the same class for this image. Thus, it is expected that small values of λ are likely to produce better segmentations.



(a) (b)

Figure 6.8. Newspaper article. (a) the original image, and (b) a hand segmentation of the article into three different textures.

Table 6.4. Newspaper article segmentation results. The error rates (%) are listed for different wavelet transforms and smoothing parameters. Also listed are the 95% confidence intervals for the overall mean error rate.

Transform	0	1	2	3	5	8	13	21	34	55	Mean	95% C.I.
DT-CxWT2	16.86	16.03	16.25	13.25	11.94	11.46	**10.99**	11.20	11.80	12.89	13.27	(11.65 14.89)
DT-CxWT3	16.14	15.44	13.62	12.55	11.63	16.26	19.96	**10.09**	10.38	11.12	13.72	(11.45 15.99)
DWT2	17.99	17.00	15.33	14.00	12.87	**12.27**	12.62	13.40	14.32	15.74	14.55	(13.18 15.93)
DWT3	18.10	17.26	15.80	14.53	13.46	12.70	**12.67**	13.06	13.75	15.06	14.64	(13.28 16.00)

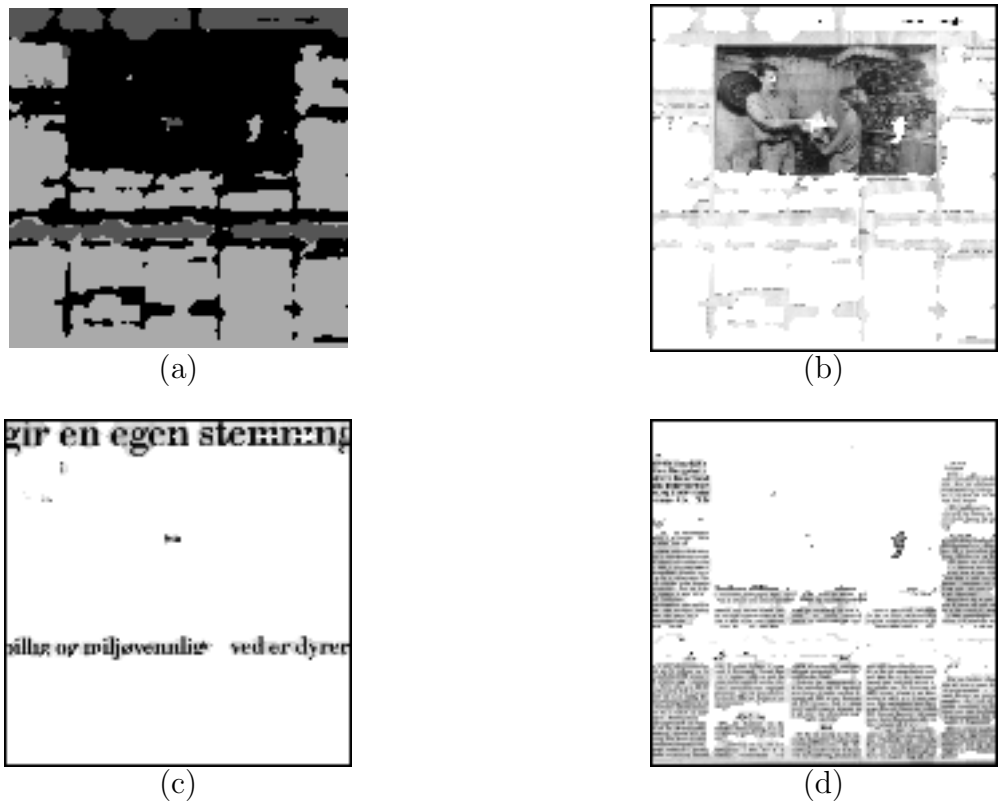


Figure 6.9. Segmentation of newspaper article. (a) segmentation into separate regions, (b) region 1 of original image, (c) region 2 and (d) region 3.

The error rates from the various cases are listed in table 6.4. With the different textures residing in so many disconnected clusters, it is not surprising that the error rates are not very low. Also, the clusters are mostly not convex, which can cause difficulties for the modified K-means algorithm. In this example, there is a greater amount of variation between the different transform and smoothing parameters, with the error rate varying from 10.1% to 19.96%. The DT-CxWT features performed better than the the DWT features, but the difference is only within 2%. In fact, all the mean error rates of the different transform type and depth combinations lie within the 95% confidence interval of each other. Thus, it can be concluded that no particular type or depth of transform is superior to each other. In figure 6.9, the best segmented image is shown, along with the three separated classes. The visual results are good, with clear separation of the text, headline and picture areas. The picture (including background) region contains very little residual main or headline text. In fact, the most significant errors are in the picture area, where two small regions are misclassified as text and headline. Upon close inspection of

figures 6.9(c) and 6.9(d), it can be seen that the misclassified regions appear remarkably similar to headline and text, respectively. It must once again be stressed that the result in figure 6.9 are obtained from texture information alone, with no auxilliary information used in the process. Therefore, it is entirely unsurprising to see the errors present in figures 6.9(c) and 6.9(d), given the visual similarity of the error regions in question.

6.4 Applications Summary

This chapter illustrated several typical examples for texture segmentation in real-world applications. The modified K-means algorithm was used to achieve the segmentation results showed in this chapter. In these examples, the modified K-means clustering algorithm on wavelet texture features produced visually satisfactory results. In all the cases, the major constituent textures in the images are correctly identified. It was found that, on average, the segmentation accuracy is inferior to the artificial mosaics used in the experiments in chapter 5. This finding is unsurprising, given the complexity of information in real images and difficulty of dealing with complicated boundaries. In particular, the use of texture information alone to segment real images, such as those in a scene understanding application in section 6.2, is generally inadequate. In a fully functional computer vision system, other information must be incorporated. Knowledge about the expected shapes of objects would greatly assist the correct location of boundaries, which is one of problems experienced with the modified K-means algorithm.

This page is blank

Chapter 7

Conclusions and Future Work

Texture segmentation has been a long-standing problem in image processing and computer vision. The importance of this problem has long been established. Many methods have been developed to solve this problem within specific spheres of interests, and there is certainly room for further improvements. This work addresses the problem from the perspective of an innovative approach, based on wavelet theory and a novel modification of K-means clustering. This chapter provides an overall summary of the thesis, followed by some possible future directions for this work.

7.1 Summary

The wavelet transform is a relatively recent mathematical development that has quickly been widely adopted in many scientific applications. Wavelet transforms provide the analytical backbone of the work in this thesis. Chapter 2 presents a review of the history, theory and practical techniques of applying wavelets to texture analysis. The historical perspective of wavelets provides key insights into their attractiveness for analysis of textures. The focus of our review on wavelets is on their multiresolution analysis and computational properties. These are the primary reasons for choosing wavelets as the analytical tool. Several types of wavelets are discussed and compared. In particular, a relatively new type of wavelets, the Dual-Tree Complex Wavelet Transform, is discussed in detail. With approximate shift-invariance and improved directional selectivity, this type of wavelet transform plays a major role in this thesis.

Without an unambiguous model for describing textures and their attributes, researchers have spent much effort on uncovering descriptive, compact features for describing textures. Chapter 3 explores the role of texture features in a segmentation system, and presents the central techniques employed for the work in this thesis. Firstly, the overall texture segmentation system architecture for this work is defined in chapter 3. This architecture identifies the four distinct stages of processing in a texture segmentation system, and forms the foundation for all subsequent work in the thesis. In discussing texture feature extraction, an extensive review of the numerous existing feature extraction methods is provided. In this review, the main statistical, structural and filter-based texture analysis methods in the literature are described and compared. Collectively, these techniques provide the background for the introduction of a novel feature extraction method, which is discussed in detail in section 3.3. This novel feature extraction is based on DWT coefficients of a digital image, extracting feature vectors directly from the wavelet transform subbands. The technique is later extended to embrace DT-CxWT coefficients as well. It has long been established that features extracted from filter-based schemes are much more effective if suitable post-processing is applied. The collection of post-processing steps and techniques is termed the feature conditioning stage. A range of techniques have been surveyed for this stage, but three particular ones - normalising, windowing and non-linear transformation - are identified and used for this work. In particular, windowing is expected to yield large benefits for texture segmentation. The smoothing effect of using spatial windows of various shapes and sizes is desirable for segmentation applications. The effectiveness of windowing is consistent with the texton theory of textures, which describes textures in terms of primitives.

The final stage in the segmentation system is clustering. In this stage, the extracted and conditioned texture features are grouped together into different clusters to produce the final segmentation. There are many clustering techniques in the pattern recognition literature. Chapter 4 describes a few common ones, with most attention being focussed on the K-means algorithm. This particular algorithm is used as the main clustering mechanism in the thesis; it has long been a staple in pattern recognition problems, and is simple and well-understood. The modular architecture of the texture segmentation system means that alternative clustering methods can be directly substituted for the K-means algorithm. Other examples of classifiers or clustering methods described in this chapter are: fuzzy K-means, neural networks and Support Vector Machines (SVM).

With each element in our texture segmentation system having been discussed, the system is tested through extensive experiments. Chapter 5 contains the detailed results from the experiments performed. The results are separated into two main sections. Firstly, a series of experiments designed to measure the separability of extracted (and conditioned) feature sets are presented. The experiments are performed on a set of artificial image mosaics, which is illustrated in its entirety in Appendix A. When selecting the test set, the images are chosen from publicly available repositories on the World Wide Web, so as to maintain generality and comparability with existing published results. The results from the separability experiments provide rudimentary indications on the likely success of segmentation, based purely on the separability of feature sets. This serves as a validity test for the feature extraction and conditioning methods described in Chapter 3. It has been demonstrated, through three different measures - spatial separability ratio, feature contrast and distance histograms, that the extracted feature sets have good potential to be separated. In addition, DT-CxWT based texture features demonstrated greater separability, particularly in the distance histograms, than DWT derived features. Once the separability of the extracted features are established, full segmentation experiments are performed on the test images, using both conventional and fuzzy K-means clustering. In all cases, the ground truths are known, but these are only used for evaluating the error rates of the segmented images. With both conventional and fuzzy K-means clustering algorithm, there is no need for explicit training of the classifiers. As a general trend, it has been noted that the results of these experiments showed good performance for simple texture mosaics containing two textures, but failed to extend the good performance for more complicated mosaics with a greater number of textures.

It is a reasonable requirement for a texture segmentation system capable of processing real-world images to be able to segment multiple textures. The segmentation system with K-means clustering is clearly unsuitable. The separability measurements of the texture features indicate that the features are separable, even for multiple clusters, which implies the failings of the original system are likely due to K-means clustering. In response to this problem, a novel modification to the conventional K-means algorithm is proposed. This involves the addition of a spatial location term, and a variable weighting system is used to shift the emphasis between feature similarity and spatial proximity. The resultant modified K-means algorithm is examined in detail through segmentation experiments, performed with the same set of test images as for the previous experiments.

The segmentation performance of the system with modified K-means is greatly improved, especially for multi-texture mosaics, when compared to conventional or fuzzy K-means. The average segmentation accuracy also compares well with other experiments on the same mosaics that are in the literature. As in the previous case, the DT-CxWT derived features perform better than DWT features, at the corresponding transform depths.

A particular issue with the modified K-means algorithm is the need to estimate optimal values of the spatial proximity factor which weights the distances in feature and physical spaces. In the experiments, it has been observed that different values of this factor lead to wildly contrasting segmentation results. This places great emphasis on the need to be able to select appropriate values of this particular parameter. A possible technique to address this problem, which is based on measuring distortions of a segmented image, discussed. The justification for this technique is by understanding the role of the proximity factor in the modified K-means clustering algorithm. This method has been applied to some examples to yield approximations of optimal values of the proximity factor. While this technique cannot pin-point the optimum with precision, its main utility is in narrowing the range of suitable values, and thus assisting in the search for a global optimum.

With the good performance of wavelet texture features and modified K-means clustering for test mosaics, the segmentation system is tested with some real-world examples where texture segmentation can be useful. Our segmentation system performed reasonably well in all the cases. The results are encouraging, but it also reinforces the notion that texture is only one of many important visual cues that a successful machine vision system must be able to process, if it is to emulate the power and flexibility of biological systems.

7.2 Future Directions

There are several directions in which this work can be advanced further:

7.2.1 Smart Feature Selection

The focus of this thesis is on the extraction and conditioning aspects of texture features. However, it has been known for some time that a properly selected set of features can

be very valuable to a segmentation system. The main benefit will be dimensionality reduction; smaller dimensions will simplify the clustering task, as well as lowering the computational effort throughout the system. The feature extraction method described in this thesis produces vectors with 7 to 40 dimensions. However, the intrinsic dimensionality of the texture features may be significantly less. Historically, there have been many ways in which feature selection can be approached. For instance, Principal Component Analysis has been used in signal processing for many years to reduce dimensionality of data into the main components which contain most of the energy. It is interesting to examine the effects of implementing feature selection into the texture segmentation system. In particular, genetic algorithm and simulated annealing are interesting techniques which have been shown to be effective in solving multi-dimensional optimisation problems. It will be a challenge to identify a suitable measure or objective for the selection of features since it is likely that separability, rather than conventional energy or entropy measures, will be most effective for gauging the relative fitness of features. In addition, this thesis concentrated on experiments with wavelet subband coefficient energy as the primary feature. There are many other types of statistics that can be extracted from wavelet transform coefficients, and it remains to be investigated whether multiple statistics methods will lead to improvements in segmentation performance. Future work could look into finding the optimum combination of different statistical measures to be used as texture features.

7.2.2 Classifier Improvements

The use of staple clustering methods such as K-means clustering in this thesis is mainly for simplicity. It was desirable to have a simple, well-understood method for clustering, so as to allow the focus to be on evaluating the relative merits of different features. Modified K-means was borne out from a desire to improve K-means segmentation performance for multi-texture mosaics. A difficulty with the modified K-means algorithm is the necessity to choose the spatial proximity factor wisely. This thesis introduces a simple indicator based on distortion measures which can assist in choosing the proximity factor, but this technique can only provide a reasonable approximation. It is obviously desirable to have more accurate means to estimate optimal values of the proximity factor. This may be possible with a more comprehensive analysis of the segmentations, rather than the distance-based distortion measure.

More generally, it will be interesting to see whether the application of more sophisticated clustering algorithms will improve the texture segmentation performance of this system. Indeed, the results from this thesis may be used as valuable training data for more powerful classifiers. In Chapter 4, neural networks and support vector machines are mentioned as potential candidates for classifiers. Neural networks have the advantage of being a more mature topic of study, with many of their properties and oddities are known. Support Vector Machines, being a newer technique, has demonstrated a tremendous potential of being able to identify the complexity, and then solve, classification problems. While the implementation costs for large SVMs may currently be prohibitive, the rapid progress of digital hardware will surely make SVMs a practically viable tool for texture segmentation.

7.2.3 Integration with Other Vision Systems

While being an interesting field of study, texture segmentation by itself has limited value, with applications in only very confined environments. The real-world application examples in this thesis reinforces this viewpoint. The logical extension of a successful texture segmentation scheme would be to embed it into full computer vision systems. There are many challenges associated with such an integration. For instance, questions on how to meaningfully fuse information from several visual cue analyses must be answered. A fully flexible machine vision system must also be dynamic over time, adapting to the information content in each scene. As a result, the texture analysis engine of the vision system may also need to adapt over time, and this will pose a significant challenge for future texture segmentation research.

7.3 Closing Remarks

Texture segmentation has been an important computer vision research topic. Its vital role in vision systems is long established, as are its fundamental challenges. Many advances have been made, but the panacea of a correct, powerful and efficient texture segmentation has not yet been realised. This thesis attempts texture segmentation with novel techniques. It is hoped that this thesis represents continued progress in the field, such

that one day, the ideal texture segmentation subsystem for complete machine vision can be achieved.

This page is blank

Appendix A

Texture Segmentation Data

In this appendix, all images used in the texture segmentation experiments are presented. They include the images and their ground truths, whenever they are known.

A.1 Ground Truth Images

Figure A.1. Two-texture ground truth, for the "D" series of texture pairs, shown in figures A.8 to A.11.

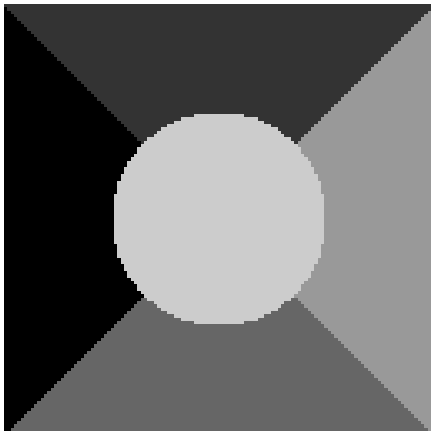


Figure A.2. Five-texture ground truth, for the "Nat" series of mosaics, as shown in figures A.14 to A.19.

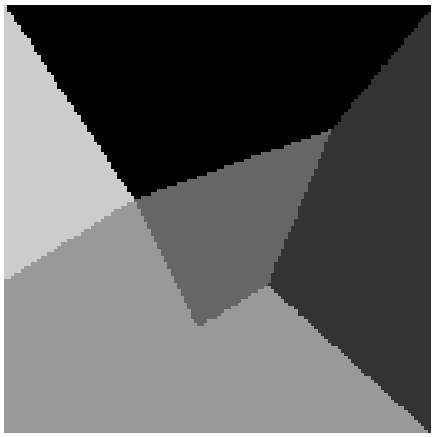


Figure A.3. Ground truth for image "My 5a", shown in figure A.12.

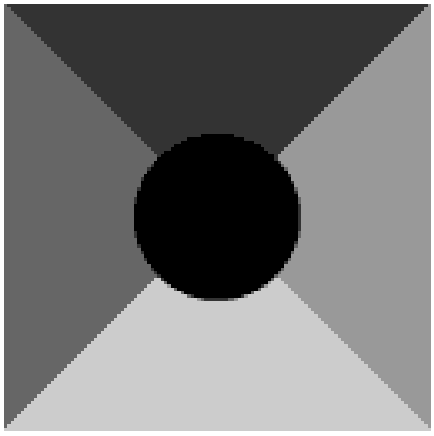


Figure A.4. Five-texture ground truth, for the "bonn" series of mosaics, as shown in figures A.20 to A.119.

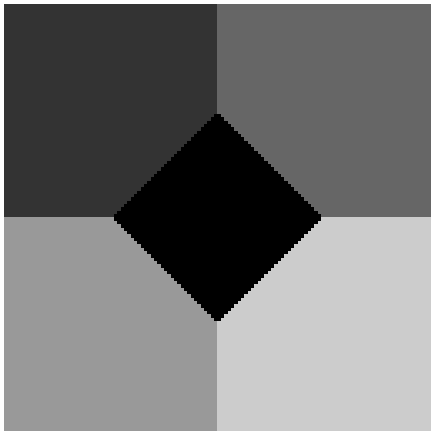


Figure A.5. Ground truth for image "My 5b", shown in figure A.13.

Figure A.6. Ten-texture ground truth, for images "Nat 10" and "Nat 10v", as shown in figures A.120 and A.121.

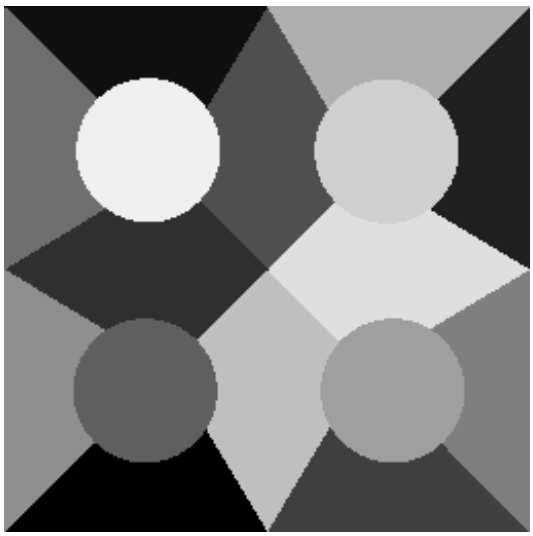


Figure A.7. Sixteen-texture ground truth, for image "Nat 16b" in figure A.122.

A.2 Two-texture mosaics

Images of two-texture mosaics used for experiments. All images have size 256×128 pixels, and are obtained from Randen via the WWW [59]. All images in this section are used for experiments in Randen's publications, and are therefore chosen for direct performance comparisons.

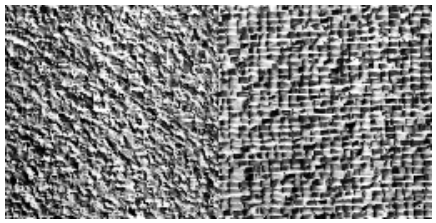


Figure A.8. D4-D84

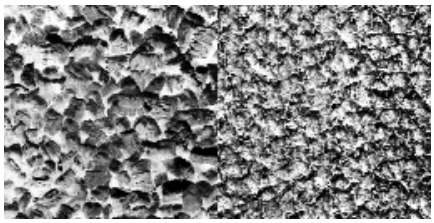


Figure A.9. D5-D92

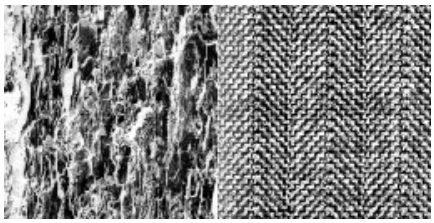


Figure A.10. D12-D17

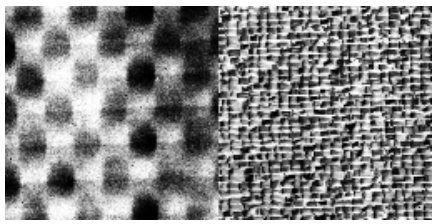


Figure A.11. D8-D84

A.3 Five-texture mosaics

A.3.1 Original mosaics

These two mosaics are constructed by the author; image size is 256×256 pixels. The constituent textures are deliberately chosen to be identical for both mosaics. The only difference between them is the set of boundaries separating the different texture regions.

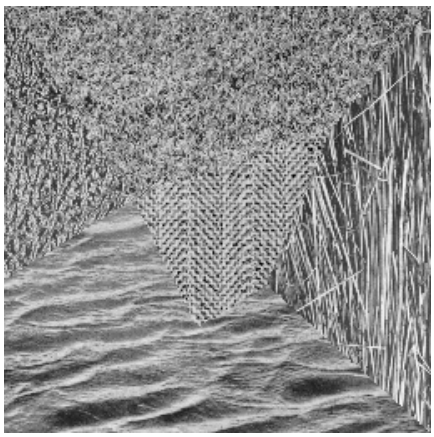


Figure A.12. My 5a

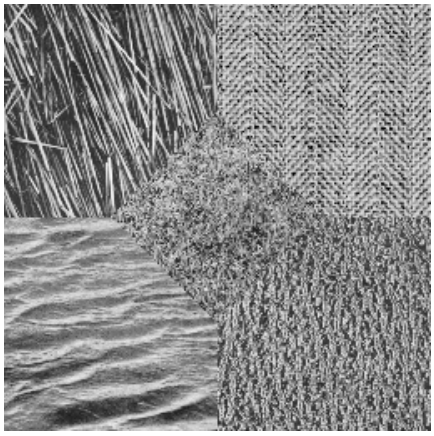


Figure A.13. My 5b

A.3.2 Randen mosaics

This set of six mosaics are obtained from Randen via the WWW [59]. All images have size 256×256 pixels. Many of the images in this section are used for experiments in Randen's publications, and are therefore chosen for direct performance comparisons.

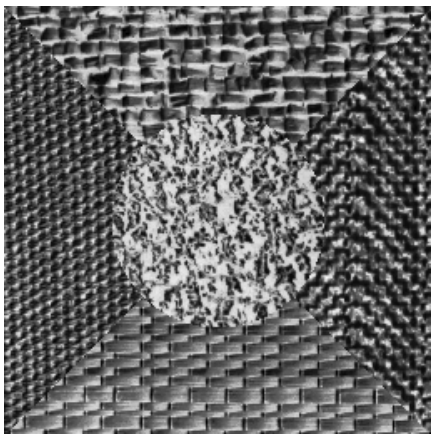


Figure A.14. Nat 5b

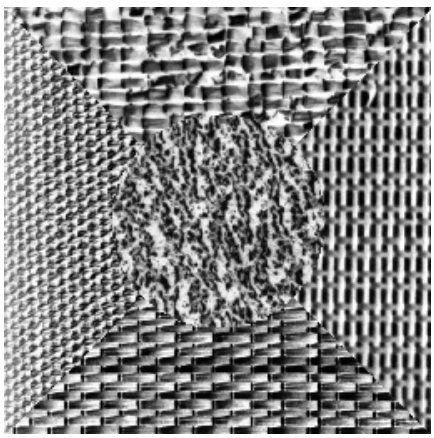


Figure A.15. Nat 5c

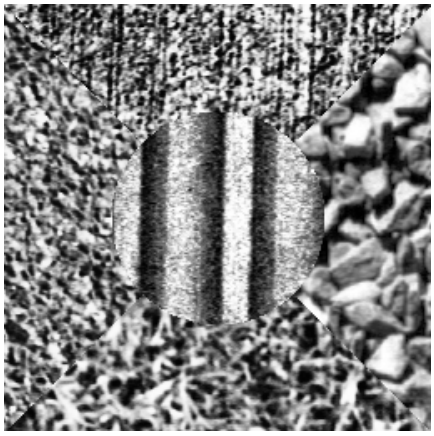


Figure A.16. Nat 5m

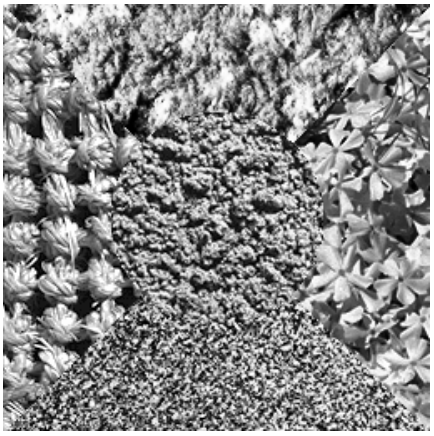


Figure A.18. Nat 5v2

Figure A.17. Nat 5v

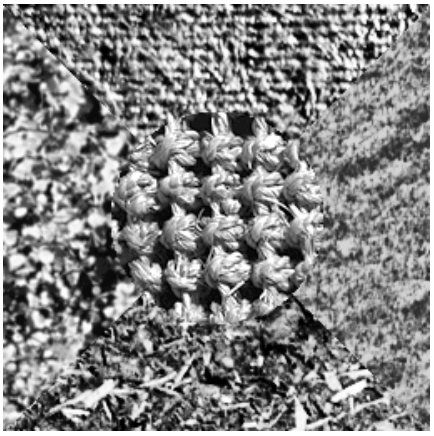


Figure A.19. Nat 5v3

A.3.3 University of Bonn database

Textures downloaded from the University of Bonn database [14]. There are 100 images contained in this database, all generated from the ground truth shown in figure A.1 with Brodatz textures. The mosaics are numbered individually from 0 to 99. All the original images had size 512×512 pixels, but in the interest of speed, they are all reduced to 256×256 for the experiments. Reduction is accomplished with $2:1$ downsampling along both horizontal and vertical directions.

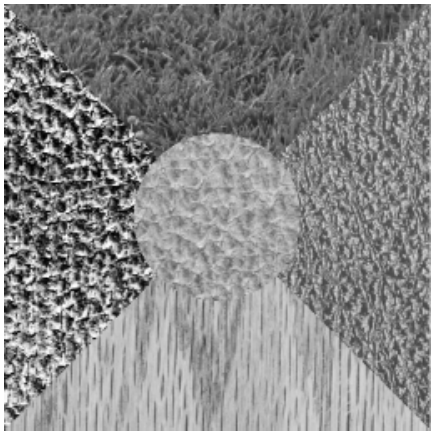


Figure A.20. Bonn 00

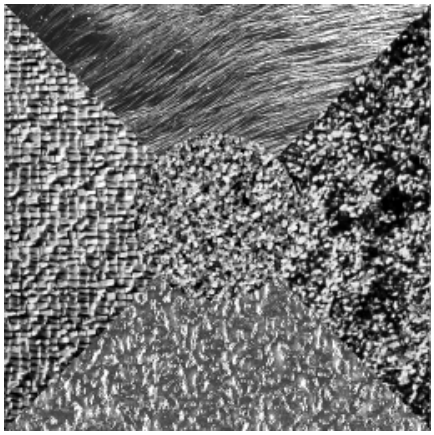


Figure A.21. Bonn 01

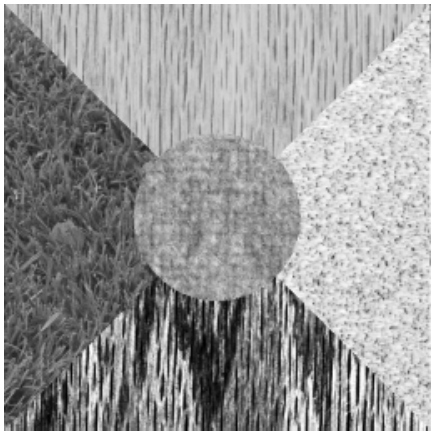


Figure A.22. Bonn 02

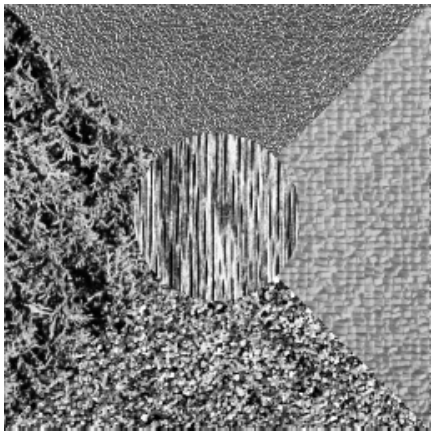


Figure A.23. Bonn 03

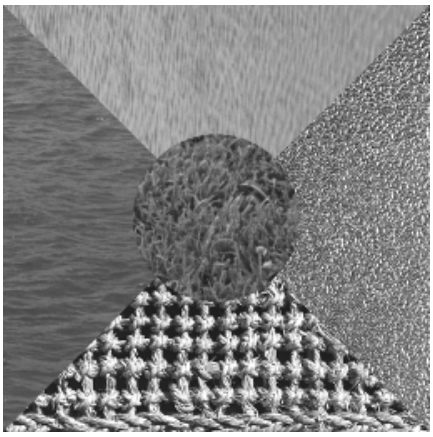


Figure A.24. Bonn 04

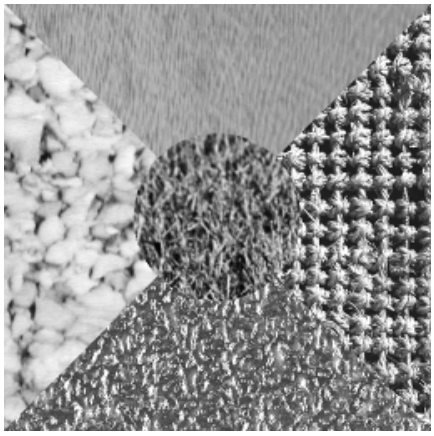


Figure A.25. Bonn 05

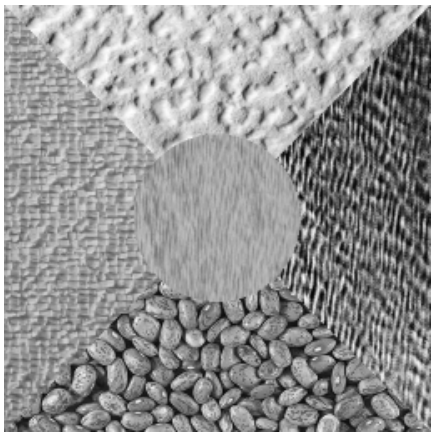


Figure A.26. Bonn 06

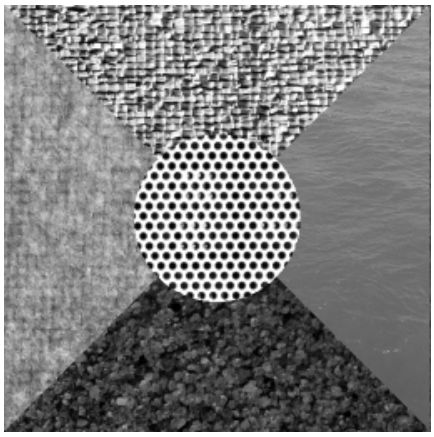


Figure A.27. Bonn 07

Figure A.28. Bonn 08

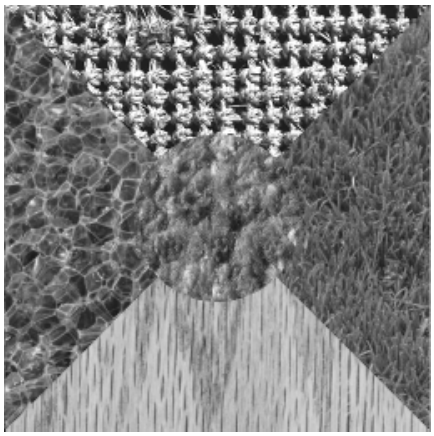


Figure A.29. Bonn 09

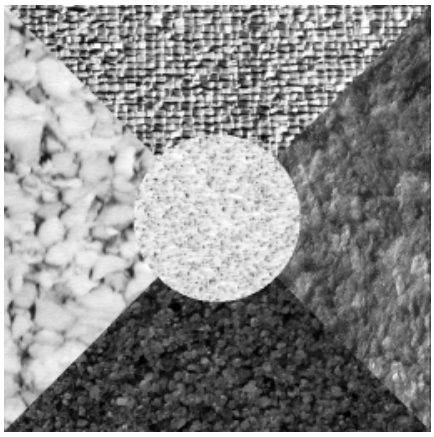


Figure A.30. Bonn 10

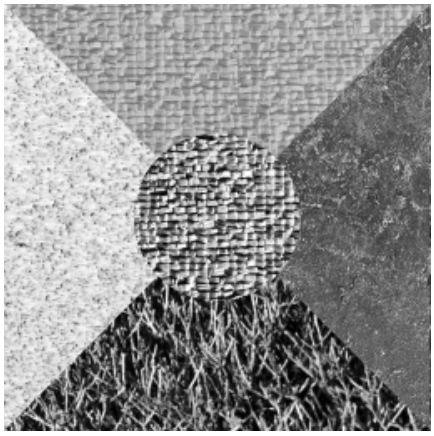


Figure A.31. Bonn 11

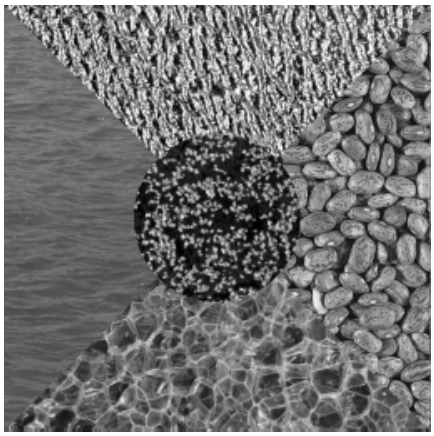


Figure A.32. Bonn 12

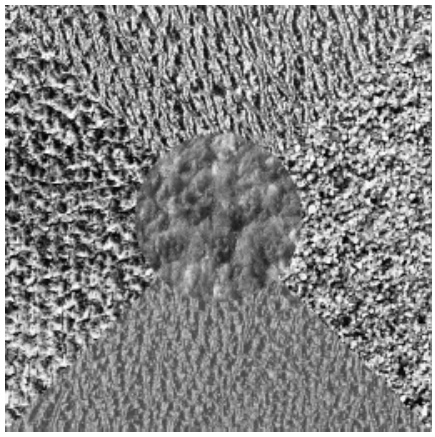


Figure A.33. Bonn 13

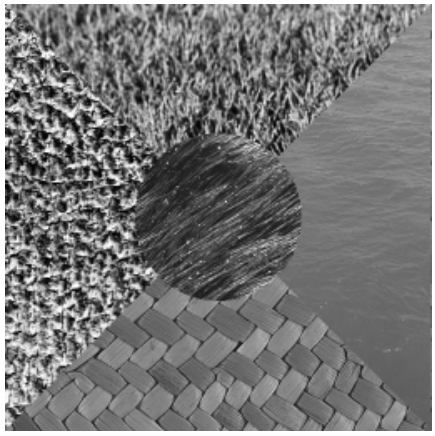


Figure A.34. Bonn 14

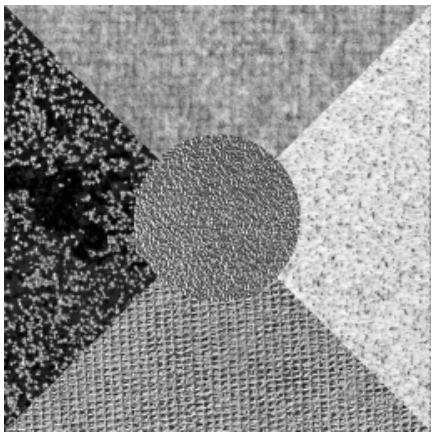


Figure A.35. Bonn 15

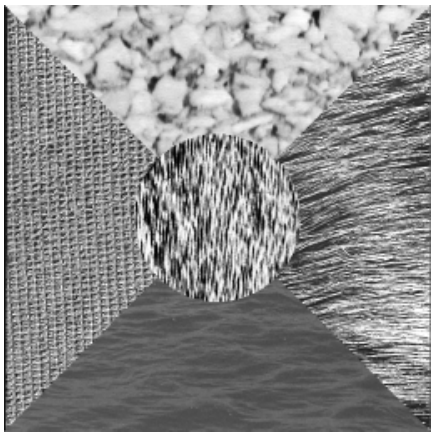


Figure A.36. Bonn 16

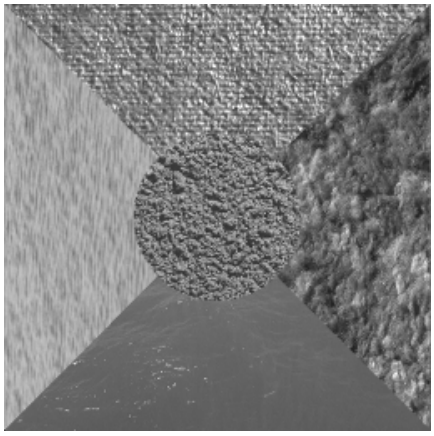


Figure A.37. Bonn 17

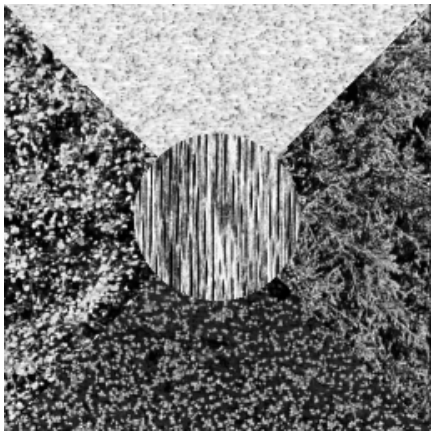


Figure A.38. Bonn 18

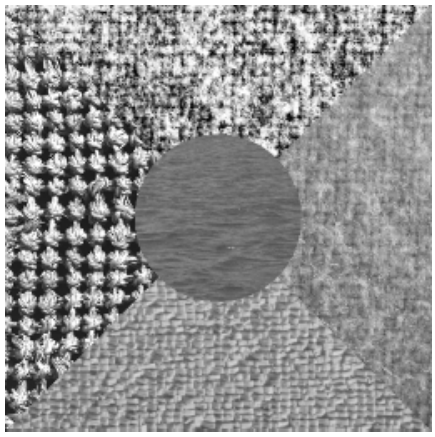


Figure A.39. Bonn 19

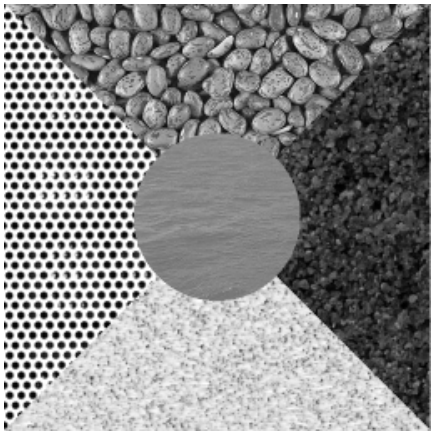


Figure A.40. Bonn 20

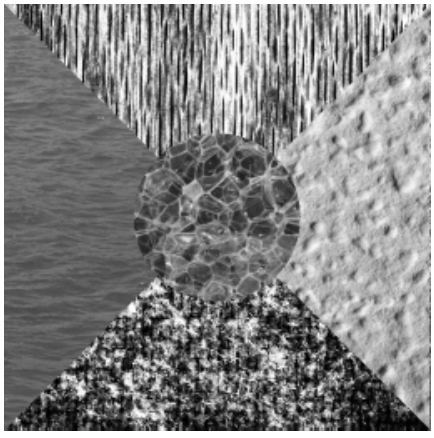


Figure A.41. Bonn 21

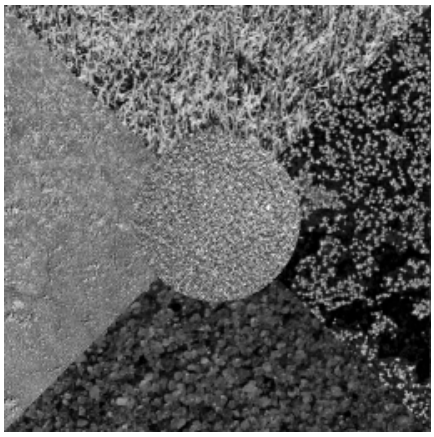


Figure A.42. Bonn 22

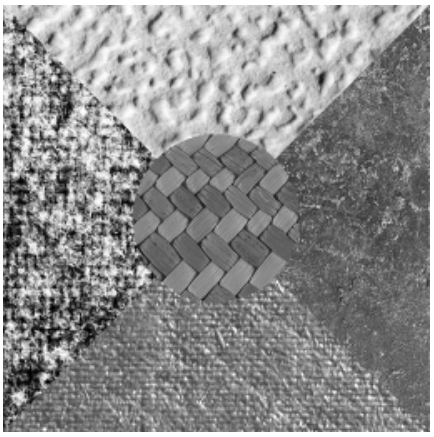


Figure A.45. Bonn 25

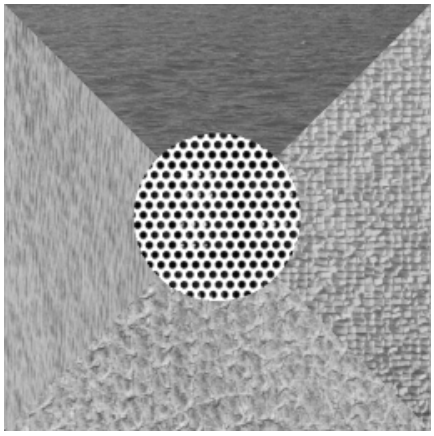


Figure A.43. Bonn 23

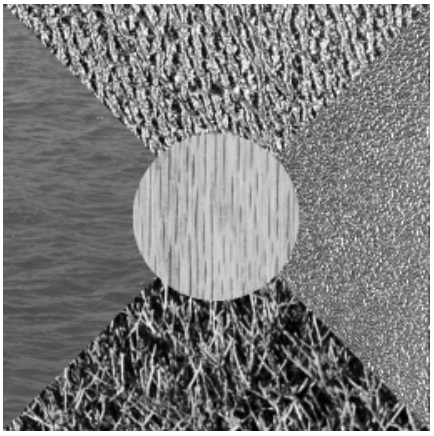


Figure A.46. Bonn 26

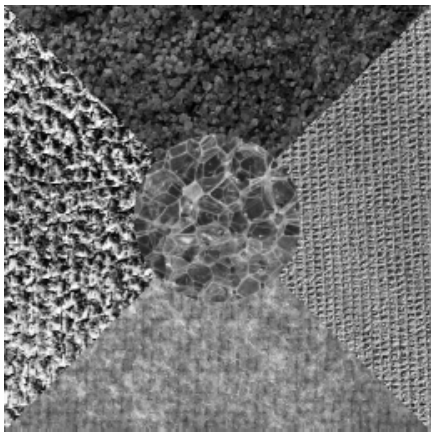


Figure A.44. Bonn 24

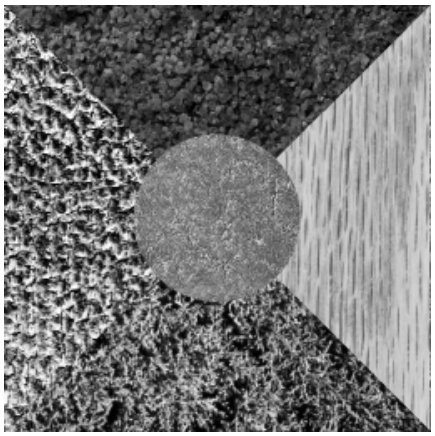


Figure A.47. Bonn 27

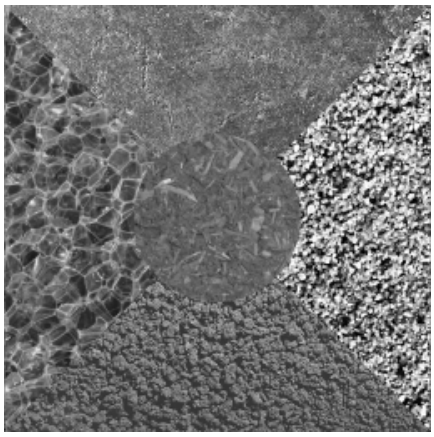


Figure A.48. Bonn 28

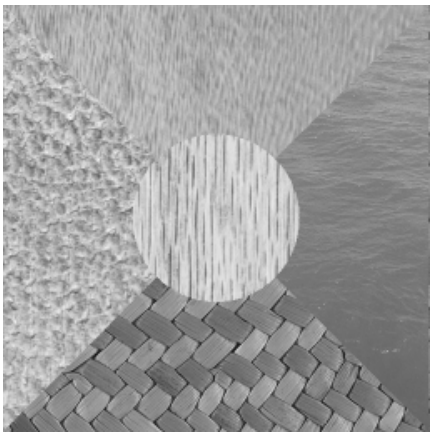


Figure A.51. Bonn 31

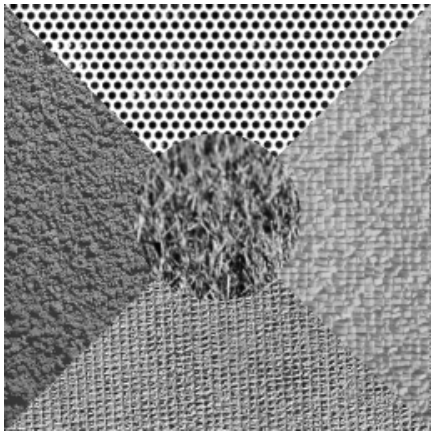


Figure A.49. Bonn 29

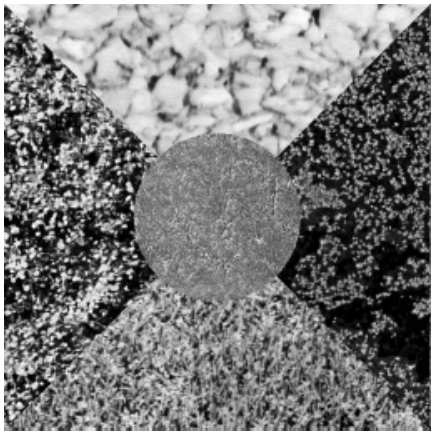


Figure A.52. Bonn 32

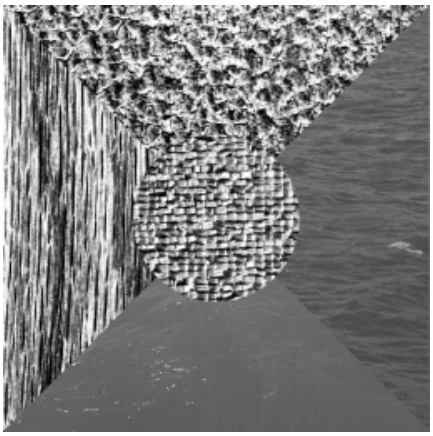


Figure A.50. Bonn 30

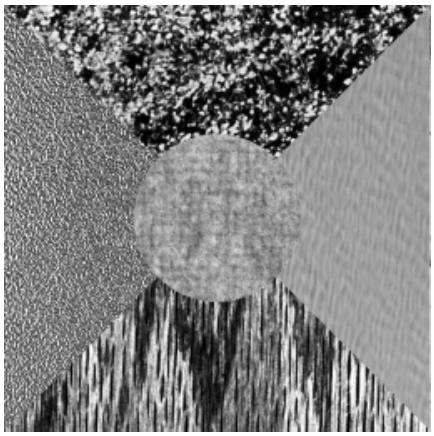


Figure A.53. Bonn 33

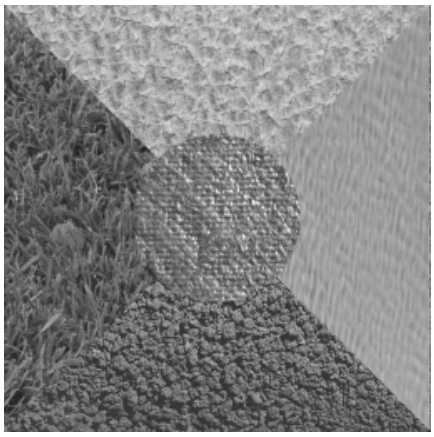


Figure A.54. Bonn 34

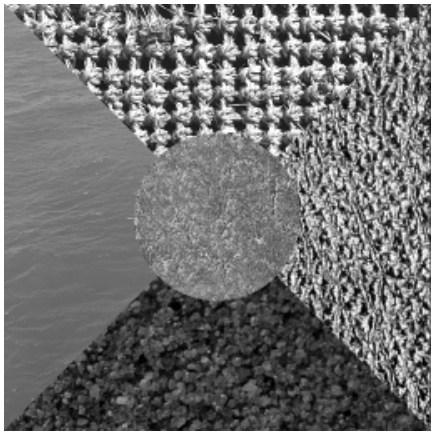


Figure A.55. Bonn 35

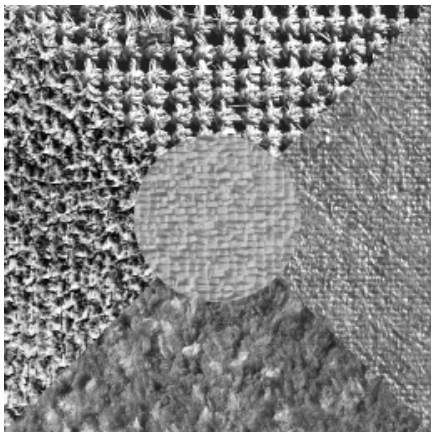


Figure A.56. Bonn 36

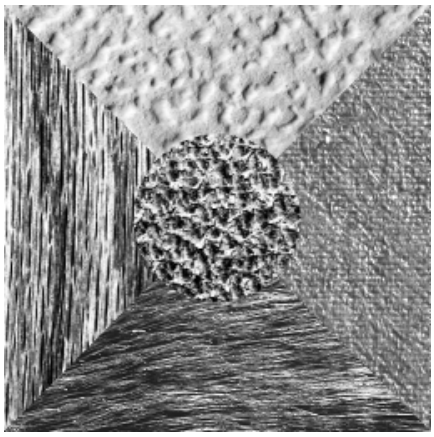


Figure A.57. Bonn 37

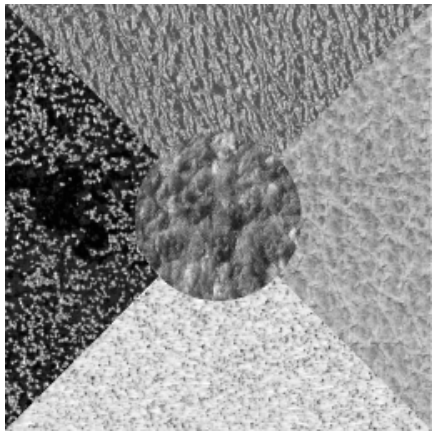


Figure A.58. Bonn 38

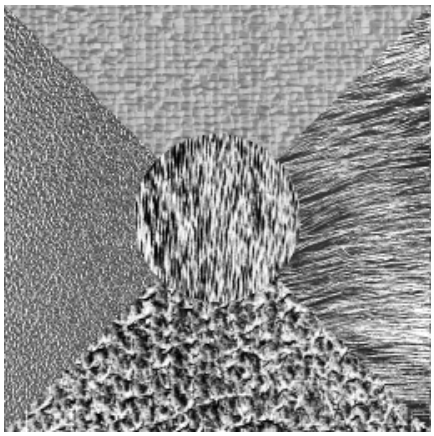


Figure A.59. Bonn 39

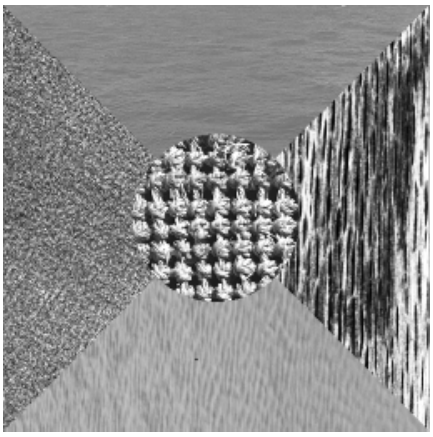


Figure A.60. Bonn 40

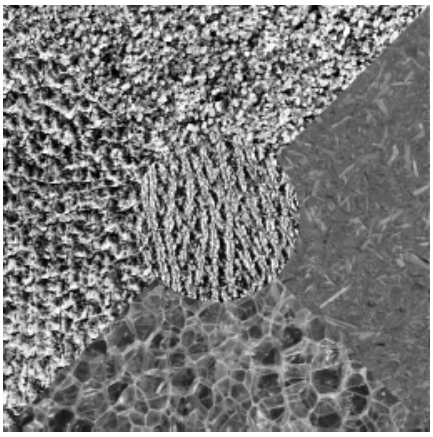


Figure A.63. Bonn 43

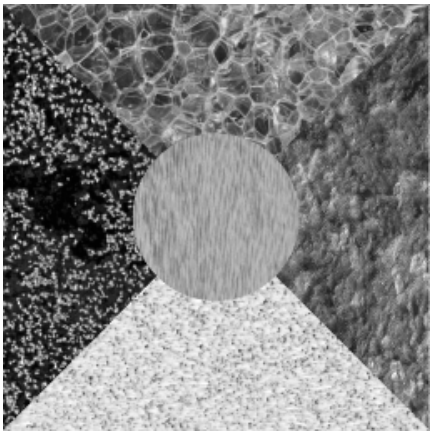


Figure A.61. Bonn 41

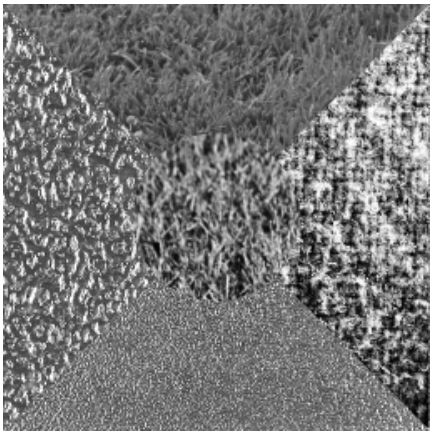


Figure A.64. Bonn 44

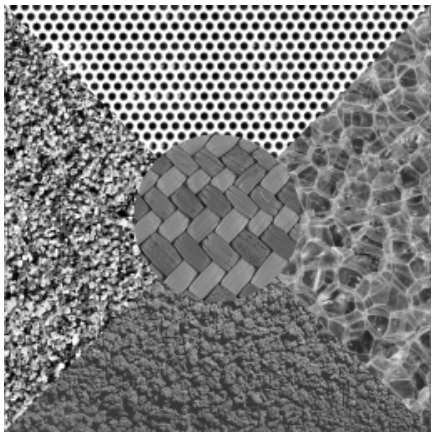


Figure A.62. Bonn 42

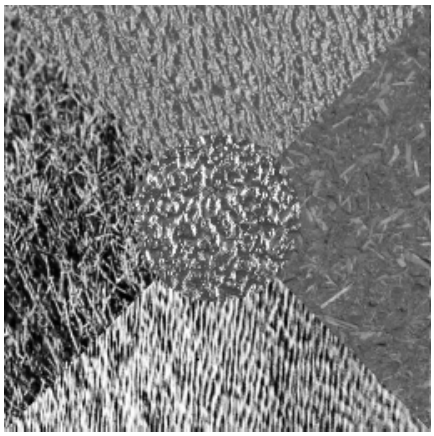


Figure A.65. Bonn 45

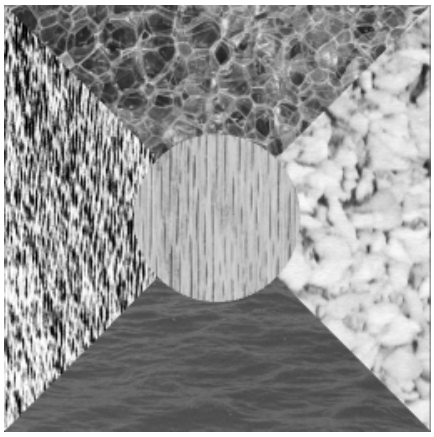


Figure A.66. Bonn 46

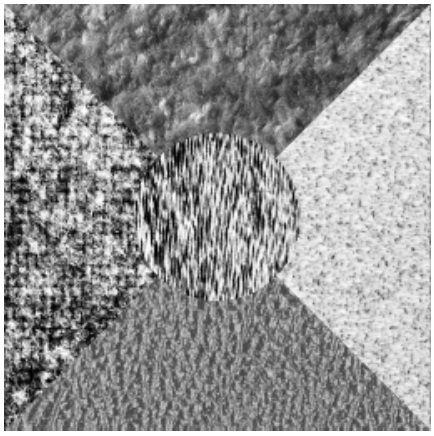


Figure A.67. Bonn 47

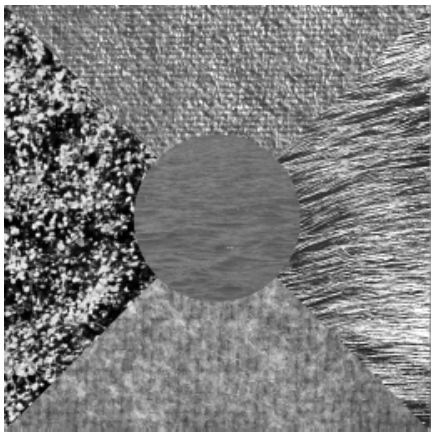


Figure A.68. Bonn 48

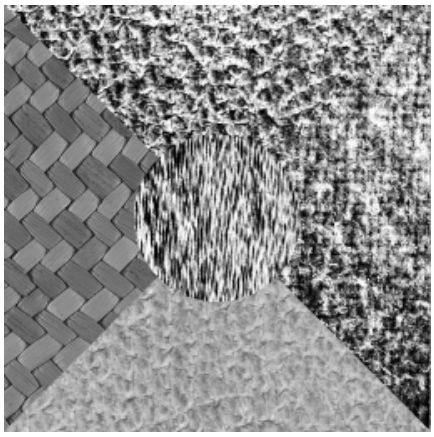


Figure A.69. Bonn 49

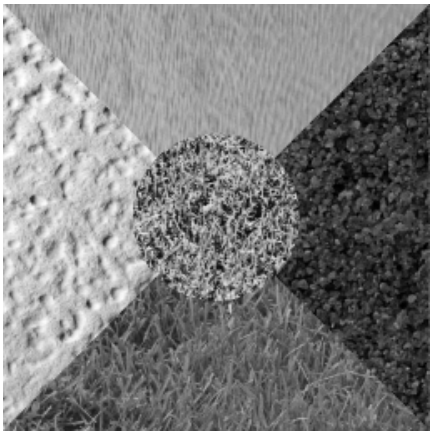


Figure A.70. Bonn 50

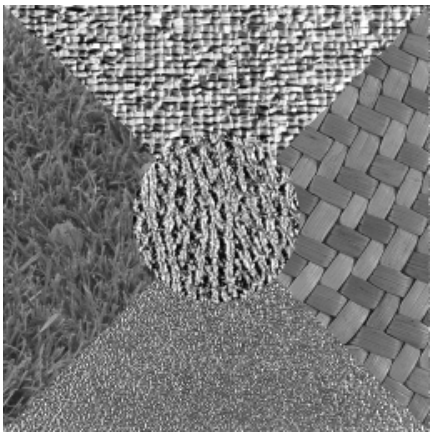


Figure A.71. Bonn 51

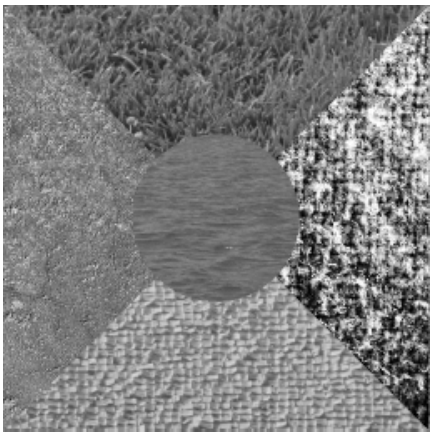


Figure A.72. Bonn 52

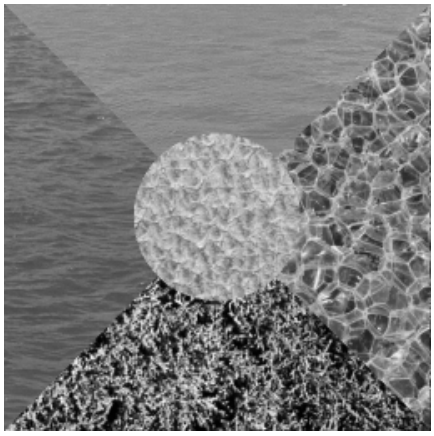


Figure A.73. Bonn 53

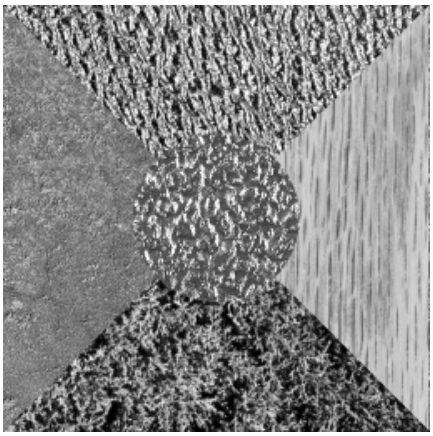


Figure A.74. Bonn 54

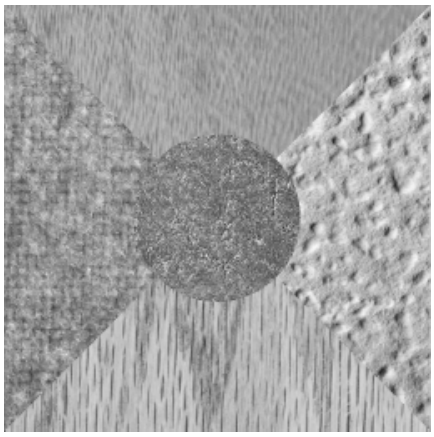


Figure A.75. Bonn 55

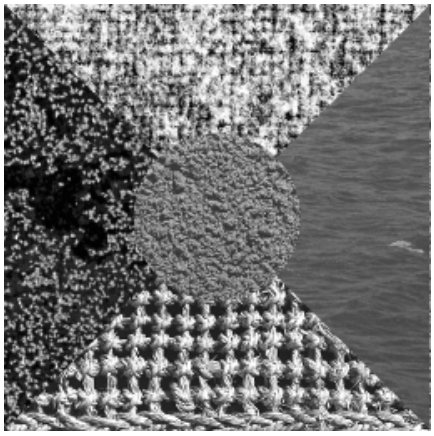


Figure A.76. Bonn 56

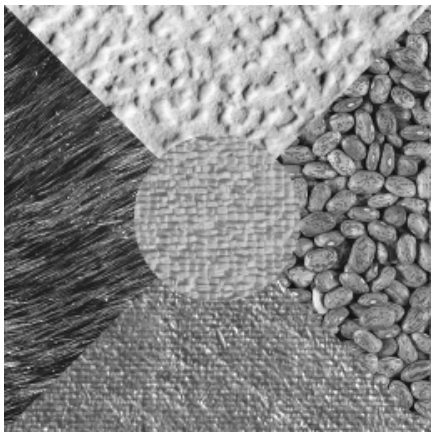


Figure A.77. Bonn 57

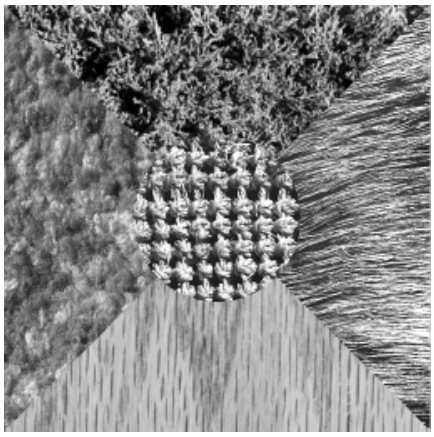


Figure A.78. Bonn 58

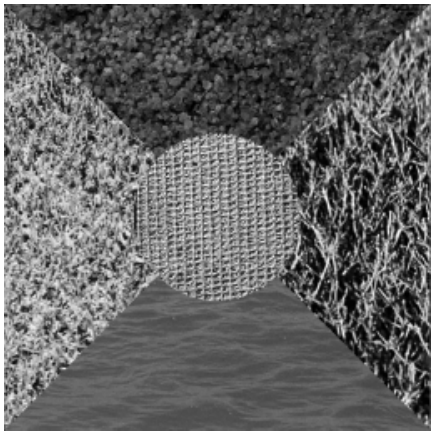


Figure A.79. Bonn 59

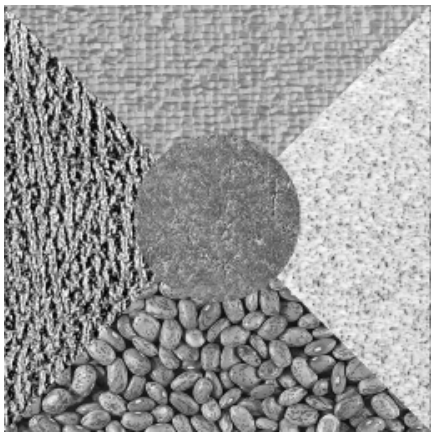


Figure A.80. Bonn 60

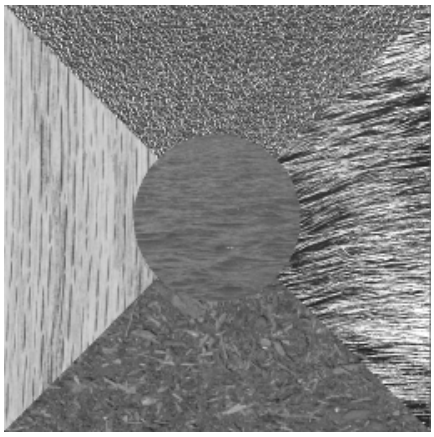


Figure A.81. Bonn 61

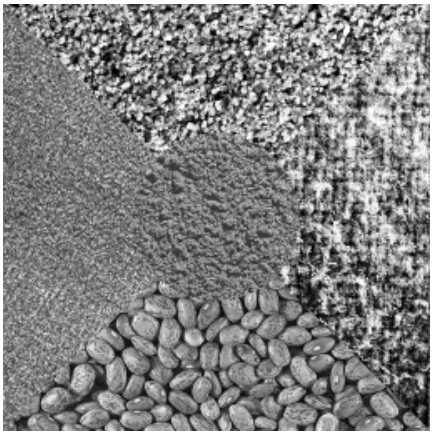


Figure A.82. Bonn 62

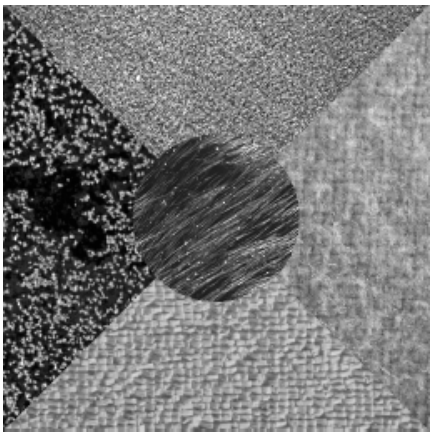


Figure A.83. Bonn 63

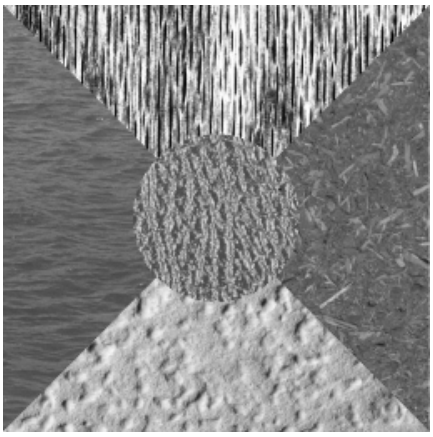


Figure A.84. Bonn 64

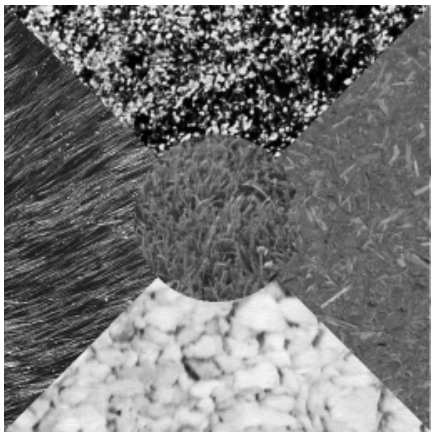


Figure A.87. Bonn 67

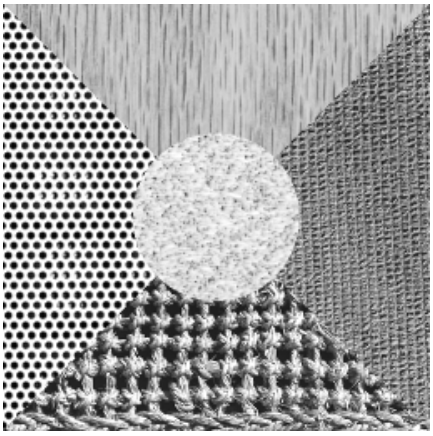


Figure A.85. Bonn 65

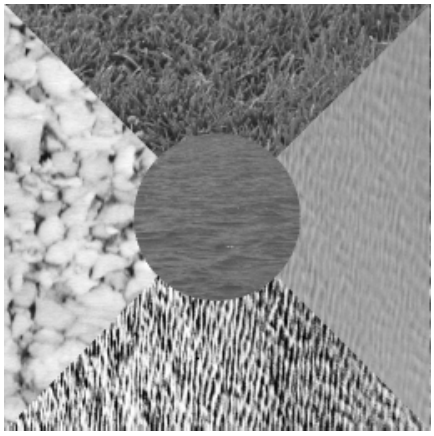


Figure A.88. Bonn 68

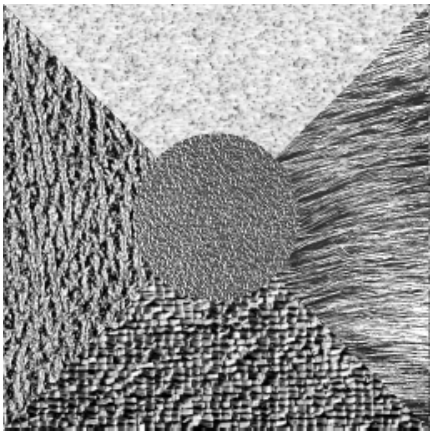


Figure A.86. Bonn 66

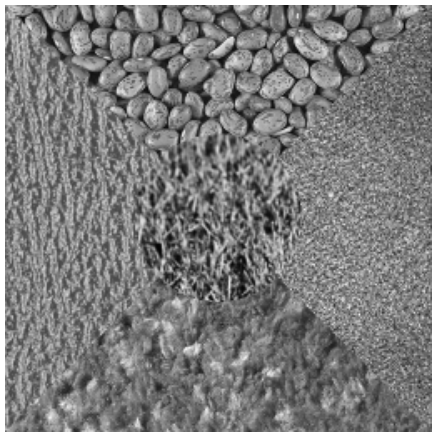


Figure A.89. Bonn 69

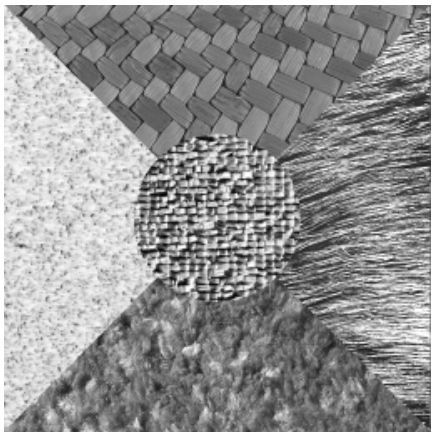


Figure A.90. Bonn 70

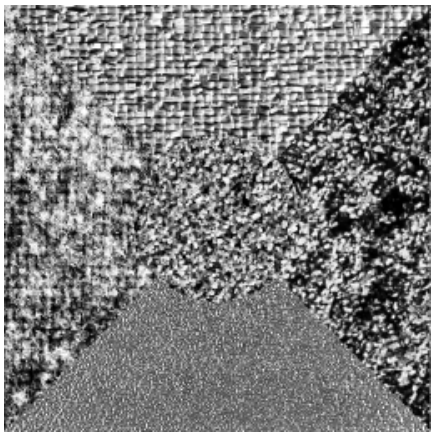


Figure A.93. Bonn 73

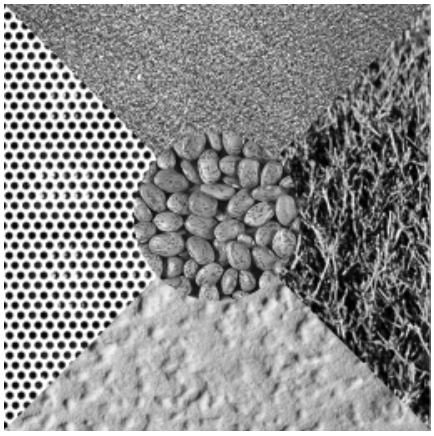


Figure A.91. Bonn 71

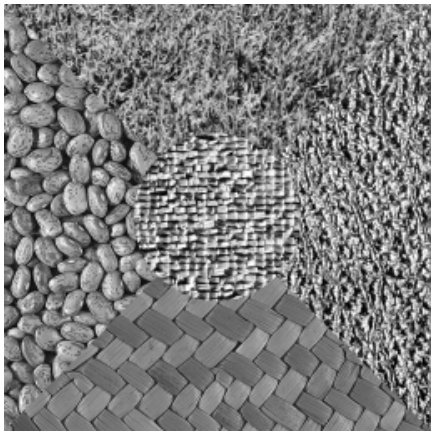


Figure A.94. Bonn 74

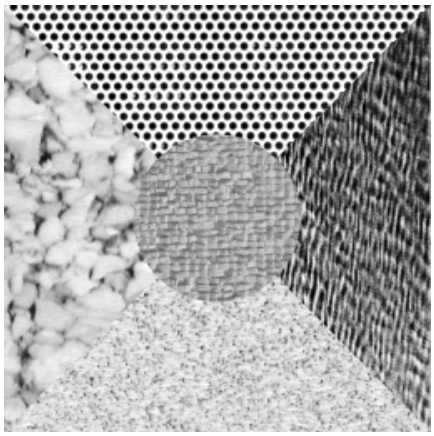


Figure A.92. Bonn 72

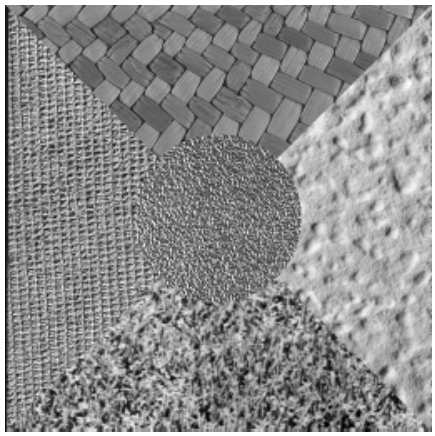


Figure A.95. Bonn 75

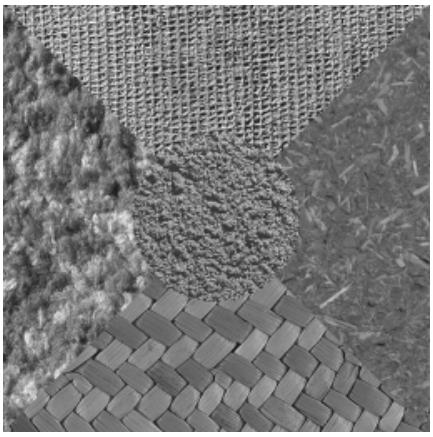


Figure A.96. Bonn 76

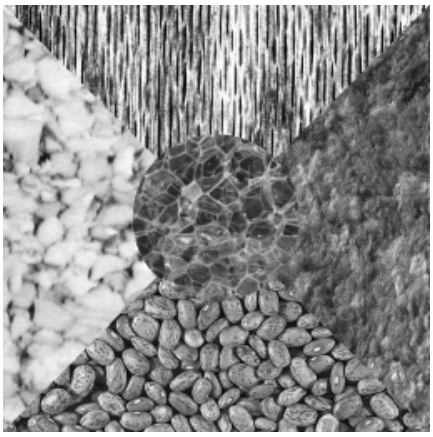


Figure A.99. Bonn 79

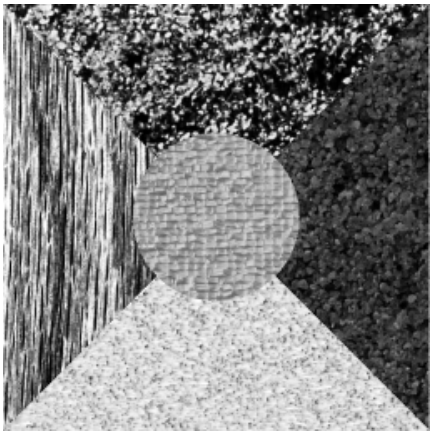


Figure A.97. Bonn 77

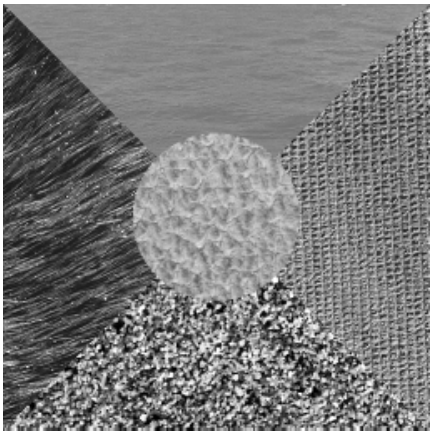


Figure A.100. Bonn 80

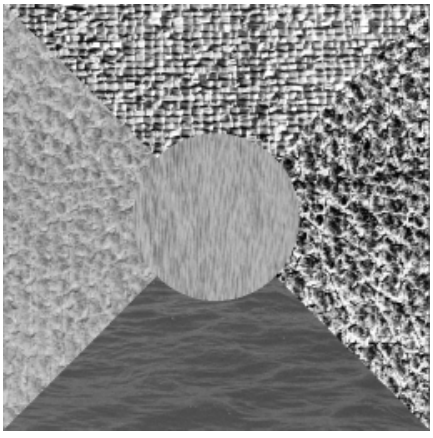


Figure A.98. Bonn 78

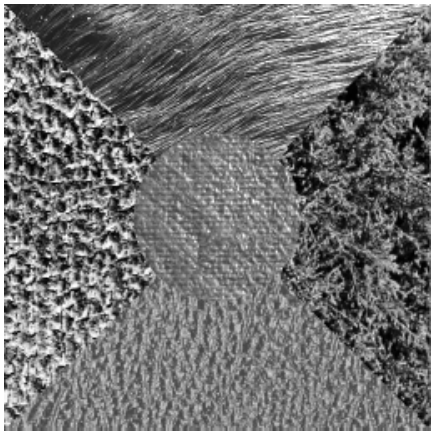


Figure A.101. Bonn 81

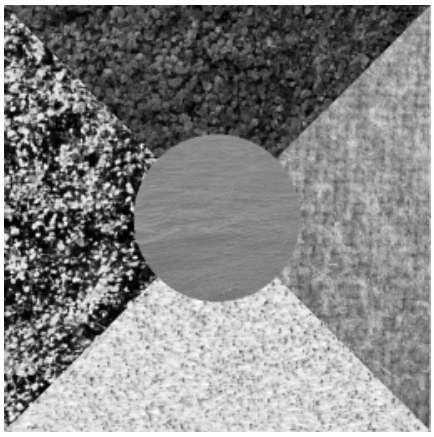


Figure A.102. Bonn 82

Figure A.103. Bonn 83

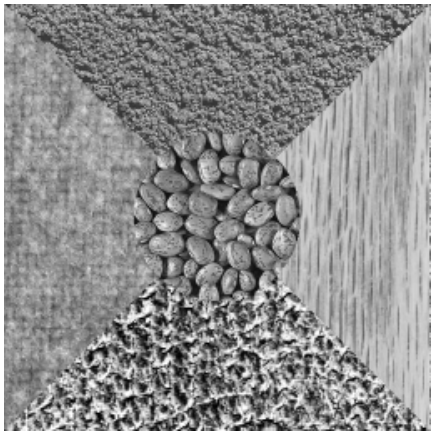


Figure A.104. Bonn 84

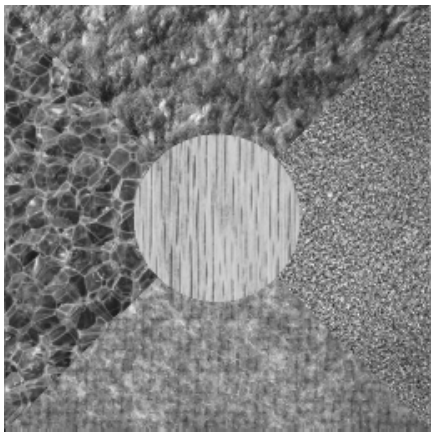


Figure A.105. Bonn 85

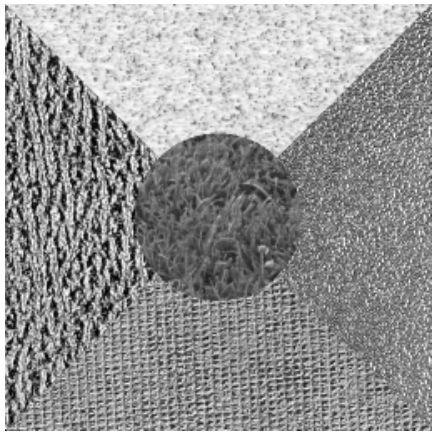


Figure A.106. Bonn 86

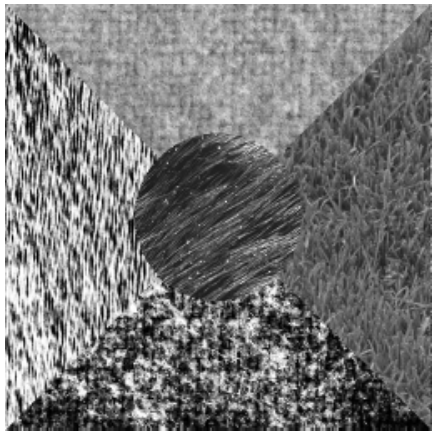


Figure A.107. Bonn 87

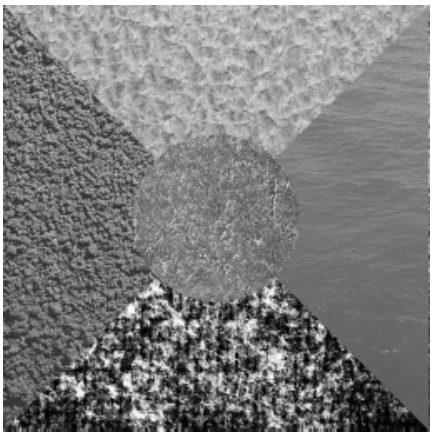


Figure A.108. Bonn 88

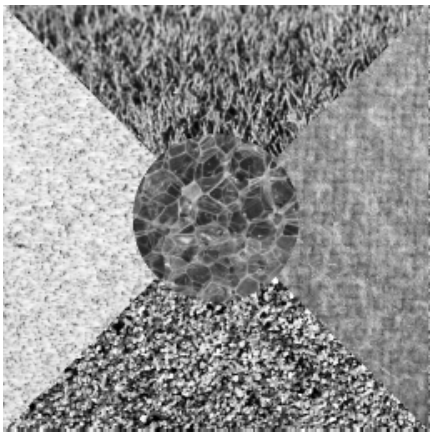


Figure A.111. Bonn 91

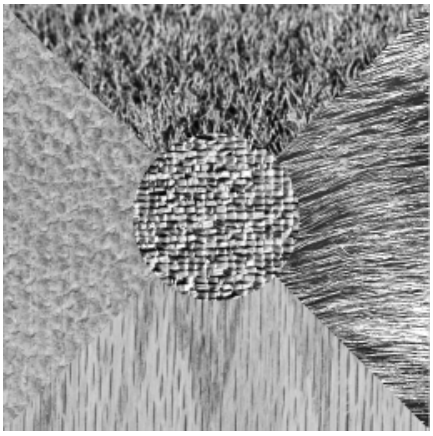


Figure A.109. Bonn 89

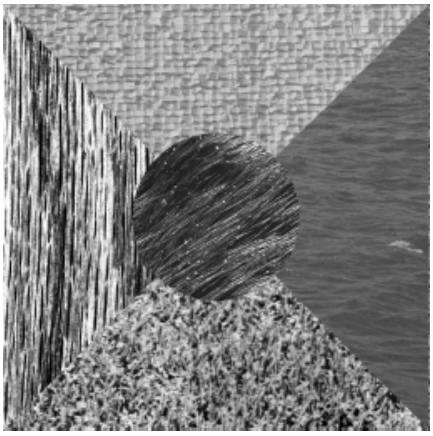


Figure A.112. Bonn 92

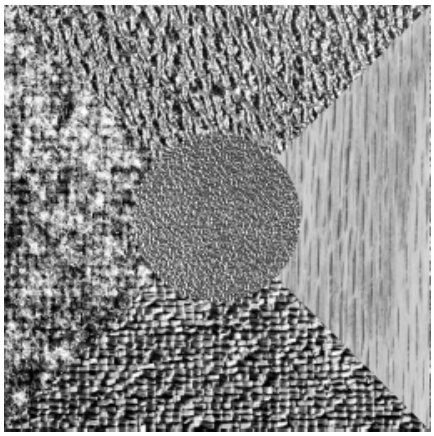


Figure A.110. Bonn 90

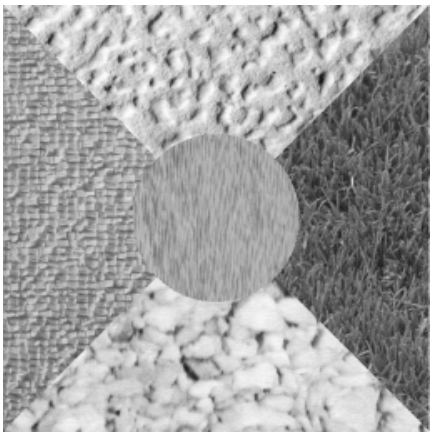


Figure A.113. Bonn 93

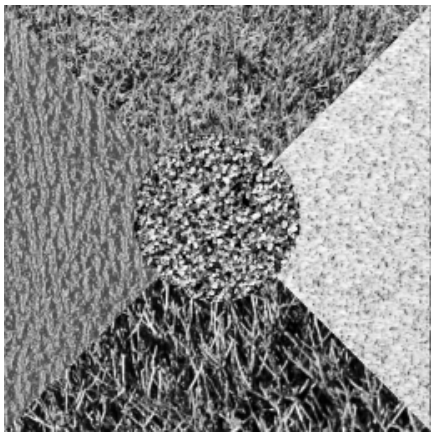


Figure A.114. Bonn 94

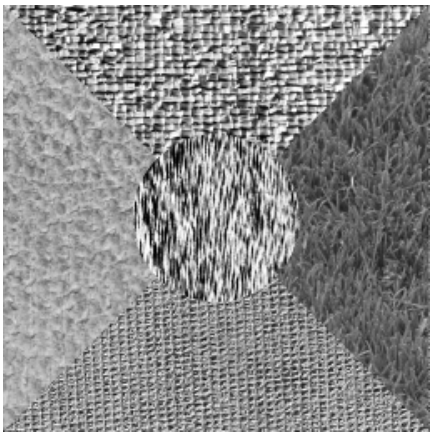


Figure A.117. Bonn 97

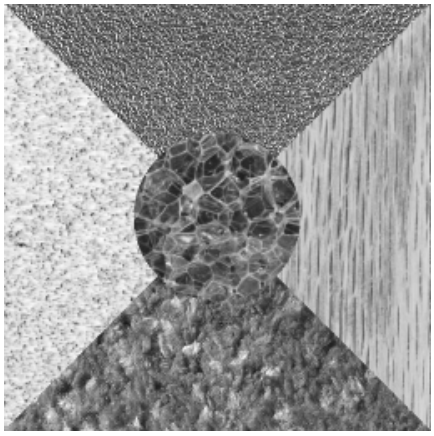


Figure A.115. Bonn 95

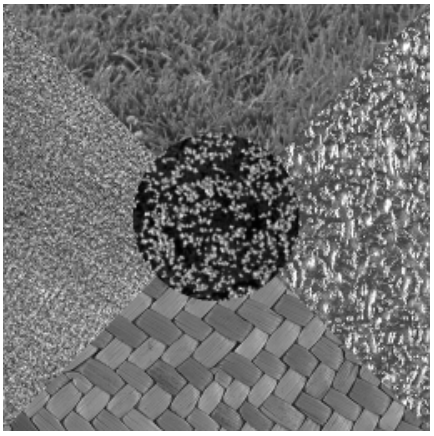


Figure A.118. Bonn 98

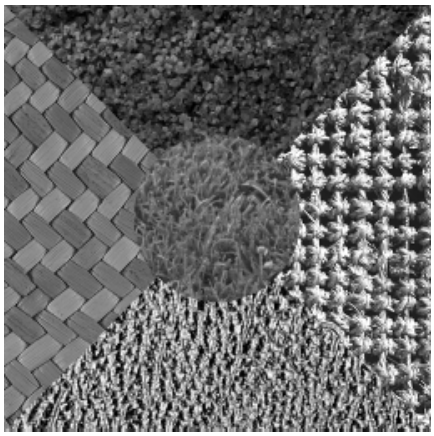


Figure A.116. Bonn 96

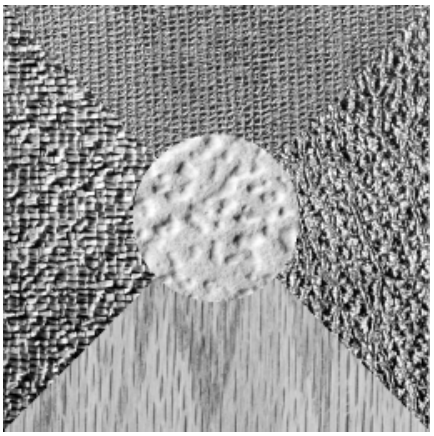


Figure A.119. Bonn 99

A.4 Ten-texture mosaics

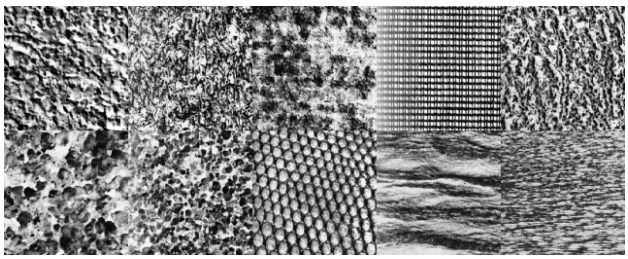


Figure A.120. Nat 10

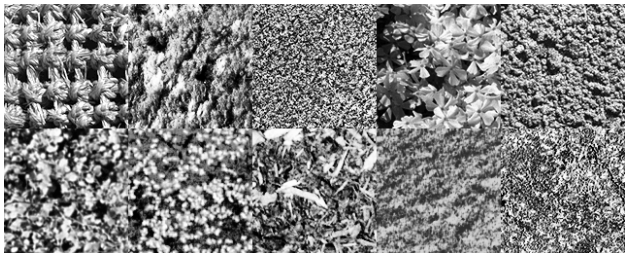


Figure A.121. Nat 10v

A.5 Sixteen-texture mosaics

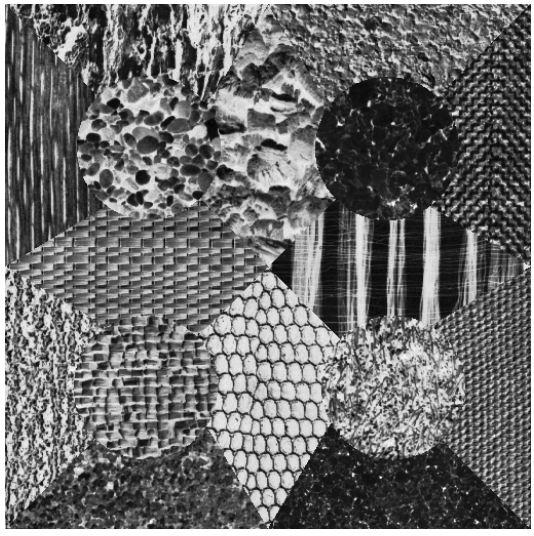


Figure A.122. Nat 16b

This page is blank

Appendix B

Detailed Texture Segmentation Results

This Appendix contains the detailed tables of results for the artificial texture mosaic segmentation experiments. The segmentation error rates for all combinations of wavelet transforms and Kaiser smoothing parameters are listed. However, the median filter smoothing results are omitted. For the sake of brevity, these have been omitted from Chapter 5 of the main text. The four sections of results in this appendix are arranged according to the type of K-Means clustering used in each section. They are listed in the follow order: conventional, fuzzy, modified and modified fuzzy. A general noticeable trend with these results is how poorly some values of the Kaiser smoothing parameter, β, performed.

B.1 Conventional K-Means

Table B.1. Summary of segmentation error rates for K-Means, DWT depth 3. The transform uses the Daubechies 9-7 filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	3.92	3.76	**3.61**	3.81	5.03	7.10	10.13	14.69	20.51	26.18	6.07
D5-D92	2.56	2.55	**2.50**	2.69	4.36	6.51	9.34	12.88	16.10	20.92	5.43
D8-D84	1.11	1.10	0.96	0.92	0.89	**0.84**	0.88	0.94	1.31	2.29	0.95
D12-D17	2.11	2.05	1.86	**1.83**	2.14	2.33	2.67	3.14	3.85	5.11	2.23
My 5a	18.67	26.42	18.75	28.00	27.11	**11.39**	13.90	20.90	22.28	24.09	21.59
My 5b	22.83	**8.37**	8.53	8.70	9.67	23.30	16.07	26.10	27.88	26.07	19.45
Nat 10	**23.10**	31.86	26.09	34.53	33.91	46.34	51.41	56.88	57.75	61.96	40.44
Nat 10v	**32.68**	37.16	38.35	44.77	41.51	46.37	56.72	64.11	67.99	71.77	45.57
Nat 16b	**37.54**	42.24	44.25	46.32	53.38	59.88	67.02	71.47	73.59	76.24	56.63
Nat 5b	**9.54**	9.58	31.94	36.54	39.27	42.24	40.45	46.26	48.50	53.85	39.86
Nat 5c	33.65	**13.51**	13.60	33.34	34.46	35.88	24.17	39.76	41.61	46.54	34.05
Nat 5m	10.93	**10.85**	11.41	35.67	35.47	37.03	38.21	39.84	41.67	46.59	36.35
Nat 5v2	31.58	37.01	**31.09**	32.71	35.49	39.12	52.06	59.30	60.92	62.13	38.06
Nat 5v3	**16.56**	16.72	17.73	21.44	27.05	34.16	42.81	48.82	52.48	57.24	30.61
Nat 5v	**15.20**	15.53	16.67	18.10	21.55	24.98	28.77	32.96	48.75	51.29	23.27
bonn 00	20.57	20.27	19.59	**19.42**	46.34	26.67	26.06	27.09	27.38	49.66	26.37
bonn 01	24.61	**24.41**	25.85	25.81	27.34	32.18	31.30	36.29	41.67	51.90	29.32
bonn 02	37.00	36.75	47.56	36.55	36.91	37.87	40.26	43.57	44.79	**34.27**	37.43
bonn 03	**6.90**	7.09	7.76	8.45	32.60	11.89	43.13	34.24	49.26	51.12	22.25
bonn 04	31.25	**31.18**	31.24	31.54	32.87	33.28	33.98	34.85	37.35	48.00	33.07
bonn 05	11.36	10.89	10.33	**10.19**	29.35	15.91	42.32	43.58	46.05	49.16	22.63
bonn 06	25.67	25.44	25.05	21.50	**5.70**	7.32	28.08	30.92	32.21	44.88	25.56
bonn 07	28.62	28.47	7.62	27.01	35.03	34.23	**5.51**	30.73	34.77	36.59	29.68
bonn 08	**28.46**	28.78	33.56	35.44	42.43	56.11	59.53	66.80	68.90	71.55	49.27
bonn 09	22.61	**22.00**	22.44	26.10	32.37	33.67	39.42	50.90	51.64	54.33	33.02
bonn 10	18.98	**18.96**	19.13	44.89	47.27	50.40	25.78	53.73	51.72	66.90	46.08
bonn 11	53.16	**24.81**	53.05	32.35	33.87	56.92	34.75	58.12	36.21	60.89	44.63
bonn 12	8.34	**8.33**	8.36	8.98	10.82	14.38	20.79	28.06	41.06	45.50	12.60
bonn 13	22.36	22.44	23.33	31.76	32.28	**16.38**	35.93	38.04	45.38	44.35	32.02
bonn 14	29.91	29.80	29.47	33.47	33.34	33.09	**12.38**	17.58	38.22	40.75	31.50
bonn 15	31.88	31.59	5.65	**5.58**	25.44	25.68	35.19	26.94	27.61	42.05	27.28
bonn 16	**6.62**	6.67	6.83	6.95	7.63	8.58	10.64	13.16	15.89	19.62	8.11
bonn 17	**28.09**	50.75	43.02	38.03	38.56	41.17	43.08	43.96	46.46	48.67	43.05
bonn 18	33.54	**20.76**	34.47	34.75	35.35	37.19	37.34	38.76	42.32	48.04	36.27
bonn 19	47.97	**27.87**	28.63	29.88	31.86	39.58	36.50	37.18	38.56	59.66	36.84
bonn 20	42.42	42.23	40.09	50.73	55.81	56.96	48.90	55.37	**36.87**	56.57	49.81
bonn 21	**11.37**	11.51	11.68	12.04	12.62	13.33	26.49	29.67	34.24	44.19	12.97
bonn 22	17.93	28.22	34.84	**15.70**	35.03	29.25	49.20	51.87	54.77	57.09	34.94
bonn 23	31.22	31.02	30.72	30.52	29.95	30.71	25.62	**13.43**	32.01	23.75	30.62
bonn 24	27.51	27.92	**11.78**	29.17	39.62	38.79	39.76	41.09	49.33	49.52	39.21
bonn 25	21.34	**7.71**	8.03	23.49	25.80	17.56	29.20	31.87	34.47	40.06	24.64
bonn 26	17.99	18.23	27.55	23.16	23.80	25.03	26.34	**13.01**	30.06	35.47	24.42
bonn 27	31.49	30.74	**28.92**	31.00	32.01	30.32	31.64	36.06	39.43	59.17	31.56
bonn 28	24.69	**24.55**	36.29	34.99	33.37	36.10	37.41	41.83	56.01	61.04	36.19
bonn 29	6.53	6.30	5.87	**5.50**	5.55	5.85	6.91	8.58	27.92	39.40	6.41
bonn 30	24.47	24.39	**24.15**	25.81	32.62	36.28	37.34	38.18	38.79	40.17	34.45
bonn 31	28.31	**28.14**	28.83	34.99	36.10	37.05	39.13	42.18	45.21	50.36	36.57
bonn 32	**33.04**	33.37	36.79	34.78	33.31	34.12	36.21	42.90	43.99	44.12	35.50
bonn 33	18.39	18.55	**15.73**	19.98	21.11	22.97	24.92	26.34	27.19	28.62	22.04
bonn 34	14.84	14.56	**14.14**	14.76	30.09	31.01	39.58	37.35	39.89	46.95	30.55
bonn 35	6.07	5.90	**5.49**	16.36	5.51	6.12	25.72	26.50	29.22	32.04	11.24
bonn 36	24.40	24.71	**12.23**	26.97	33.91	35.18	37.16	61.70	41.78	62.63	34.54
bonn 37	4.28	4.17	3.99	**3.86**	4.00	4.33	5.19	6.86	9.34	12.72	4.31
bonn 38	33.25	33.06	**9.06**	10.65	24.11	26.21	26.40	50.21	38.02	50.57	29.73
bonn 39	2.78	32.90	33.42	**2.38**	33.81	33.87	34.05	35.94	9.69	15.46	33.16
bonn 40	28.11	**28.09**	32.88	33.48	33.44	33.76	34.17	34.50	31.29	32.42	33.16
bonn 41	34.66	34.83	**34.43**	34.55	35.74	37.01	39.73	41.48	44.23	48.10	36.37
bonn 42	19.31	19.08	19.01	**18.93**	45.60	24.75	53.51	46.59	30.47	35.68	27.61
bonn 43	24.29	24.04	23.21	**17.41**	22.78	28.81	29.22	48.54	50.22	53.76	26.55
bonn 44	24.24	24.21	**9.55**	10.14	13.99	32.32	33.35	34.34	32.72	36.27	28.28
bonn 45	14.98	14.62	**14.04**	26.17	15.08	43.76	46.87	50.23	49.96	52.68	34.97
bonn 46	**28.62**	28.85	29.96	31.34	34.03	37.88	40.30	53.93	51.35	51.90	35.96
bonn 47	11.40	**11.33**	28.32	31.96	32.58	32.97	33.92	58.71	37.35	40.11	32.77
bonn 48	**25.62**	25.76	25.90	25.79	25.82	26.47	27.31	28.88	31.44	38.75	26.18
bonn 49	**28.42**	28.74	29.78	33.52	42.55	41.59	43.60	46.44	48.06	48.98	42.07
bonn 50	11.69	**11.68**	12.14	12.86	14.25	16.76	36.82	37.49	29.75	49.67	15.51
bonn 51	24.27	24.32	24.15	**4.51**	32.81	5.40	6.84	32.86	33.98	35.78	24.30
bonn 52	27.99	27.82	**21.68**	28.49	35.05	28.00	29.98	32.78	40.85	43.79	29.23
bonn 53	28.70	28.47	**28.23**	35.49	35.52	35.86	35.94	37.13	39.93	50.18	35.69
bonn 54	9.89	**9.66**	9.72	10.61	37.88	40.66	42.84	49.95	51.20	55.47	39.27
bonn 55	7.70	**7.51**	30.64	8.23	10.18	12.68	15.83	19.05	22.37	27.09	14.26
bonn 56	34.27	**25.30**	26.86	35.87	37.31	38.76	38.73	40.44	41.84	49.33	38.02
bonn 57	**6.54**	6.70	6.93	8.07	10.00	12.95	16.91	30.86	32.64	34.77	11.47
bonn 58	**7.97**	8.04	8.09	8.41	38.79	9.50	10.53	12.46	16.05	41.61	10.02
bonn 59	10.45	10.29	10.30	**10.20**	10.77	11.90	15.10	19.25	23.91	29.74	11.33
bonn 60	**21.57**	48.88	49.13	50.35	55.53	51.97	54.42	30.82	54.90	57.13	51.16

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 61	21.33	21.15	20.75	20.37	24.03	**20.22**	20.50	20.87	22.19	24.96	21.01
bonn 62	25.38	25.42	**23.13**	30.10	32.32	34.36	37.49	41.30	45.14	56.31	33.34
bonn 63	24.81	24.30	23.83	34.19	35.55	48.82	**15.85**	33.79	50.63	51.01	33.99
bonn 64	**14.94**	15.36	18.51	34.64	34.39	33.23	33.45	35.24	43.74	45.72	33.92
bonn 65	21.19	**21.18**	21.40	21.99	23.18	23.52	24.11	24.82	25.41	33.15	23.35
bonn 66	3.30	3.28	3.22	**3.14**	22.43	3.55	24.77	24.78	12.81	17.90	8.18
bonn 67	**19.52**	19.67	19.92	20.01	20.25	22.13	26.72	28.53	31.07	56.08	21.19
bonn 68	**21.64**	21.77	21.96	22.53	23.74	27.45	31.40	35.54	46.80	50.29	25.60
bonn 69	**7.89**	8.17	9.33	11.24	28.75	19.54	23.33	27.34	31.71	38.41	21.44
bonn 70	16.02	**15.93**	16.04	16.61	18.62	40.16	41.30	40.95	43.94	48.05	29.39
bonn 71	19.36	8.33	8.34	**8.25**	8.26	8.82	10.48	14.09	27.14	29.24	9.65
bonn 72	17.47	17.29	18.54	18.58	**4.43**	19.19	19.78	20.65	22.13	33.40	18.88
bonn 73	**20.14**	20.25	20.64	21.59	32.64	40.48	39.98	44.29	50.34	54.52	36.31
bonn 74	6.09	6.04	**5.88**	6.38	8.54	12.06	18.98	27.08	45.46	49.39	10.30
bonn 75	29.69	29.55	48.77	47.08	**21.28**	22.40	48.69	49.32	50.45	51.79	47.88
bonn 76	**10.99**	11.07	11.47	33.16	30.95	28.22	36.90	38.10	43.78	45.72	32.06
bonn 77	39.72	42.73	44.00	46.12	46.53	47.60	54.03	**36.16**	37.65	52.23	45.06
bonn 78	20.28	20.04	19.56	**19.00**	27.72	29.47	53.28	54.55	57.34	56.82	28.59
bonn 79	25.16	24.79	**24.35**	24.40	28.52	34.52	40.20	45.65	47.92	51.68	31.52
bonn 80	7.01	6.85	**6.68**	7.32	8.05	25.02	25.60	25.90	25.79	27.29	16.54
bonn 81	28.94	**25.16**	25.33	25.59	26.75	30.22	36.66	39.06	41.16	44.29	29.58
bonn 82	37.48	37.20	**37.08**	39.41	41.66	48.44	49.37	50.46	50.84	52.44	45.05
bonn 83	6.87	19.47	6.41	6.42	**6.37**	7.42	24.61	28.61	35.10	39.31	13.45
bonn 84	7.85	7.62	7.11	6.57	**6.22**	36.73	37.43	18.23	42.88	46.48	13.04
bonn 85	**16.05**	16.16	33.62	20.01	24.22	27.72	42.02	33.23	34.75	36.55	30.48
bonn 86	**17.41**	21.13	23.69	23.99	24.33	47.27	47.69	47.25	28.74	49.08	26.54
bonn 87	32.40	32.20	37.68	**31.60**	38.04	36.96	39.56	41.49	48.90	47.30	37.86
bonn 88	8.82	8.68	8.35	8.04	**8.03**	9.22	25.68	26.79	27.67	30.07	9.02
bonn 89	6.67	**6.51**	33.15	32.95	32.51	32.45	7.63	10.35	14.07	19.59	16.83
bonn 90	5.47	5.25	26.84	**4.93**	5.38	26.78	29.86	34.09	38.72	32.07	26.81
bonn 91	24.23	24.28	24.22	23.91	23.89	**23.75**	25.64	27.90	55.02	56.73	24.26
bonn 92	20.09	19.81	19.65	19.54	22.75	23.24	26.38	**8.59**	12.95	41.61	19.95
bonn 93	**8.98**	9.12	9.59	10.97	20.23	34.93	32.16	32.98	34.60	36.85	26.20
bonn 94	**8.04**	8.05	8.20	8.61	10.33	13.95	18.80	26.83	35.35	37.72	12.14
bonn 95	**15.42**	15.45	15.45	16.28	18.10	22.03	24.66	25.55	37.12	37.99	20.07
bonn 96	**16.79**	16.88	24.52	24.83	25.01	25.70	26.46	54.63	31.32	57.37	25.35
bonn 97	24.54	24.57	24.60	24.54	**5.60**	28.98	32.62	32.57	35.25	36.98	26.79
bonn 98	16.52	**16.33**	16.49	17.04	34.13	34.51	35.68	38.90	41.20	46.05	34.32
bonn 99	5.02	4.93	**4.72**	4.75	5.27	21.81	27.61	29.86	32.09	33.53	13.54
Median	20.28	20.27	20.75	23.16	28.75	29.47	33.35	35.24	38.56	46.05	29.32

Table B.2. Summary of segmentation error rates for K-Means, DT-CWT depth 3. The transform uses the Kingsbury filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	1.12	1.04	0.89	**0.78**	0.93	1.05	1.38	1.93	2.71	3.89	1.09
D5-D92	2.40	**2.39**	2.44	2.42	2.91	4.30	6.16	8.34	11.05	14.82	3.60
D8-D84	1.03	1.03	0.98	0.94	0.77	0.71	0.57	0.40	**0.39**	0.46	0.74
D12-D17	2.03	1.97	1.88	1.76	**1.73**	1.75	1.86	1.95	2.25	2.55	1.92
My 5a	18.14	18.03	17.85	15.11	16.82	15.19	**8.73**	26.98	10.96	12.63	16.00
My 5b	21.00	7.31	20.84	23.41	**7.06**	7.43	20.97	8.81	23.77	24.16	20.90
Nat 10	24.35	36.79	**13.07**	14.15	28.19	20.84	34.57	43.16	49.23	52.83	31.38
Nat 10v	**34.20**	41.70	45.97	47.22	49.85	47.26	44.71	49.01	57.93	64.91	47.24
Nat 16b	**30.72**	30.89	31.49	33.50	39.41	48.02	55.07	58.09	62.19	67.50	43.71
Nat 5b	28.93	28.81	28.65	30.49	29.67	**28.58**	29.76	37.70	37.82	39.21	29.71
Nat 5c	5.32	23.99	4.89	**4.72**	32.51	31.15	31.65	7.00	17.66	22.39	20.03
Nat 5m	**6.63**	25.76	31.93	32.06	31.02	12.45	33.08	34.92	36.18	39.35	31.99
Nat 5v2	32.67	**12.27**	33.50	34.07	25.74	28.97	32.36	44.62	47.87	45.86	33.08
Nat 5v3	**11.86**	31.97	12.79	14.45	18.69	25.05	36.46	37.84	41.11	47.08	28.51
Nat 5v	**14.17**	14.78	16.03	17.83	30.63	36.04	35.83	36.97	38.40	40.28	33.23
bonn 00	20.98	20.72	19.92	**19.75**	20.53	23.68	22.44	23.21	23.82	47.43	21.71
bonn 01	25.59	25.53	25.25	24.85	26.61	25.02	27.23	**21.50**	25.90	33.06	25.56
bonn 02	46.68	46.82	28.89	28.78	**28.30**	46.83	46.70	46.69	46.79	35.65	46.68
bonn 03	34.06	34.45	**4.78**	4.94	5.88	36.08	38.38	42.36	10.60	12.93	23.49
bonn 04	30.81	30.74	30.45	30.21	29.69	29.46	**29.41**	29.99	31.09	33.26	30.33
bonn 05	10.85	10.64	**10.42**	10.81	11.73	13.35	16.93	39.24	40.48	43.19	12.54
bonn 06	4.46	4.29	**4.03**	25.78	25.29	5.16	7.34	30.05	30.02	49.04	16.31
bonn 07	27.12	26.76	26.51	**26.20**	33.14	35.28	34.55	33.69	32.89	32.81	32.85
bonn 08	**27.78**	29.53	30.90	28.47	34.53	35.17	37.22	41.75	49.54	52.57	34.85
bonn 09	22.36	22.16	**21.99**	23.25	30.15	32.28	34.27	49.62	49.73	51.43	31.21
bonn 10	17.85	**17.77**	40.04	45.42	46.09	38.30	43.35	50.98	60.99	68.65	44.39
bonn 11	**23.54**	23.60	23.72	24.10	35.15	29.63	29.13	35.32	29.34	55.96	29.24
bonn 12	8.22	8.06	7.81	**7.74**	9.21	11.99	34.64	35.56	35.93	38.40	10.60
bonn 13	18.84	3.96	**3.93**	19.00	25.82	25.28	26.76	29.30	31.95	37.01	25.55
bonn 14	7.93	**7.64**	20.17	21.22	23.03	28.29	32.46	11.72	35.66	36.47	22.13
bonn 15	31.29	31.79	32.14	32.32	**25.54**	47.13	26.40	34.57	33.78	33.09	32.23
bonn 16	5.08	**5.06**	5.22	5.58	6.30	6.82	7.83	26.10	10.38	12.38	6.56
bonn 17	42.24	42.63	43.17	54.84	34.96	35.43	36.42	**33.87**	34.93	35.53	35.98
bonn 18	30.50	30.44	**25.18**	33.16	35.70	36.22	37.58	38.26	39.34	41.75	35.96
bonn 19	21.73	21.45	20.62	20.26	25.70	**6.20**	25.96	57.15	27.23	28.47	23.72
bonn 20	39.96	39.43	38.80	37.93	37.79	38.30	39.29	40.47	**28.03**	47.71	39.04

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 21	**16.20**	16.83	16.78	16.77	17.03	17.32	24.88	28.03	30.96	49.48	17.18
bonn 22	24.75	37.04	36.94	37.29	38.15	**16.33**	42.77	44.47	47.54	38.33	37.72
bonn 23	28.77	28.56	28.88	28.38	27.98	26.16	29.44	**25.39**	25.75	27.60	28.18
bonn 24	**29.91**	30.78	31.59	31.43	35.98	37.58	37.34	38.06	47.31	47.41	36.66
bonn 25	19.57	**19.57**	20.42	20.94	23.41	24.99	26.50	27.01	28.34	48.08	24.20
bonn 26	17.78	17.55	17.98	23.68	22.59	25.70	26.21	**8.89**	32.03	31.01	23.14
bonn 27	27.14	**26.84**	27.08	28.25	27.45	27.48	27.56	28.18	29.63	53.25	27.52
bonn 28	20.03	19.82	**19.46**	20.38	26.07	28.35	27.55	28.86	31.65	38.68	26.81
bonn 29	4.02	3.86	3.56	3.33	**3.08**	24.46	3.80	25.16	25.54	27.22	3.94
bonn 30	24.88	24.71	24.42	24.32	24.08	**23.99**	34.00	34.55	35.10	35.79	24.80
bonn 31	5.18	5.04	4.56	**4.44**	33.93	33.65	34.01	35.17	35.91	41.97	33.79
bonn 32	**32.37**	32.58	33.53	45.49	44.43	44.45	44.26	45.59	52.50	54.29	44.44
bonn 33	15.03	**14.94**	15.38	17.74	22.68	23.56	23.73	24.51	25.30	24.75	23.12
bonn 34	26.19	25.79	25.02	24.28	25.41	**17.60**	19.31	34.63	32.10	36.21	25.60
bonn 35	18.01	17.95	6.02	25.88	5.11	**4.94**	25.06	24.80	24.39	23.90	20.96
bonn 36	21.70	21.79	21.81	**13.43**	24.78	27.55	26.53	26.59	28.30	53.54	25.65
bonn 37	26.86	26.45	3.08	25.67	**2.77**	28.17	29.02	28.59	4.40	5.75	26.06
bonn 38	33.36	**33.23**	33.42	33.59	35.19	36.49	36.60	36.86	46.14	45.68	35.84
bonn 39	2.46	2.37	2.27	2.23	2.08	**2.01**	31.06	2.40	3.29	4.93	2.39
bonn 40	27.40	5.71	**5.14**	26.54	29.66	30.94	30.79	28.23	28.57	28.63	28.40
bonn 41	35.58	35.46	35.23	**35.20**	35.77	36.57	55.35	36.96	38.91	62.57	36.17
bonn 42	18.34	**18.17**	40.66	40.21	39.67	21.28	22.88	24.19	51.53	25.78	24.98
bonn 43	26.79	26.57	26.18	**6.83**	8.10	33.73	39.84	19.02	40.73	46.47	26.68
bonn 44	37.54	37.23	35.61	30.96	37.09	37.81	39.17	41.72	**28.10**	31.70	37.16
bonn 45	14.70	15.10	14.90	**14.67**	14.84	15.02	15.67	17.27	19.96	45.56	15.06
bonn 46	**25.98**	26.13	27.75	29.11	31.86	35.36	40.48	46.50	48.61	50.58	33.61
bonn 47	**6.03**	6.09	6.10	33.33	33.54	59.11	34.47	34.52	35.91	36.43	34.01
bonn 48	**6.35**	20.92	22.61	22.88	24.30	24.88	25.71	25.90	26.59	27.59	24.59
bonn 49	**25.76**	25.91	26.61	28.56	36.91	37.99	39.33	40.17	40.83	41.91	37.45
bonn 50	**13.62**	30.62	30.16	29.56	28.91	29.47	30.29	31.20	38.69	42.90	30.22
bonn 51	24.08	23.99	23.81	31.80	23.56	55.12	32.79	**4.92**	32.51	32.47	27.94
bonn 52	23.93	24.08	24.03	47.09	35.31	31.45	**7.23**	31.41	32.77	34.81	31.43
bonn 53	**29.57**	33.97	34.73	35.71	37.65	38.84	40.65	43.20	49.04	49.46	38.25
bonn 54	25.01	**9.11**	24.77	25.52	28.78	38.41	39.36	40.52	17.91	44.46	27.15
bonn 55	6.92	**6.78**	6.88	7.34	8.42	29.21	11.90	14.14	17.38	34.50	10.16
bonn 56	31.90	**22.45**	31.35	31.02	30.54	30.73	31.86	34.38	33.23	34.33	31.61
bonn 57	5.78	5.69	**5.58**	5.90	23.75	24.33	24.74	25.51	15.75	28.72	19.75
bonn 58	6.10	6.01	5.77	5.66	5.43	**5.40**	5.50	6.05	7.12	10.39	5.89
bonn 59	10.10	9.91	9.78	**9.69**	9.69	10.00	10.80	12.07	14.37	19.43	10.05
bonn 60	50.99	48.97	48.29	50.17	48.28	53.02	52.59	52.93	**26.18**	52.28	50.58
bonn 61	19.82	19.46	**7.01**	7.13	20.05	22.34	15.56	22.86	15.69	22.55	19.64
bonn 62	22.79	22.49	**7.60**	7.75	24.96	27.80	31.94	35.58	36.90	43.46	26.38
bonn 63	32.29	32.59	32.40	33.00	48.40	**25.60**	49.09	48.53	47.78	47.68	40.34
bonn 64	14.09	**13.97**	15.66	16.03	18.00	33.36	35.35	37.36	40.34	42.38	25.68
bonn 65	20.79	20.67	20.39	20.08	19.74	19.56	19.45	**19.37**	19.59	19.98	19.86
bonn 66	2.88	2.84	**2.76**	24.21	18.66	32.56	32.30	33.31	24.74	25.61	24.48
bonn 67	**17.46**	18.40	18.43	18.68	21.80	24.28	25.58	50.15	50.70	50.42	23.04
bonn 68	**21.02**	21.09	21.20	21.61	22.49	23.08	24.82	26.77	30.10	35.63	22.78
bonn 69	**7.42**	36.36	7.54	7.82	9.44	12.22	37.02	19.73	23.10	27.70	15.98
bonn 70	32.39	**31.20**	31.50	34.35	36.43	39.43	44.06	42.29	43.77	46.62	37.93
bonn 71	6.80	6.69	6.35	6.01	**5.95**	6.54	7.31	8.77	49.12	48.80	6.74
bonn 72	17.71	17.55	**17.21**	20.37	20.14	19.97	20.25	20.54	21.08	21.83	20.19
bonn 73	**15.89**	15.94	16.08	16.25	16.97	17.59	29.21	33.89	30.87	34.99	17.28
bonn 74	4.68	4.50	**4.36**	4.45	4.79	5.88	7.71	10.74	15.23	24.73	5.34
bonn 75	**28.22**	34.12	34.09	33.48	33.15	32.20	45.86	44.66	45.90	45.31	34.11
bonn 76	27.03	28.06	**6.92**	8.83	21.16	28.87	30.27	27.84	27.15	28.64	27.50
bonn 77	40.34	40.39	**40.22**	40.32	43.33	44.82	45.83	51.42	45.10	47.36	44.07
bonn 78	18.59	18.38	17.90	17.34	**16.75**	24.48	25.75	25.37	25.70	26.46	21.53
bonn 79	**11.60**	11.90	13.46	17.26	22.17	42.14	42.68	49.21	49.30	50.00	32.16
bonn 80	6.16	6.01	5.68	5.40	5.07	**4.83**	23.30	20.90	22.28	21.20	6.08
bonn 81	26.43	26.33	28.91	29.07	28.39	30.61	**12.41**	29.06	33.91	39.43	28.98
bonn 82	**37.83**	38.93	49.52	48.93	49.46	49.54	50.71	50.05	50.56	56.79	49.53
bonn 83	5.35	5.19	4.81	4.63	**4.59**	23.88	23.83	17.49	19.02	35.75	11.42
bonn 84	7.42	7.07	6.49	**6.15**	31.57	34.18	33.54	34.33	20.25	40.76	25.91
bonn 85	28.64	**11.87**	12.71	14.45	20.43	24.02	26.19	28.73	31.34	33.31	25.11
bonn 86	18.95	**18.44**	39.07	39.29	41.17	20.65	46.44	23.39	23.37	24.59	23.99
bonn 87	31.71	31.41	31.28	**31.10**	45.72	47.34	33.96	36.99	55.18	40.05	35.47
bonn 88	22.25	21.98	21.58	**7.68**	21.06	21.55	23.55	26.30	27.29	27.51	22.12
bonn 89	33.80	33.50	5.31	4.96	32.13	31.38	**4.29**	31.16	31.18	10.78	31.17
bonn 90	5.16	**5.06**	24.53	24.43	24.74	25.95	27.21	6.67	8.82	12.83	18.63
bonn 91	10.33	10.12	**9.88**	34.86	36.05	37.04	50.56	48.86	49.05	49.80	36.54
bonn 92	27.81	27.75	27.52	**27.12**	34.32	33.56	52.73	32.95	33.00	33.42	32.97
bonn 93	**7.61**	7.73	8.40	23.76	19.87	23.63	30.38	26.42	28.42	33.11	23.70
bonn 94	6.77	6.59	**6.30**	6.43	6.77	35.33	37.04	14.86	21.89	34.22	10.81
bonn 95	14.75	**14.75**	15.20	15.36	16.95	19.58	22.37	24.96	27.72	32.67	18.26
bonn 96	43.99	44.07	**15.69**	25.71	24.84	24.27	24.02	24.17	24.54	25.42	24.69
bonn 97	24.71	24.54	24.50	30.39	24.11	**4.71**	27.58	32.90	32.51	32.70	26.14
bonn 98	7.68	29.22	**7.52**	7.61	9.75	33.04	32.87	33.87	35.47	38.23	31.05
bonn 99	4.85	4.66	4.23	3.83	**3.67**	20.99	22.79	24.43	22.75	26.59	12.92
Median	20.98	21.09	20.17	23.41	24.96	27.55	30.27	31.16	31.18	35.75	25.91

Table B.3. Summary of segmentation error rates for K-Means, DWT depth 2. The transform uses the Daubechies 9-7 filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	2.93	2.58	**2.23**	2.56	3.61	5.55	8.67	13.54	19.93	26.50	4.58
D5-D92	2.33	**2.09**	2.11	2.55	3.41	5.60	8.46	12.11	16.85	21.84	4.50
D8-D84	1.29	1.26	1.09	1.03	0.87	0.73	**0.72**	1.06	1.93	4.11	1.07
D12-D17	2.83	2.84	2.84	2.87	**2.80**	3.05	3.25	3.69	4.44	6.10	2.96
My 5a	18.17	18.09	22.99	24.74	**9.85**	11.13	26.14	28.27	23.87	26.31	23.43
My 5b	20.33	20.15	**5.76**	6.14	7.65	9.95	23.07	29.34	21.91	28.85	20.24
Nat 10	**29.76**	31.18	30.81	34.29	38.14	51.75	56.79	61.74	65.57	71.99	44.94
Nat 10v	39.56	39.06	44.26	**38.44**	42.93	52.77	56.36	66.16	71.44	74.96	48.52
Nat 16b	**41.00**	43.73	42.60	45.65	56.45	60.78	67.81	72.19	77.32	80.57	58.62
Nat 5b	37.95	37.81	37.73	**16.35**	41.95	28.86	36.25	51.46	57.85	65.86	37.88
Nat 5c	29.26	**13.89**	29.74	31.19	36.08	35.03	22.28	31.37	47.08	49.51	31.28
Nat 5m	10.64	**10.52**	33.17	33.47	35.11	36.97	40.03	43.50	48.69	54.79	36.04
Nat 5v2	28.66	**28.61**	29.49	31.56	34.76	38.45	45.00	55.09	58.99	61.63	36.60
Nat 5v3	**15.97**	16.22	18.24	20.94	27.51	51.83	45.58	60.07	53.30	57.87	36.54
Nat 5v	**17.96**	18.13	19.21	20.44	23.75	27.39	38.72	40.94	46.83	52.09	25.57
bonn 00	20.45	20.06	19.67	**19.14**	21.57	25.08	25.18	25.56	27.79	32.72	23.32
bonn 01	26.18	26.15	25.47	25.49	26.40	28.93	**25.10**	31.12	42.64	48.64	26.29
bonn 02	37.31	36.84	36.89	36.46	36.50	**36.35**	37.40	41.16	43.71	44.78	37.10
bonn 03	**6.95**	6.99	7.18	7.96	31.28	11.69	13.89	16.90	21.41	26.64	12.79
bonn 04	30.88	30.71	30.51	30.13	**29.82**	30.24	31.35	33.73	37.77	44.24	30.80
bonn 05	13.12	12.73	12.03	**11.81**	12.71	14.72	48.44	50.62	54.28	60.64	13.92
bonn 06	22.06	4.67	**4.52**	4.62	5.44	7.37	10.63	36.28	41.52	45.09	9.00
bonn 07	26.93	26.73	6.53	25.65	33.67	34.56	**5.87**	34.67	35.11	35.88	30.30
bonn 08	**30.82**	31.05	31.80	33.50	37.70	53.61	58.08	60.77	65.12	67.48	45.65
bonn 09	23.41	**22.95**	26.09	27.87	31.32	33.55	36.10	50.88	51.86	53.25	32.44
bonn 10	20.11	**20.07**	20.20	20.39	29.05	30.29	32.36	35.41	42.27	50.81	29.67
bonn 11	26.93	26.80	51.33	51.87	33.05	33.36	33.79	57.31	**11.10**	38.37	33.58
bonn 12	8.47	8.22	**7.98**	8.24	10.58	14.34	18.84	25.59	43.04	48.16	12.46
bonn 13	23.39	23.33	**23.28**	23.96	33.40	35.64	38.71	41.19	44.90	50.66	34.52
bonn 14	**8.87**	34.33	34.14	34.21	30.76	10.33	14.27	36.79	38.50	36.09	34.18
bonn 15	7.09	6.88	**6.60**	25.07	26.37	7.38	8.65	10.97	14.36	42.19	9.81
bonn 16	25.29	25.02	24.64	24.16	**15.50**	16.67	18.13	19.79	22.60	28.63	23.38
bonn 17	53.12	53.20	53.86	25.74	**16.13**	37.77	38.37	38.98	41.93	48.60	40.46
bonn 18	**20.81**	20.97	21.56	23.00	25.49	48.07	47.59	38.72	42.62	45.49	32.11
bonn 19	27.28	26.92	29.12	30.66	34.66	59.72	**18.62**	22.19	43.87	62.76	29.89
bonn 20	29.70	29.41	**28.71**	30.44	36.34	36.11	54.57	29.05	53.58	40.42	33.27
bonn 21	11.52	**11.32**	35.63	12.20	13.28	12.87	22.99	41.19	43.08	47.98	18.13
bonn 22	19.31	19.12	**18.52**	38.96	41.59	44.16	47.27	51.09	53.39	55.92	42.87
bonn 23	29.47	29.24	28.89	28.57	28.55	28.78	**10.37**	23.16	14.89	32.59	28.68
bonn 24	19.67	19.20	18.85	**18.57**	19.32	25.28	30.15	33.50	56.78	60.16	22.48
bonn 25	**22.13**	22.68	23.44	24.02	25.63	25.51	28.86	32.11	41.33	54.58	25.57
bonn 26	18.80	18.66	26.61	18.44	24.96	28.77	30.82	**13.81**	35.84	38.18	25.78
bonn 27	10.69	10.49	**10.27**	10.34	11.25	34.27	33.74	24.73	44.28	52.20	17.99
bonn 28	26.97	26.48	26.08	**17.62**	34.43	37.32	40.36	43.02	46.57	59.69	35.87
bonn 29	7.72	7.42	6.92	**6.73**	6.75	7.15	7.64	9.33	27.98	30.02	7.53
bonn 30	26.92	24.52	26.33	25.09	**24.38**	34.95	37.98	39.28	39.43	40.51	30.94
bonn 31	28.06	28.08	18.63	33.96	**4.89**	28.08	38.74	43.14	48.41	23.83	28.08
bonn 32	29.63	21.37	**13.96**	29.20	15.02	17.39	20.28	26.18	44.63	51.11	23.77
bonn 33	**5.65**	13.90	18.79	18.32	19.73	22.51	25.11	26.27	27.89	29.88	21.12
bonn 34	23.72	23.43	22.22	**21.36**	29.25	30.58	33.51	39.72	43.46	50.19	29.92
bonn 35	5.90	**5.66**	17.91	18.94	22.35	5.76	7.29	10.64	34.11	38.21	14.28
bonn 36	**14.92**	27.59	29.24	15.14	16.71	18.65	21.69	41.22	45.75	52.91	24.64
bonn 37	4.93	4.80	4.69	**4.64**	4.71	5.07	6.18	9.19	34.03	20.12	5.00
bonn 38	23.63	**9.41**	24.01	24.18	25.87	26.46	26.31	26.49	29.48	30.70	26.09
bonn 39	3.15	2.96	2.75	2.50	**2.39**	2.59	3.66	31.18	9.54	15.67	3.05
bonn 40	4.20	**4.05**	25.65	33.84	33.57	34.53	35.20	35.14	30.82	32.35	32.96
bonn 41	**14.00**	14.24	14.95	15.22	16.77	38.24	39.12	40.58	42.88	47.47	27.50
bonn 42	20.65	20.53	20.01	**19.70**	27.83	27.81	55.10	29.25	37.29	53.88	27.82
bonn 43	32.34	32.23	32.33	**15.89**	48.05	22.42	31.54	50.09	54.25	57.20	32.33
bonn 44	10.89	10.64	23.54	**10.60**	14.45	19.69	33.31	34.46	37.34	40.72	21.61
bonn 45	15.05	15.01	15.06	**14.77**	27.80	17.06	19.64	23.30	32.21	52.45	18.35
bonn 46	**35.21**	35.28	36.39	36.83	38.98	51.56	53.83	51.39	51.99	51.40	45.18
bonn 47	8.33	**8.07**	25.35	33.31	33.64	33.80	34.49	33.67	36.05	39.50	33.65
bonn 48	25.14	25.08	25.61	**7.21**	26.38	25.88	26.03	27.59	30.62	35.33	25.96
bonn 49	12.12	**12.00**	32.60	33.63	41.58	22.90	46.56	46.20	48.12	50.07	37.61
bonn 50	29.86	**14.54**	29.08	28.59	28.53	29.29	29.80	24.71	28.34	41.78	28.83
bonn 51	**21.17**	21.22	21.17	21.80	22.28	33.74	33.61	33.69	34.99	37.64	27.94
bonn 52	5.10	28.60	29.39	**4.77**	4.96	5.40	7.37	35.55	37.73	45.34	17.98
bonn 53	28.92	28.72	**25.30**	34.49	34.98	35.35	36.18	38.95	42.22	52.98	35.16
bonn 54	9.27	9.06	**8.90**	9.21	10.94	15.72	19.06	23.03	29.64	54.17	13.33
bonn 55	10.58	**10.41**	10.42	11.02	12.04	13.57	41.32	24.37	26.15	35.47	12.81
bonn 56	40.92	40.76	43.82	**37.70**	38.46	41.02	43.01	44.85	49.03	50.01	42.01
bonn 57	**8.40**	8.53	9.77	15.01	17.39	23.36	28.45	34.25	48.96	49.73	20.38
bonn 58	22.38	22.20	21.94	22.48	**18.76**	19.06	19.15	22.88	46.28	47.56	22.29
bonn 59	13.13	12.95	12.53	**12.38**	13.03	14.39	16.40	19.43	32.10	35.18	13.76
bonn 60	21.59	**6.13**	21.00	6.41	27.26	55.18	13.74	32.12	35.53	40.11	24.42
bonn 61	50.27	21.80	**17.76**	21.88	20.67	23.47	22.30	24.80	25.80	28.55	22.89
bonn 62	23.54	**9.55**	26.58	27.23	29.18	29.19	37.41	48.28	51.85	60.93	29.19
bonn 63	9.52	**9.47**	31.09	9.52	36.08	10.57	12.79	28.82	32.64	40.96	20.80
bonn 64	23.49	**23.32**	23.39	32.64	32.60	34.00	37.07	38.84	43.04	49.24	33.32
bonn 65	26.07	11.04	33.53	10.28	**9.92**	31.97	32.93	19.12	33.51	40.32	29.02

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 66	3.71	3.50	**3.34**	24.11	32.97	3.70	5.08	8.45	14.20	20.74	6.77
bonn 67	21.28	**21.17**	21.44	21.61	22.53	22.58	27.21	27.83	29.27	32.93	22.55
bonn 68	30.42	30.57	31.33	32.28	30.62	**28.94**	49.98	47.87	48.60	48.33	31.81
bonn 69	8.61	**8.56**	8.98	29.28	14.52	18.63	23.39	31.32	37.04	48.19	21.01
bonn 70	26.59	26.50	26.06	25.24	**25.00**	25.34	27.09	39.17	45.16	46.06	26.54
bonn 71	18.80	18.53	8.37	**8.33**	8.83	10.06	13.41	23.09	28.13	30.61	15.97
bonn 72	18.79	**5.86**	26.25	46.30	6.26	7.28	23.13	25.19	32.12	37.83	24.16
bonn 73	28.53	**28.41**	28.53	29.19	30.03	33.91	39.45	42.57	50.70	53.06	31.97
bonn 74	5.29	5.00	**4.89**	5.13	6.90	10.13	15.95	24.73	42.50	49.17	8.51
bonn 75	8.59	8.48	8.01	**7.78**	37.23	8.98	11.00	26.45	39.14	45.25	9.99
bonn 76	33.90	**9.19**	9.44	10.27	34.24	36.07	38.92	41.73	45.34	50.11	35.15
bonn 77	28.31	**27.76**	45.66	46.89	46.73	46.76	35.63	36.32	50.81	39.78	42.72
bonn 78	19.38	19.27	19.22	**18.69**	19.59	27.93	29.83	33.93	59.74	61.61	23.76
bonn 79	**15.48**	23.41	23.18	23.92	25.95	30.01	35.61	38.81	42.39	46.60	27.98
bonn 80	8.95	8.66	**8.18**	8.56	10.18	11.52	26.84	28.06	29.85	32.90	10.85
bonn 81	26.47	**23.77**	26.95	27.40	29.02	32.29	36.69	49.75	50.01	52.51	30.65
bonn 82	39.94	**30.64**	39.50	40.42	42.21	43.34	45.21	50.68	51.32	51.19	42.78
bonn 83	8.09	**7.98**	8.57	8.70	25.04	25.70	26.40	28.04	32.69	41.55	25.37
bonn 84	9.58	9.16	8.40	**7.79**	7.80	36.33	38.48	21.03	50.59	52.00	15.31
bonn 85	**15.83**	15.87	16.47	34.04	38.08	25.66	28.85	31.39	34.51	47.64	30.12
bonn 86	17.41	17.30	17.05	16.79	**16.46**	16.86	25.16	27.85	29.67	35.05	17.36
bonn 87	32.67	32.49	32.29	**12.50**	32.35	33.48	35.49	43.34	46.92	50.34	33.07
bonn 88	20.62	20.50	7.90	7.61	**7.38**	7.68	9.11	18.32	27.06	31.86	13.71
bonn 89	6.98	6.70	**6.37**	30.25	31.92	32.73	35.33	10.06	14.95	45.09	22.60
bonn 90	18.68	18.48	5.38	**5.07**	5.18	6.82	11.36	19.45	27.96	34.31	14.92
bonn 91	24.01	23.85	23.47	**15.40**	23.06	26.67	28.10	28.92	31.45	56.73	25.34
bonn 92	5.11	**4.98**	19.36	5.06	5.23	5.70	6.68	8.80	38.56	41.14	6.19
bonn 93	25.88	25.68	25.16	**24.88**	25.68	27.41	29.41	33.37	29.59	33.86	26.64
bonn 94	7.48	7.34	**7.01**	7.09	8.65	11.97	16.89	23.42	31.42	37.18	10.31
bonn 95	**15.64**	15.81	15.95	16.35	17.16	18.64	20.26	22.12	38.75	41.44	17.90
bonn 96	**19.67**	20.42	21.26	21.78	23.23	26.06	29.10	57.35	59.20	61.68	24.65
bonn 97	25.76	25.62	25.34	25.36	25.84	33.95	29.43	**8.28**	34.91	38.37	25.80
bonn 98	**9.81**	32.23	10.11	10.54	35.88	37.43	40.33	43.16	45.42	48.08	36.65
bonn 99	45.50	45.50	16.47	**16.26**	17.31	24.19	28.08	31.45	37.89	40.59	29.76
Median	20.33	19.27	21.56	21.36	25.63	26.67	29.10	32.11	39.14	45.34	25.80

Table B.4. Summary of segmentation error rates for K-Means, DT-CWT depth 2. The transform uses the Kingsbury filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	1.14	1.12	**0.95**	0.98	0.98	1.17	1.65	3.10	5.08	8.13	1.15
D5-D92	2.62	2.47	2.20	**2.15**	2.67	3.89	6.16	9.45	13.55	18.44	3.28
D8-D84	1.16	1.14	1.07	0.92	0.85	0.78	0.70	**0.66**	0.78	1.33	0.89
D12-D17	2.72	2.67	2.54	2.45	**2.43**	2.51	2.59	2.67	3.05	3.76	2.63
My 5a	17.51	17.63	17.59	17.46	**7.35**	8.25	28.20	28.85	12.02	24.73	17.55
My 5b	6.67	25.27	**6.26**	19.53	6.88	25.26	19.71	25.53	26.09	27.17	22.48
Nat 10	24.00	24.03	24.80	**17.83**	22.41	39.88	42.54	47.12	54.06	62.93	32.34
Nat 10v	36.14	38.65	46.26	**36.05**	36.10	44.59	49.10	61.77	65.74	73.43	45.43
Nat 16b	45.07	**37.94**	42.29	45.42	54.16	58.34	62.94	66.39	72.82	76.46	56.25
Nat 5b	28.50	28.42	34.38	34.00	33.73	35.32	**22.34**	41.06	49.13	57.03	34.19
Nat 5c	8.04	**7.97**	29.74	29.77	33.91	30.18	30.70	33.15	37.71	38.04	30.44
Nat 5m	26.09	26.04	**7.61**	33.12	33.70	33.42	32.70	37.73	41.68	47.75	33.27
Nat 5v2	**26.40**	26.77	28.08	29.91	32.98	35.94	42.68	48.28	52.51	56.88	34.46
Nat 5v3	**14.28**	14.40	15.26	17.23	23.16	28.78	37.81	41.97	59.36	61.80	25.97
Nat 5v	**24.36**	32.48	32.34	33.04	34.14	35.37	35.88	37.60	40.04	44.16	34.76
bonn 00	21.31	21.12	20.43	20.03	**19.90**	24.21	25.48	25.01	25.03	46.85	22.76
bonn 01	26.32	26.48	26.51	25.32	24.85	25.96	28.42	**23.98**	28.92	35.03	26.40
bonn 02	47.20	47.12	34.85	**34.39**	34.44	34.63	35.24	36.85	39.98	43.30	36.04
bonn 03	27.07	27.04	**5.18**	5.46	6.10	7.48	9.18	11.47	14.28	18.03	10.32
bonn 04	30.74	30.57	26.48	**26.20**	29.36	31.82	29.36	30.47	32.92	38.46	30.52
bonn 05	12.23	11.93	**11.72**	12.22	14.09	17.22	25.93	46.25	47.99	53.11	15.66
bonn 06	4.35	4.24	**4.06**	4.15	5.05	6.46	30.00	33.44	55.81	36.33	5.76
bonn 07	26.26	**7.30**	25.59	25.92	31.94	35.71	33.87	33.08	32.54	32.38	32.16
bonn 08	**29.79**	30.27	31.50	33.17	35.08	37.19	45.18	50.81	54.25	60.05	36.14
bonn 09	23.33	**23.29**	23.46	25.30	29.94	32.76	35.61	38.87	48.61	50.14	31.35
bonn 10	18.62	**18.38**	18.87	19.17	28.04	31.43	33.19	38.00	45.25	69.78	29.73
bonn 11	23.44	23.46	30.47	23.82	33.07	57.01	60.08	**5.05**	34.67	57.60	31.77
bonn 12	8.39	8.07	**7.61**	7.76	9.50	13.36	17.95	37.62	38.89	43.40	11.43
bonn 13	20.10	20.05	19.92	20.03	23.42	28.77	**12.45**	32.80	36.29	39.91	21.76
bonn 14	30.58	30.28	30.07	20.43	**7.56**	8.61	11.32	14.75	36.62	36.74	25.25
bonn 15	6.54	6.41	**6.07**	29.38	31.10	32.70	35.26	21.36	34.52	34.75	30.24
bonn 16	8.89	**8.84**	8.96	9.22	9.62	11.29	12.43	14.29	16.37	19.02	10.45
bonn 17	24.62	53.25	34.63	45.69	**6.26**	35.76	34.90	34.98	35.47	36.85	35.23
bonn 18	22.91	22.87	32.15	**17.83**	23.82	29.14	37.68	36.70	39.76	42.69	30.65
bonn 19	23.25	23.46	24.67	26.25	28.33	**13.09**	33.33	35.40	37.57	40.53	27.29
bonn 20	30.36	30.05	29.43	28.68	**28.55**	40.25	28.72	29.61	30.53	46.65	29.83
bonn 21	**15.32**	15.92	16.60	16.85	23.75	17.54	23.78	39.17	39.70	56.47	20.64
bonn 22	18.54	36.32	36.76	18.25	17.74	**14.80**	42.38	44.95	47.78	50.69	36.54
bonn 23	28.28	29.01	28.67	28.20	27.97	27.96	29.22	**11.16**	24.61	30.63	28.24
bonn 24	26.20	25.92	**24.96**	25.10	34.15	38.35	39.47	39.59	46.56	46.23	36.25
bonn 25	**20.07**	20.34	21.04	21.77	23.67	25.56	26.86	27.40	29.00	51.31	24.62

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 26	16.96	16.72	16.61	16.73	21.25	22.20	**7.65**	32.26	32.20	32.72	19.11
bonn 27	10.26	22.24	**9.61**	26.39	28.41	31.12	33.26	35.48	37.52	61.84	29.77
bonn 28	23.11	22.77	**22.46**	22.66	29.64	33.03	33.45	35.82	41.44	49.24	31.33
bonn 29	17.06	16.83	5.02	4.79	**4.71**	4.82	5.28	5.81	7.27	34.38	5.54
bonn 30	24.98	24.87	24.68	24.35	**24.13**	24.33	35.49	32.59	34.63	34.12	24.92
bonn 31	35.25	**28.59**	29.22	29.01	29.66	35.09	36.29	36.99	39.91	41.57	35.17
bonn 32	36.03	**35.61**	35.69	47.78	37.18	37.59	39.20	46.92	53.19	53.12	38.39
bonn 33	5.73	13.31	13.66	**5.59**	22.23	23.07	23.67	24.82	25.93	27.65	22.65
bonn 34	24.18	**23.85**	23.96	24.18	24.53	36.23	38.86	37.96	38.11	38.60	30.38
bonn 35	18.02	5.40	4.92	18.52	20.08	**4.24**	4.66	6.32	26.09	31.48	12.17
bonn 36	22.58	22.66	**9.59**	24.32	11.01	33.19	32.39	37.84	40.72	56.52	28.35
bonn 37	3.85	3.75	3.55	**3.34**	3.35	3.56	28.31	29.23	7.45	10.41	3.80
bonn 38	23.70	23.72	24.05	**8.58**	9.14	26.92	27.49	29.83	48.17	48.18	25.49
bonn 39	2.34	2.24	2.08	1.93	**1.83**	33.59	30.51	2.26	3.52	6.15	2.30
bonn 40	27.66	25.20	4.41	**4.08**	33.50	28.61	33.86	29.29	29.60	29.68	28.95
bonn 41	35.46	35.42	**35.14**	35.24	35.93	36.93	37.84	39.86	40.90	43.93	36.43
bonn 42	19.05	**18.99**	19.59	40.09	39.75	45.37	26.34	26.65	27.31	29.75	26.98
bonn 43	28.51	9.69	**9.67**	9.87	11.21	14.84	19.44	25.62	45.94	48.54	17.14
bonn 44	36.32	36.32	8.49	**8.17**	9.18	11.49	15.26	21.65	34.67	35.12	18.46
bonn 45	16.65	16.39	15.28	14.90	**14.65**	15.64	17.25	19.31	21.49	48.46	16.52
bonn 46	32.04	31.92	**28.78**	29.42	35.86	33.11	54.83	55.32	53.92	55.54	34.48
bonn 47	5.84	**5.78**	23.90	32.91	33.64	34.39	7.53	34.10	34.59	36.35	33.27
bonn 48	22.25	19.87	**19.28**	23.99	26.86	24.19	24.29	25.09	26.97	28.37	24.24
bonn 49	26.35	9.09	**8.86**	28.59	31.31	39.98	43.63	44.97	44.51	45.51	35.65
bonn 50	29.49	29.20	28.66	27.84	**27.42**	27.51	28.41	29.97	39.90	53.54	28.93
bonn 51	22.96	22.88	22.90	22.77	**22.58**	22.97	32.79	32.54	23.19	33.21	22.96
bonn 52	29.24	29.32	29.17	28.83	31.53	**5.07**	35.13	35.85	39.40	40.23	30.43
bonn 53	29.11	28.96	**28.60**	35.37	36.75	39.54	43.05	53.74	49.73	50.01	38.15
bonn 54	9.17	8.94	8.37	**7.61**	37.52	39.53	40.83	19.62	23.89	25.15	21.75
bonn 55	6.98	6.88	6.52	**6.27**	6.30	6.99	8.70	12.02	23.64	28.41	6.99
bonn 56	36.15	35.95	35.72	**28.86**	41.42	42.66	45.92	42.38	44.40	48.02	41.90
bonn 57	6.20	6.20	**6.13**	23.37	8.12	12.58	18.44	29.64	30.83	38.21	15.51
bonn 58	8.31	8.11	7.78	**7.65**	7.81	8.53	9.48	11.25	18.82	19.83	8.42
bonn 59	11.22	11.11	10.72	**10.58**	11.04	11.55	12.73	14.57	16.97	21.99	11.39
bonn 60	21.26	21.04	**20.65**	48.88	52.95	55.24	55.17	29.03	32.35	58.33	40.62
bonn 61	17.33	5.80	17.66	21.92	**5.02**	23.46	24.49	24.54	25.04	26.26	22.69
bonn 62	24.11	7.63	7.29	**7.10**	8.28	13.55	30.71	35.14	42.62	59.80	18.83
bonn 63	31.45	31.68	32.04	32.13	**24.18**	25.12	49.30	25.45	26.43	29.85	30.65
bonn 64	11.73	11.57	**11.46**	11.57	12.70	35.53	37.16	38.68	41.72	46.19	24.11
bonn 65	**5.63**	29.07	28.15	17.80	17.61	18.44	18.60	23.61	24.26	25.17	21.10
bonn 66	2.74	2.67	**2.54**	34.22	34.30	34.39	23.86	24.35	24.79	8.22	24.11
bonn 67	**19.73**	20.03	19.74	20.05	21.56	24.60	38.03	39.98	29.83	51.91	23.08
bonn 68	12.43	**12.34**	24.33	25.41	30.04	24.45	26.23	34.27	47.10	39.88	25.82
bonn 69	6.51	**6.38**	6.67	24.70	26.43	13.34	16.66	21.61	27.06	35.48	19.13
bonn 70	**13.71**	26.66	32.95	26.10	18.98	35.41	40.52	43.80	45.80	48.00	34.18
bonn 71	8.49	8.45	**8.40**	8.45	8.74	9.48	11.88	19.76	24.59	25.69	9.11
bonn 72	38.25	38.15	37.87	16.94	16.55	45.48	17.32	18.79	**10.94**	34.42	26.61
bonn 73	**15.59**	15.66	15.92	25.42	25.49	27.28	31.60	36.78	41.64	39.83	26.39
bonn 74	4.91	4.79	4.47	**4.46**	4.79	6.51	9.56	15.25	36.17	39.76	5.71
bonn 75	4.28	4.12	**3.93**	3.98	25.15	33.95	33.45	32.95	33.18	35.93	29.05
bonn 76	29.37	**7.28**	7.95	9.24	13.89	17.33	21.08	37.88	25.59	44.09	19.20
bonn 77	40.17	**39.94**	40.44	40.25	46.47	54.19	45.61	48.91	54.05	49.05	46.04
bonn 78	18.73	18.51	18.03	17.53	**16.86**	17.14	25.08	26.51	27.87	29.89	18.62
bonn 79	**12.99**	13.45	23.47	23.79	26.70	31.04	43.01	51.72	50.21	52.08	28.87
bonn 80	22.34	7.29	6.80	6.32	**5.87**	6.28	7.89	22.30	22.16	23.91	7.59
bonn 81	30.01	29.86	29.76	**8.39**	28.86	27.97	31.02	36.33	37.88	43.18	29.94
bonn 82	47.94	48.09	48.24	48.54	48.72	48.98	50.60	**40.78**	49.96	51.06	48.63
bonn 83	20.03	6.41	6.01	**5.85**	6.07	25.90	24.52	25.99	36.56	38.75	22.28
bonn 84	8.47	8.12	7.54	**7.04**	32.14	31.90	34.04	35.59	39.36	48.24	32.02
bonn 85	**11.95**	11.96	12.70	35.71	39.59	40.40	41.82	43.30	31.83	34.69	35.20
bonn 86	19.47	19.24	**18.78**	19.40	20.96	24.01	22.02	22.30	23.91	26.30	21.49
bonn 87	31.78	31.51	31.18	**31.04**	31.20	31.97	34.95	36.47	38.46	55.75	31.88
bonn 88	21.73	21.37	8.29	20.00	**7.68**	7.76	24.01	25.35	26.54	28.61	21.55
bonn 89	5.81	32.82	32.43	32.06	31.66	31.40	**5.15**	31.23	7.98	32.81	31.53
bonn 90	5.11	4.98	4.77	4.57	**4.46**	26.92	6.01	8.56	14.09	23.27	5.56
bonn 91	18.00	17.47	**16.95**	39.58	39.70	22.23	25.11	26.18	50.13	29.64	25.65
bonn 92	4.69	27.32	4.34	**4.16**	21.85	33.55	33.22	33.06	30.69	10.32	24.58
bonn 93	25.52	25.57	23.53	28.97	31.27	**23.52**	29.87	35.44	32.89	36.43	29.42
bonn 94	6.26	6.14	5.90	**5.69**	6.45	8.28	11.79	16.41	24.77	34.77	7.36
bonn 95	**13.01**	13.14	20.84	30.73	15.91	16.86	18.78	22.60	40.21	40.59	19.81
bonn 96	17.48	17.42	**17.29**	17.67	22.58	25.36	24.96	26.04	54.02	55.19	23.77
bonn 97	24.92	24.90	24.55	24.35	24.15	25.01	32.36	**5.02**	31.98	32.39	24.91
bonn 98	7.82	7.82	**7.71**	32.21	34.84	34.70	34.81	36.98	41.52	43.87	34.76
bonn 99	16.13	15.94	4.79	4.44	**4.08**	23.55	23.73	24.32	26.69	28.84	19.84
Median	20.03	20.03	18.87	21.92	23.82	26.92	29.36	32.26	34.67	38.75	25.82

B.2 Fuzzy K-Means

Table B.5. Summary of segmentation error rates for fuzzy K-Means, DWT depth 3. The transform used is the DWT with Daubechies 9-7 filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	4.21	4.11	**4.08**	4.81	7.67	15.01	29.35	39.66	49.37	29.05	11.34
D5-D92	**2.54**	2.71	3.08	3.91	6.48	9.53	12.46	15.88	18.69	21.02	8.01
D8-D84	0.95	1.01	0.99	0.87	**0.81**	0.87	0.99	1.03	1.29	1.66	0.99
D12-D17	1.87	1.83	1.72	**1.62**	1.81	2.17	2.65	3.36	4.11	4.91	2.02
My 5a	11.78	**11.71**	21.17	24.26	19.84	21.37	22.77	23.44	26.92	29.78	22.07
My 5b	9.12	**9.09**	9.29	9.68	10.83	12.08	14.48	19.90	30.93	57.28	11.45
Nat 10	**51.77**	52.85	60.84	62.78	69.20	71.27	73.89	80.43	80.74	81.00	70.23
Nat 10v	**55.08**	55.21	56.33	59.38	67.14	67.83	68.89	70.42	72.20	74.65	67.49
Nat 16b	**57.15**	58.87	63.51	66.28	72.56	77.66	80.27	82.06	82.79	83.51	75.11
Nat 5b	**10.47**	10.63	11.57	13.55	19.70	53.45	56.71	59.20	63.12	64.00	36.58
Nat 5c	**15.90**	15.95	16.46	18.18	22.88	32.09	50.84	45.18	50.02	51.14	27.48
Nat 5m	**10.82**	11.01	11.63	13.23	17.19	22.99	34.55	46.86	54.43	61.06	20.09
Nat 5v2	**40.08**	40.14	41.19	42.41	44.45	46.44	49.51	54.08	59.84	64.43	45.44
Nat 5v3	**22.66**	23.66	33.44	45.47	59.84	55.16	59.99	63.13	65.08	65.84	57.50
Nat 5v	**13.59**	14.11	15.30	17.35	22.52	37.28	44.59	47.53	58.12	52.00	29.90
bonn 00	26.73	27.05	27.28	27.34	27.07	**19.56**	27.71	28.17	28.86	30.08	27.31
bonn 01	32.44	**32.17**	32.34	32.98	34.32	45.84	49.21	59.25	59.92	60.65	40.08
bonn 02	15.43	**15.38**	15.77	16.25	18.41	31.39	33.13	36.02	37.59	39.80	24.90
bonn 03	**7.45**	7.48	7.79	8.50	11.29	13.76	27.03	28.99	31.34	32.60	12.52
bonn 04	36.17	36.15	36.05	35.91	36.02	36.18	36.12	36.05	36.13	**35.47**	36.09
bonn 05	9.09	**9.03**	9.32	10.79	13.21	23.33	24.70	23.24	23.66	25.80	18.23
bonn 06	4.97	4.83	4.79	**4.79**	5.31	6.46	33.05	35.35	41.84	57.39	5.89
bonn 07	8.69	8.44	7.93	33.31	6.78	6.45	**6.38**	35.25	35.10	14.16	8.57
bonn 08	**41.46**	41.89	44.48	54.66	59.97	60.72	60.78	59.24	64.51	65.31	59.61
bonn 09	**28.01**	30.19	30.47	30.91	32.35	34.31	36.99	39.66	40.51	42.50	33.33
bonn 10	**25.98**	26.37	27.01	28.30	37.07	48.88	53.69	54.24	55.71	56.30	42.98
bonn 11	**5.38**	34.29	34.19	34.12	34.11	5.74	7.34	28.09	37.51	50.54	34.12
bonn 12	**9.01**	9.01	9.32	10.06	12.11	15.34	21.56	25.84	30.05	37.87	13.72
bonn 13	**12.95**	13.19	13.85	15.61	31.09	33.97	38.12	40.66	42.20	44.81	32.53
bonn 14	7.21	6.98	6.55	**6.34**	6.78	21.00	27.45	31.35	33.75	34.14	14.11
bonn 15	6.16	**6.11**	22.72	23.15	22.97	8.27	9.62	10.96	16.80	21.02	13.88
bonn 16	**5.90**	5.94	6.12	6.46	7.45	8.46	10.13	11.96	14.34	46.41	7.95
bonn 17	16.54	16.04	**15.90**	16.12	17.21	19.09	28.30	28.00	29.00	32.19	18.15
bonn 18	**29.13**	29.32	29.52	29.40	43.01	44.82	46.58	47.86	49.08	50.12	43.91
bonn 19	37.43	37.38	37.24	37.12	36.24	35.05	**31.24**	32.10	32.97	34.45	35.64
bonn 20	**31.08**	31.76	32.92	33.89	34.80	35.14	35.83	35.94	36.21	36.50	34.97
bonn 21	11.24	11.13	**10.93**	11.09	12.16	13.30	14.93	17.05	19.26	22.61	12.73
bonn 22	**25.58**	25.93	26.89	30.00	30.43	27.48	42.00	44.20	46.91	50.34	30.22
bonn 23	28.33	10.26	10.06	**9.92**	10.43	30.93	31.08	18.85	21.82	25.07	20.34
bonn 24	22.95	23.44	24.82	25.56	26.34	26.63	26.38	25.95	23.64	**22.11**	25.19
bonn 25	**10.99**	11.22	12.13	25.73	17.20	24.60	24.85	25.45	28.86	43.52	24.72
bonn 26	6.39	6.38	**6.18**	6.21	24.72	23.85	23.44	26.98	30.62	33.89	23.65
bonn 27	30.52	30.69	31.34	30.68	**10.64**	28.01	26.04	29.26	32.38	35.05	30.60
bonn 28	**33.03**	33.31	33.95	34.54	36.10	38.40	48.49	50.80	50.75	50.21	37.25
bonn 29	6.79	6.53	6.25	**6.09**	6.35	7.08	8.40	26.78	17.24	23.61	6.94
bonn 30	**32.18**	32.79	34.12	35.75	38.81	38.09	38.13	37.84	38.54	38.85	37.96
bonn 31	6.13	6.09	5.89	**5.87**	6.30	7.93	30.74	35.24	37.43	38.85	7.12
bonn 32	12.35	**12.35**	12.83	13.96	19.18	33.91	36.10	38.47	40.52	47.88	26.54
bonn 33	**8.02**	21.94	22.29	22.59	22.94	22.60	22.03	21.48	20.79	18.58	21.98
bonn 34	13.06	**13.02**	13.37	17.10	20.23	36.53	37.26	38.26	39.18	43.98	28.38
bonn 35	7.13	6.92	**6.54**	6.59	7.06	7.84	21.51	19.48	20.69	22.29	7.49
bonn 36	26.48	**12.46**	13.62	15.09	18.55	29.86	32.45	37.79	41.19	43.44	28.17
bonn 37	4.35	4.31	4.14	**4.00**	4.10	4.27	5.04	8.90	42.94	46.37	4.33
bonn 38	24.59	**9.78**	10.21	10.69	12.71	14.84	15.92	17.05	18.69	34.92	15.38
bonn 39	3.22	3.12	2.99	**2.96**	3.20	3.95	5.88	9.92	19.10	28.88	3.59
bonn 40	5.46	5.39	5.19	5.09	**4.95**	5.08	5.38	5.90	30.14	7.26	5.38
bonn 41	**19.60**	19.81	20.68	22.53	25.66	28.85	36.90	38.23	43.55	50.84	27.26
bonn 42	22.31	25.71	24.37	25.12	22.97	27.83	**22.07**	29.85	27.15	25.46	25.29
bonn 43	**20.75**	20.77	21.15	22.28	26.45	33.15	35.06	51.38	55.73	54.96	29.80
bonn 44	**11.71**	13.60	25.18	23.30	23.13	19.79	22.42	25.71	30.30	34.14	23.21
bonn 45	16.31	**16.19**	16.47	18.01	20.92	24.47	29.56	33.74	47.21	48.32	22.69
bonn 46	**26.90**	27.35	28.16	31.05	32.15	33.05	35.05	36.02	38.94	41.33	32.60
bonn 47	11.41	**11.33**	11.51	11.91	13.23	15.25	33.15	36.94	35.28	38.45	14.24
bonn 48	**7.22**	25.34	25.17	25.02	24.66	24.02	23.30	21.16	27.61	34.42	24.84
bonn 49	36.60	36.56	36.29	35.74	**19.62**	35.46	38.22	47.43	48.38	49.88	36.58
bonn 50	**13.24**	13.34	13.70	14.68	16.45	32.64	36.24	42.57	44.57	46.67	24.54
bonn 51	4.38	**4.30**	32.05	4.53	31.86	31.15	8.18	10.14	13.01	21.06	11.57
bonn 52	27.16	27.33	34.44	**5.91**	30.58	32.32	34.88	37.88	40.23	42.07	33.38
bonn 53	35.18	35.11	34.84	**34.53**	35.35	35.97	35.99	35.22	36.24	41.29	35.28
bonn 54	**10.49**	10.50	10.68	10.77	11.22	13.09	17.50	18.86	20.50	22.52	12.16
bonn 55	8.16	8.03	**7.97**	8.17	8.57	10.50	24.74	27.92	32.79	56.91	9.53
bonn 56	**26.06**	26.74	30.19	30.70	48.83	49.49	50.30	50.84	51.79	52.42	49.16
bonn 57	**6.85**	6.98	7.31	8.12	9.50	10.86	49.74	56.00	56.49	50.73	10.18
bonn 58	**8.36**	8.44	8.95	9.81	11.99	13.24	15.23	18.71	26.92	42.60	12.62
bonn 59	10.86	**10.82**	11.02	11.45	13.44	17.69	21.92	27.03	30.82	43.67	15.56
bonn 60	23.38	**22.20**	23.90	23.13	24.58	23.69	23.81	25.35	28.39	35.57	23.85

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 61	**9.57**	9.64	10.20	23.03	21.87	15.59	21.39	22.10	22.40	22.27	21.63
bonn 62	23.11	23.12	22.82	**22.64**	24.35	29.33	43.55	47.19	49.89	52.20	26.84
bonn 63	**7.66**	7.73	8.54	10.71	15.91	20.50	25.83	29.57	33.44	37.21	18.20
bonn 64	**14.62**	14.74	15.59	16.65	19.60	23.03	27.10	30.93	36.78	38.11	21.32
bonn 65	23.99	4.05	**4.03**	4.15	23.82	26.16	26.39	21.69	24.91	27.13	23.90
bonn 66	3.34	**3.32**	3.38	3.50	3.87	4.66	6.09	8.95	15.21	30.00	4.27
bonn 67	18.41	**18.35**	18.70	19.01	24.88	25.25	25.51	25.66	26.29	26.84	25.07
bonn 68	10.13	**10.02**	25.25	25.84	27.46	30.56	32.73	34.45	35.42	36.08	29.01
bonn 69	**9.80**	9.92	10.93	13.57	31.71	48.57	52.65	53.86	58.50	59.27	40.14
bonn 70	**11.29**	11.66	12.06	12.95	14.63	16.99	21.27	26.15	32.89	48.68	15.81
bonn 71	9.48	9.36	9.20	**9.14**	9.50	10.48	12.94	14.65	25.16	25.93	9.99
bonn 72	4.82	4.74	**4.71**	4.92	5.29	6.44	25.95	26.87	27.80	39.70	5.86
bonn 73	50.01	**49.95**	50.17	50.76	51.47	52.40	52.34	51.54	50.42	51.24	51.00
bonn 74	7.14	**7.09**	7.18	7.86	30.11	38.13	39.62	41.72	43.74	45.82	34.12
bonn 75	7.21	**7.12**	7.12	7.39	8.15	9.81	17.27	22.01	34.36	42.96	8.98
bonn 76	**11.54**	11.57	12.32	13.93	17.82	21.26	24.53	28.43	32.23	37.18	19.54
bonn 77	34.61	**34.51**	34.74	34.82	35.00	34.89	34.72	36.34	35.85	36.78	34.85
bonn 78	23.58	23.50	**7.71**	7.86	9.71	21.35	23.85	51.49	28.27	29.51	23.54
bonn 79	**13.67**	13.84	14.72	17.30	22.17	26.62	30.62	33.96	38.31	45.32	24.40
bonn 80	5.55	5.52	**5.38**	5.88	7.79	11.28	23.44	21.58	20.78	19.68	9.53
bonn 81	28.88	28.94	**26.42**	26.77	26.49	27.31	33.78	41.60	47.20	53.32	28.91
bonn 82	**39.01**	39.07	39.21	39.88	39.70	39.82	39.73	40.56	41.60	41.76	39.78
bonn 83	24.49	24.65	25.96	26.25	26.10	26.31	25.93	**13.33**	17.43	22.52	25.29
bonn 84	7.26	6.92	6.46	**6.10**	7.06	10.57	13.08	15.29	17.80	21.60	8.91
bonn 85	**29.19**	29.33	29.71	30.18	30.58	31.21	49.65	50.95	61.36	61.92	30.90
bonn 86	**23.99**	24.07	24.22	24.37	24.83	25.12	25.43	25.04	29.55	32.37	24.94
bonn 87	33.09	**33.03**	33.47	33.93	34.81	36.40	37.65	38.98	37.51	37.78	35.60
bonn 88	8.90	8.76	**8.40**	8.51	9.37	30.51	29.19	28.63	28.05	27.97	18.67
bonn 89	5.97	5.83	5.49	**5.38**	5.57	6.26	7.54	10.58	15.20	23.48	6.11
bonn 90	5.48	**5.40**	5.41	5.61	7.01	12.27	24.85	32.66	41.54	43.12	9.64
bonn 91	**10.95**	11.66	15.81	21.88	23.86	22.99	22.03	22.38	23.77	24.30	22.20
bonn 92	5.96	5.99	**5.87**	6.03	9.73	16.24	17.27	18.29	20.26	29.32	12.99
bonn 93	**10.22**	10.46	11.16	12.12	14.94	19.14	25.21	60.61	63.30	64.08	17.04
bonn 94	**9.28**	9.31	9.61	10.80	30.08	36.00	37.58	38.57	40.05	43.13	33.04
bonn 95	**18.21**	18.49	20.97	28.59	27.06	26.31	25.64	25.86	28.14	33.79	26.08
bonn 96	21.84	21.84	21.85	21.40	**20.56**	21.37	22.45	21.69	22.60	26.44	21.84
bonn 97	6.10	5.98	**5.91**	6.07	6.32	29.77	8.90	12.82	15.82	40.36	7.61
bonn 98	**14.97**	15.20	16.19	17.69	21.36	25.46	32.07	37.96	46.69	51.93	23.41
bonn 99	5.34	5.22	5.18	**5.16**	17.81	18.65	20.80	31.86	35.81	40.26	18.23
Median	11.78	12.35	13.85	16.25	20.56	24.60	27.10	31.35	35.10	38.85	23.65

Table B.6. Summary of segmentation error rates for fuzzy K-Means, DT-CWT depth 3. The transform uses the Kingsbury filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	1.28	1.27	1.10	1.05	**1.01**	1.10	1.27	1.68	2.01	2.72	1.27
D5-D92	**2.43**	2.49	2.75	2.98	3.96	5.81	8.12	10.40	12.67	15.20	4.88
D8-D84	0.98	0.95	0.95	0.85	0.76	0.68	0.63	0.49	0.43	**0.37**	0.72
D12-D17	1.81	1.78	1.70	1.61	1.54	**1.46**	1.51	1.71	1.97	2.12	1.70
My 5a	15.61	**15.58**	30.05	31.66	40.06	25.10	22.73	42.40	42.69	43.57	30.85
My 5b	8.19	8.04	7.57	7.10	**6.68**	51.23	39.87	25.54	38.30	50.99	16.87
Nat 10	**64.46**	64.52	64.85	65.56	79.91	80.07	80.06	80.08	80.10	80.03	79.97
Nat 10v	53.35	**53.03**	54.79	62.72	58.71	67.77	67.75	75.89	80.07	80.12	65.24
Nat 16b	**69.15**	69.95	75.75	76.24	77.01	80.98	84.27	84.88	87.47	87.55	78.99
Nat 5b	**8.07**	8.18	8.54	41.64	59.92	59.70	56.52	41.60	60.24	61.15	49.08
Nat 5c	**5.99**	6.07	6.42	6.98	28.98	46.53	40.62	33.41	40.99	45.68	31.20
Nat 5m	**7.59**	7.83	10.24	14.39	18.77	35.95	54.27	44.64	58.04	60.14	27.36
Nat 5v2	31.48	**31.28**	34.58	40.03	44.54	59.39	60.11	61.46	62.14	62.86	51.97
Nat 5v3	**30.65**	35.13	36.58	50.24	42.41	43.06	50.52	56.45	58.65	58.53	46.65
Nat 5v	**41.41**	43.73	45.24	45.14	46.01	47.00	48.40	50.25	60.02	60.70	46.50
bonn 00	25.76	25.67	18.26	17.62	**16.94**	17.00	25.42	27.09	29.39	30.92	25.55
bonn 01	**16.85**	17.14	22.83	33.02	36.77	41.85	50.79	52.67	57.42	57.93	39.31
bonn 02	**20.39**	20.95	27.42	27.46	27.43	27.53	36.54	39.69	45.57	57.17	27.50
bonn 03	4.11	4.03	**4.03**	4.36	5.12	24.84	25.99	27.23	28.06	29.08	14.98
bonn 04	34.14	34.14	34.23	34.33	34.00	33.69	33.38	33.08	**31.67**	35.44	34.07
bonn 05	8.71	8.44	8.15	**8.14**	8.28	10.76	17.41	16.53	19.29	48.44	9.74
bonn 06	4.58	4.49	4.29	**4.21**	4.55	33.36	38.25	51.31	52.91	56.29	18.97
bonn 07	8.11	7.79	7.31	6.85	31.26	30.99	**5.88**	10.27	17.06	30.70	9.19
bonn 08	56.59	51.09	**37.23**	55.58	60.50	52.15	59.67	60.10	50.49	55.37	55.48
bonn 09	28.59	28.75	**20.40**	26.78	30.21	33.45	35.52	38.36	39.50	40.10	31.83
bonn 10	**28.47**	29.80	35.82	47.05	50.52	48.68	50.22	51.61	54.24	53.39	49.45
bonn 11	32.76	32.91	32.88	32.16	31.15	31.80	**26.65**	27.97	33.53	48.55	32.46
bonn 12	8.38	8.23	**8.07**	8.42	10.17	11.44	15.28	22.08	34.68	38.70	10.81
bonn 13	**32.05**	32.57	34.34	35.62	36.27	36.85	37.09	37.67	39.03	41.50	36.56
bonn 14	7.01	6.79	6.15	**5.64**	20.52	21.37	22.44	23.53	25.66	30.50	20.95
bonn 15	20.13	20.75	20.72	20.54	19.56	18.60	**5.49**	11.41	25.54	37.05	20.34
bonn 16	**4.52**	4.56	4.64	4.92	5.76	6.59	7.85	10.18	30.92	46.37	6.17
bonn 17	9.17	**9.03**	9.47	12.64	16.55	26.20	24.93	24.24	26.65	28.16	20.39
bonn 18	**24.97**	25.23	30.08	40.57	42.02	43.21	44.35	45.51	46.33	56.37	42.61
bonn 19	22.97	23.10	23.71	25.65	29.26	**8.12**	25.84	25.42	25.03	25.32	25.18
bonn 20	28.49	**28.42**	28.46	28.65	28.67	28.75	29.65	30.91	33.30	37.63	28.71

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 21	12.03	11.88	**11.80**	12.08	13.07	13.93	15.36	17.44	22.30	40.61	13.50
bonn 22	23.84	**23.74**	24.71	35.88	36.25	39.61	40.42	42.07	44.55	53.74	37.93
bonn 23	9.61	9.42	8.89	**8.83**	9.27	11.04	29.72	15.89	18.80	22.24	10.33
bonn 24	**8.81**	23.33	23.25	23.18	23.05	22.06	22.13	21.83	23.28	27.86	23.12
bonn 25	22.19	22.10	21.86	21.48	**14.15**	20.21	20.00	23.91	41.18	44.99	21.98
bonn 26	23.33	23.00	21.48	**18.53**	19.37	20.35	23.18	28.76	33.06	35.37	23.09
bonn 27	29.77	29.09	26.65	8.06	**8.02**	19.80	21.01	22.96	25.04	27.45	24.00
bonn 28	26.90	26.94	**26.78**	26.82	27.36	27.15	34.17	35.71	39.77	41.20	27.26
bonn 29	4.40	4.30	4.19	4.00	**3.90**	4.14	7.61	20.36	23.50	34.74	4.35
bonn 30	30.19	30.18	30.18	27.62	**24.88**	29.36	28.43	33.52	35.72	34.92	30.18
bonn 31	5.26	5.22	4.97	**4.92**	5.04	17.74	30.46	34.61	37.36	40.01	11.50
bonn 32	**11.09**	11.19	12.59	21.14	29.08	30.14	44.79	45.75	47.03	48.10	29.61
bonn 33	20.56	20.54	19.32	20.41	19.70	18.28	18.93	17.83	**15.34**	15.77	19.13
bonn 34	**13.76**	14.83	20.80	26.12	30.95	33.50	35.47	37.66	38.33	39.21	32.22
bonn 35	6.15	6.01	5.73	5.42	5.06	**4.91**	16.89	15.41	15.54	15.72	6.08
bonn 36	22.49	**11.78**	12.30	13.17	15.95	21.18	24.18	28.39	32.73	37.93	21.84
bonn 37	3.53	3.43	3.35	**3.17**	40.39	54.68	49.69	50.15	54.69	36.64	38.52
bonn 38	23.10	7.71	**7.69**	7.98	9.09	9.48	22.67	32.04	35.84	40.35	16.07
bonn 39	2.54	2.48	2.39	**2.36**	2.52	3.17	3.78	4.29	16.20	51.91	2.86
bonn 40	32.94	5.01	33.62	33.46	**3.97**	32.60	30.28	27.31	25.70	24.77	28.79
bonn 41	**18.93**	19.03	19.45	20.43	23.25	26.86	40.72	48.12	49.57	51.43	25.05
bonn 42	20.12	20.06	19.96	19.71	19.04	**12.54**	19.35	17.83	30.84	33.11	19.83
bonn 43	28.49	28.31	46.33	**28.00**	46.86	29.60	31.22	36.88	54.41	54.72	34.05
bonn 44	**8.92**	9.01	10.13	13.61	15.65	16.42	17.76	20.36	24.29	52.40	16.03
bonn 45	**13.26**	13.42	14.09	16.25	21.72	22.80	26.76	46.14	46.51	46.99	22.26
bonn 46	**25.79**	25.95	26.51	27.91	32.63	30.33	30.44	32.71	35.84	37.76	30.39
bonn 47	**7.52**	7.60	7.78	8.39	11.32	16.77	26.40	33.62	37.00	43.63	14.04
bonn 48	**5.91**	24.11	24.02	23.75	23.09	21.94	20.37	17.14	21.88	29.10	22.51
bonn 49	**7.59**	31.37	8.24	9.45	29.13	31.50	45.52	45.66	45.92	46.62	31.43
bonn 50	**9.83**	9.88	11.19	29.87	30.98	32.57	36.06	41.50	50.45	51.12	31.77
bonn 51	3.41	**3.32**	30.90	30.40	3.64	27.42	25.01	22.68	23.12	32.51	24.06
bonn 52	28.98	**28.88**	28.94	29.35	29.13	29.55	33.80	34.26	35.57	36.15	29.45
bonn 53	34.61	34.55	34.35	34.13	33.29	**32.75**	32.78	34.91	37.12	37.56	34.45
bonn 54	**8.48**	8.56	8.97	10.31	15.47	15.64	15.98	16.24	16.89	52.72	15.55
bonn 55	7.03	**6.83**	7.06	7.41	14.19	43.17	53.14	56.77	56.35	55.21	28.68
bonn 56	25.10	25.12	24.90	**24.47**	40.30	41.15	42.21	43.33	44.21	44.94	40.73
bonn 57	**6.54**	6.59	6.66	6.69	33.82	33.08	43.77	46.05	42.48	42.62	33.45
bonn 58	6.87	6.76	6.38	**6.12**	6.17	7.83	10.28	30.87	37.26	38.17	7.35
bonn 59	10.27	10.08	**9.83**	9.86	10.57	26.65	27.62	40.68	41.36	42.05	18.61
bonn 60	21.36	21.19	20.79	19.82	18.37	17.33	**16.80**	17.06	34.23	40.82	20.31
bonn 61	**9.19**	9.26	22.57	22.02	20.34	19.12	18.35	18.07	17.65	19.12	18.74
bonn 62	**19.07**	19.38	21.30	23.14	35.28	35.77	36.82	38.82	41.89	45.32	35.53
bonn 63	23.95	23.91	**10.22**	12.04	15.22	24.35	24.71	29.91	58.06	58.11	24.15
bonn 64	**14.64**	14.83	15.91	17.09	21.45	29.89	36.31	31.26	32.74	36.53	25.67
bonn 65	22.38	22.36	22.36	22.40	22.36	21.72	20.94	19.77	18.54	**17.45**	22.04
bonn 66	3.33	3.31	3.26	**3.25**	14.30	14.81	15.05	20.49	28.25	39.88	14.56
bonn 67	22.75	**15.72**	15.99	22.94	22.73	22.43	22.27	22.38	43.40	43.68	22.58
bonn 68	23.24	23.62	23.02	22.39	**21.79**	22.12	23.66	40.96	41.69	41.22	23.43
bonn 69	9.61	9.47	**9.42**	10.24	16.43	56.60	47.24	52.06	51.66	55.54	31.83
bonn 70	**11.89**	11.94	12.26	12.94	18.54	30.76	32.31	34.43	43.55	55.88	24.65
bonn 71	7.78	7.73	7.67	**7.56**	7.63	7.61	7.86	9.69	12.24	35.97	7.76
bonn 72	5.22	**5.14**	5.15	5.33	6.21	11.68	17.90	21.79	27.55	31.97	8.94
bonn 73	**35.90**	36.00	36.24	36.52	36.84	37.55	44.28	54.64	53.56	56.27	37.20
bonn 74	**4.59**	7.65	22.19	26.40	35.75	36.38	37.24	38.56	55.70	47.05	36.07
bonn 75	**4.42**	4.45	19.14	19.46	19.27	6.79	20.87	39.35	45.02	41.23	19.37
bonn 76	**7.22**	7.45	8.29	9.97	13.69	18.93	35.49	34.39	34.61	36.22	16.31
bonn 77	35.48	35.30	34.83	34.53	33.89	33.29	**32.98**	33.95	34.19	36.02	34.36
bonn 78	19.37	19.26	18.71	17.75	16.50	**15.80**	15.81	16.38	17.05	18.05	17.40
bonn 79	**11.35**	11.55	12.70	15.39	25.78	30.93	34.67	41.70	51.70	56.58	28.35
bonn 80	5.60	5.52	5.37	**5.13**	21.15	20.69	19.90	18.98	17.69	16.09	16.89
bonn 81	25.71	25.63	25.46	25.33	25.50	**24.48**	34.55	37.96	48.95	55.86	25.67
bonn 82	36.44	36.55	36.96	**34.60**	43.52	41.69	39.78	38.78	39.18	39.36	38.98
bonn 83	23.51	23.59	23.32	23.33	22.86	22.16	**9.25**	13.17	23.70	24.69	23.32
bonn 84	7.07	6.94	**6.69**	6.89	11.57	11.28	11.07	11.15	12.64	18.98	11.11
bonn 85	**15.62**	16.10	17.30	19.17	42.46	45.53	48.10	58.47	59.33	57.10	43.99
bonn 86	21.25	14.69	**14.49**	20.84	20.59	20.03	19.72	29.32	35.09	35.25	20.71
bonn 87	**12.16**	12.51	32.35	31.90	37.91	32.68	31.98	30.71	30.90	31.96	31.93
bonn 88	25.20	25.07	24.82	**7.72**	22.19	24.38	22.42	21.42	23.43	25.02	23.90
bonn 89	5.90	5.74	5.25	4.96	**4.80**	4.87	5.57	7.78	23.03	50.45	5.65
bonn 90	4.92	4.83	4.63	**4.38**	5.87	24.48	35.78	36.32	37.08	38.01	15.18
bonn 91	**11.98**	12.95	14.21	16.81	25.40	22.65	21.30	18.45	19.02	20.42	18.73
bonn 92	5.68	**5.60**	19.48	18.85	17.76	16.61	15.85	16.05	17.16	49.54	16.89
bonn 93	46.72	**43.10**	44.53	43.74	45.99	57.54	61.77	62.32	62.82	63.27	52.13
bonn 94	**7.90**	8.03	8.12	19.65	27.00	32.06	33.37	35.05	56.40	59.63	29.53
bonn 95	**20.81**	20.82	21.36	29.34	26.82	43.83	40.17	41.84	54.80	56.69	34.75
bonn 96	23.30	23.22	22.99	22.31	20.29	15.91	15.66	**15.19**	15.57	34.33	21.30
bonn 97	**4.61**	4.64	4.80	29.36	5.98	7.67	14.57	19.37	22.03	33.45	11.12
bonn 98	**9.13**	9.28	10.47	12.84	18.82	23.83	41.80	36.18	49.42	50.15	21.33
bonn 99	4.88	4.84	**4.53**	15.58	14.15	14.57	14.89	26.92	35.51	37.36	14.73
Median	13.26	13.42	18.26	19.65	21.45	24.84	27.62	32.04	35.84	40.61	23.90

Table B.7. Summary of segmentation error rates for fuzzy K-Means, DWT depth 2. The transform uses the Daubechies 9-7 filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	2.54	2.34	**2.27**	2.66	4.06	6.62	9.75	14.12	19.02	23.23	5.34
D5-D92	2.50	**2.38**	2.42	2.77	4.30	6.84	9.75	13.96	18.02	21.45	5.57
D8-D84	1.17	1.11	0.92	0.81	0.76	**0.66**	0.73	1.06	1.64	2.88	0.99
D12-D17	2.39	2.33	2.26	**2.21**	2.23	2.45	2.73	3.16	4.03	5.46	2.42
My 5a	23.93	23.82	23.60	13.62	**13.34**	14.17	20.29	28.35	29.44	35.47	23.71
My 5b	6.78	**6.68**	7.03	7.81	9.78	11.79	13.22	16.78	21.41	27.84	10.78
Nat 10	**35.84**	36.92	36.10	38.83	47.36	53.75	64.51	70.44	73.59	76.19	50.56
Nat 10v	**41.08**	41.41	42.52	44.62	47.09	56.53	60.01	66.72	68.55	70.58	51.81
Nat 16b	**43.55**	43.74	45.77	49.24	57.64	66.19	73.10	78.30	80.74	81.80	61.91
Nat 5b	**14.67**	14.89	16.00	18.59	25.36	32.09	39.07	47.64	55.27	59.91	28.72
Nat 5c	13.35	**13.28**	13.35	13.57	15.59	20.04	34.01	43.07	47.71	49.99	17.82
Nat 5m	10.75	**10.66**	11.07	12.23	16.27	22.53	28.83	43.47	51.31	52.78	19.40
Nat 5v2	**32.53**	32.56	33.70	36.87	42.24	45.26	49.62	52.27	55.45	59.91	43.75
Nat 5v3	**13.99**	14.41	16.31	20.06	28.52	40.88	50.06	54.76	60.23	65.07	34.70
Nat 5v	**14.71**	15.23	16.38	17.85	21.03	24.33	31.57	44.08	48.60	53.68	22.68
bonn 00	22.57	22.85	24.01	24.76	25.32	25.92	**21.84**	27.98	29.79	32.23	25.04
bonn 01	18.23	18.07	17.66	**17.50**	27.15	24.73	29.72	49.51	55.03	58.71	25.94
bonn 02	**14.81**	35.31	34.90	16.11	17.68	19.68	22.21	24.88	37.70	39.54	23.55
bonn 03	**7.21**	7.30	7.49	7.88	9.46	11.37	14.28	18.12	23.33	35.88	10.42
bonn 04	31.33	31.23	**31.08**	31.27	32.34	33.75	34.52	31.91	33.00	33.37	32.13
bonn 05	11.46	11.13	10.67	**10.55**	12.08	16.41	25.17	32.40	34.78	36.93	14.25
bonn 06	4.65	**4.60**	4.60	4.64	5.37	6.96	9.88	14.20	20.06	44.76	6.16
bonn 07	7.54	7.22	6.63	6.18	**5.78**	6.60	36.80	8.05	9.36	12.79	7.38
bonn 08	**21.25**	21.60	23.03	26.23	41.53	53.74	58.15	60.93	64.18	65.98	47.63
bonn 09	16.89	**16.82**	28.51	29.43	31.40	26.95	36.35	39.15	42.22	44.89	30.41
bonn 10	**27.64**	27.85	27.86	27.72	28.88	30.80	33.18	35.94	39.23	43.38	29.84
bonn 11	3.09	2.99	**2.95**	33.59	3.41	4.10	5.33	7.31	10.57	24.41	4.71
bonn 12	8.15	7.91	**7.77**	7.89	9.30	12.17	18.93	26.51	30.67	34.32	10.73
bonn 13	10.16	**10.00**	10.28	11.17	32.16	36.89	26.92	42.11	42.08	45.28	29.54
bonn 14	6.91	6.76	6.29	**5.97**	6.22	7.73	11.19	29.91	34.97	39.40	7.32
bonn 15	6.24	6.12	**5.91**	5.99	6.56	7.53	9.08	11.65	14.89	28.78	7.04
bonn 16	7.70	7.57	**7.56**	7.82	9.37	11.27	13.84	16.77	29.20	29.93	10.32
bonn 17	12.32	12.14	**11.74**	11.76	12.46	14.04	16.36	19.70	23.86	31.06	13.25
bonn 18	**21.92**	22.50	24.09	26.12	28.45	30.54	33.94	38.49	40.23	52.18	29.50
bonn 19	29.38	29.25	29.02	28.96	39.81	40.03	**18.24**	39.24	41.55	44.43	34.31
bonn 20	29.28	**29.25**	32.59	35.35	35.46	35.31	35.59	36.37	37.75	39.13	35.40
bonn 21	10.04	9.86	9.56	**9.45**	9.69	10.82	13.08	17.96	20.80	23.16	10.43
bonn 22	13.24	**13.15**	13.48	19.18	25.10	28.11	32.18	36.41	41.46	49.75	26.61
bonn 23	27.80	27.56	27.06	26.70	**8.11**	8.22	8.76	10.57	15.70	23.61	19.66
bonn 24	17.13	16.82	**16.58**	16.77	17.61	25.08	27.78	28.81	31.20	35.99	21.35
bonn 25	23.33	23.52	23.69	24.20	**21.61**	26.29	28.90	32.50	36.23	40.15	25.25
bonn 26	6.59	6.52	6.51	**6.49**	6.66	7.81	11.54	21.50	26.78	29.10	7.24
bonn 27	10.82	10.64	**10.36**	10.50	11.14	12.99	15.65	19.26	37.64	41.14	12.07
bonn 28	**32.03**	32.15	32.86	33.56	35.73	35.85	40.12	43.19	47.27	50.59	35.79
bonn 29	5.99	5.84	5.65	**5.50**	5.96	7.21	9.30	12.71	21.59	25.32	6.60
bonn 30	31.89	25.16	34.49	30.80	31.89	37.18	39.64	**17.77**	40.94	47.58	33.19
bonn 31	5.29	5.18	4.97	**4.77**	5.42	7.75	11.49	14.97	19.34	25.05	6.59
bonn 32	13.24	**13.12**	13.32	13.60	14.95	17.72	23.32	37.38	42.11	45.36	16.34
bonn 33	6.26	6.06	5.88	**5.79**	5.88	23.32	23.58	23.77	22.88	23.09	14.57
bonn 34	12.38	**12.19**	12.46	14.23	20.11	24.47	29.27	38.13	40.19	41.69	22.29
bonn 35	5.87	5.73	5.46	**5.12**	5.21	6.03	7.85	11.79	22.93	27.12	5.95
bonn 36	12.74	**12.51**	12.77	13.74	16.13	19.57	34.75	39.56	43.40	45.95	17.85
bonn 37	4.05	3.92	**3.85**	3.88	4.20	4.96	6.86	11.36	19.00	28.31	4.58
bonn 38	**9.01**	24.30	24.44	24.41	24.70	24.91	24.84	24.02	23.18	23.92	24.35
bonn 39	3.10	2.98	2.72	2.50	**2.40**	2.59	3.36	5.16	9.07	22.98	3.04
bonn 40	4.27	4.11	3.75	3.52	**3.30**	3.55	32.54	5.15	6.77	39.54	4.19
bonn 41	**14.29**	14.31	33.65	34.72	35.15	35.46	36.30	37.42	39.43	41.74	35.30
bonn 42	**21.48**	22.05	23.42	24.26	24.93	26.97	28.07	30.09	32.82	34.98	25.95
bonn 43	**13.45**	13.48	14.12	15.55	18.96	26.16	32.04	51.06	52.20	52.01	22.56
bonn 44	11.58	**11.54**	12.70	15.47	19.18	21.08	24.18	26.08	30.50	34.77	20.13
bonn 45	14.43	14.28	14.00	**13.81**	14.29	16.14	28.06	26.37	27.55	44.83	15.28
bonn 46	**18.85**	19.43	25.18	26.99	29.31	33.70	35.65	34.77	34.52	38.09	31.50
bonn 47	8.87	8.69	**8.54**	8.62	9.09	10.18	13.64	22.67	36.83	38.82	9.63
bonn 48	27.67	27.80	27.37	24.30	24.37	24.04	**23.97**	24.76	26.51	31.12	25.64
bonn 49	11.87	**11.84**	12.56	13.90	17.24	40.76	41.55	43.30	44.12	49.76	29.00
bonn 50	10.13	10.13	**10.01**	10.29	12.13	15.80	19.92	23.66	30.00	37.64	13.96
bonn 51	4.77	4.59	4.53	**4.52**	5.16	6.61	9.06	12.57	16.86	23.58	5.89
bonn 52	23.80	23.94	24.48	**4.20**	4.49	6.08	29.48	17.34	37.01	44.76	23.87
bonn 53	34.38	34.38	34.30	**34.02**	34.49	35.07	36.01	37.48	39.13	41.06	34.78
bonn 54	9.39	**9.34**	9.65	10.17	11.62	13.05	15.65	18.97	21.78	24.06	12.33
bonn 55	10.37	10.26	**10.18**	10.28	11.38	12.56	15.10	19.38	29.89	37.90	11.97
bonn 56	**33.97**	35.03	37.54	39.77	42.39	44.83	46.37	50.40	51.49	51.92	43.61
bonn 57	7.19	**7.12**	7.28	8.40	13.62	29.00	33.86	38.04	48.43	52.14	21.31
bonn 58	23.43	23.36	**19.13**	19.37	20.59	20.90	21.62	22.72	24.60	26.20	22.17
bonn 59	12.63	12.42	**12.29**	12.37	13.40	16.26	22.12	29.52	32.45	38.31	14.83
bonn 60	21.44	6.00	**5.87**	6.12	25.19	25.73	15.53	29.61	32.82	35.06	23.32
bonn 61	5.73	5.51	**5.24**	23.20	23.01	22.47	22.49	22.86	24.12	25.36	22.68
bonn 62	24.79	**9.10**	9.75	11.89	18.36	31.46	36.84	38.26	37.68	45.18	28.12
bonn 63	6.99	**6.98**	7.18	7.58	8.47	10.27	13.35	17.79	24.80	33.01	9.37
bonn 64	10.22	**10.11**	10.23	10.58	11.90	36.85	28.46	30.62	32.57	35.05	20.18
bonn 65	10.19	**9.89**	30.94	32.14	33.86	32.94	17.19	21.71	24.69	37.94	27.82

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 66	3.19	3.09	**2.94**	2.94	3.17	3.78	5.40	8.59	12.72	17.66	3.48
bonn 67	9.41	**9.20**	9.29	10.39	16.17	19.82	21.66	26.32	26.91	28.70	18.00
bonn 68	12.76	12.52	**12.10**	28.49	29.16	29.66	30.36	30.90	31.32	33.23	29.41
bonn 69	8.29	**8.22**	8.96	10.58	14.25	21.94	29.13	41.12	57.34	59.22	18.09
bonn 70	8.03	**8.02**	8.20	8.98	11.39	17.75	22.04	25.32	28.19	31.37	14.57
bonn 71	8.31	8.14	7.86	**7.82**	7.99	9.29	14.03	21.02	23.79	24.80	8.80
bonn 72	5.83	5.66	**5.52**	5.54	5.84	6.82	8.75	12.37	24.52	36.88	6.33
bonn 73	**14.84**	15.06	16.75	20.14	32.58	36.30	45.53	48.33	49.93	50.54	34.44
bonn 74	5.47	5.30	**5.15**	5.44	7.20	10.33	15.33	27.00	42.72	46.98	8.77
bonn 75	8.62	8.44	8.35	**8.10**	8.58	9.73	12.66	17.49	28.28	34.37	9.18
bonn 76	9.21	**9.12**	9.63	34.15	36.01	36.58	26.64	33.45	40.86	46.94	33.80
bonn 77	**26.71**	26.86	27.24	34.88	32.89	35.13	35.22	36.32	38.56	40.87	35.01
bonn 78	25.05	25.39	5.96	**5.74**	5.78	24.07	24.52	25.14	24.18	26.81	24.35
bonn 79	**14.54**	14.87	16.08	17.98	21.14	24.17	26.89	29.68	32.54	35.91	22.66
bonn 80	5.98	5.87	**5.55**	5.71	6.63	8.54	11.54	25.33	26.25	26.10	7.59
bonn 81	**30.23**	30.67	31.73	30.57	31.30	33.66	34.05	43.96	43.40	52.72	32.69
bonn 82	**32.54**	32.68	40.84	41.12	41.41	41.72	42.39	43.66	44.84	45.65	41.56
bonn 83	7.88	7.77	7.54	**7.40**	8.16	25.16	25.67	13.17	16.24	21.12	10.67
bonn 84	8.87	8.59	7.93	**7.57**	7.70	10.28	25.76	22.83	24.33	25.11	9.58
bonn 85	12.81	**12.47**	12.98	14.85	22.05	26.78	29.32	31.15	34.94	39.70	24.42
bonn 86	8.72	8.04	**7.22**	21.65	21.76	22.89	24.60	26.16	26.45	34.66	22.33
bonn 87	32.66	32.43	31.95	**12.82**	14.69	18.34	23.09	29.14	38.25	45.20	30.54
bonn 88	7.16	7.01	6.77	23.77	**6.46**	7.32	10.00	16.83	30.86	29.75	8.66
bonn 89	6.24	5.97	5.68	**5.45**	5.65	6.49	7.93	9.78	13.88	20.72	6.37
bonn 90	5.27	5.10	4.99	**4.89**	5.65	7.95	13.25	21.87	32.20	39.79	6.80
bonn 91	**10.60**	10.74	12.17	24.77	23.99	24.21	24.75	25.73	27.27	28.91	24.48
bonn 92	4.89	4.78	**4.59**	4.79	5.11	6.18	9.44	14.47	20.01	25.36	5.64
bonn 93	**8.70**	8.81	9.06	9.48	10.87	12.87	15.56	19.56	24.44	29.85	11.87
bonn 94	7.50	7.29	**6.87**	7.02	8.56	11.33	15.23	25.29	35.74	41.45	9.94
bonn 95	**12.91**	13.12	13.82	14.97	16.79	18.18	20.20	23.36	28.66	32.56	17.49
bonn 96	22.36	6.64	**6.54**	6.66	16.07	22.85	25.32	28.14	29.20	32.84	22.61
bonn 97	4.43	4.34	4.04	**3.96**	4.09	4.71	6.23	8.69	12.49	28.85	4.57
bonn 98	8.89	**8.84**	9.28	10.46	16.56	24.02	29.56	38.62	48.75	59.57	20.29
bonn 99	**4.02**	4.03	4.03	4.12	4.51	19.40	20.08	21.96	39.98	43.07	11.96
Median	10.75	10.26	10.28	12.23	14.69	20.04	23.97	26.16	31.32	36.93	19.66

Table B.8. Summary of segmentation error rates for fuzzy K-Means, DT-CWT depth 2. The transform uses the Kingsbury filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	1.10	0.94	**0.78**	0.82	0.90	0.96	1.32	2.33	3.77	5.79	1.03
D5-D92	2.81	2.71	**2.61**	2.70	3.47	5.36	8.04	10.84	14.27	17.32	4.41
D8-D84	0.87	0.84	0.79	0.72	0.70	**0.61**	0.66	0.63	0.65	0.84	0.71
D12-D17	2.60	2.53	2.31	2.12	2.08	2.00	**1.95**	2.10	2.50	2.97	2.22
My 5a	12.35	12.40	**12.05**	16.17	18.69	23.11	34.58	31.90	23.84	32.52	20.90
My 5b	7.28	7.09	6.76	**6.71**	7.09	7.02	8.14	17.02	41.08	55.06	7.18
Nat 10	**38.83**	39.27	41.05	44.31	59.60	62.55	72.44	75.62	80.44	80.61	61.07
Nat 10v	**37.74**	40.51	47.98	47.00	61.58	64.60	67.48	67.90	70.22	72.92	63.09
Nat 16b	55.10	**50.34**	54.49	64.24	68.53	75.07	78.79	81.39	82.81	83.72	71.80
Nat 5b	7.45	**7.25**	7.53	9.63	19.75	32.04	37.62	56.20	54.83	63.84	25.90
Nat 5c	7.69	7.56	**7.51**	7.51	8.04	26.10	45.61	47.08	49.49	43.99	17.07
Nat 5m	**7.73**	7.73	7.99	8.72	13.29	21.45	30.52	42.41	44.91	54.32	17.37
Nat 5v2	**32.89**	32.93	33.94	35.70	37.63	40.62	46.88	54.72	55.63	59.94	39.13
Nat 5v3	**12.09**	12.66	15.69	20.04	33.45	51.72	62.30	62.57	63.13	64.01	42.58
Nat 5v	**12.59**	12.95	13.83	16.27	21.89	28.23	35.91	45.58	47.58	48.97	25.06
bonn 00	24.20	24.38	25.03	25.02	24.63	24.42	19.17	**18.84**	25.46	27.42	24.53
bonn 01	25.10	24.80	23.02	**18.09**	19.89	27.40	40.83	42.56	57.07	58.13	26.25
bonn 02	**12.09**	12.09	12.43	13.20	14.67	25.49	28.78	30.49	32.13	37.53	20.08
bonn 03	**4.58**	4.59	4.62	5.05	5.96	7.47	9.49	25.93	26.57	37.24	6.71
bonn 04	31.29	31.13	30.73	30.63	31.79	33.14	33.29	30.15	36.97	**27.56**	31.21
bonn 05	10.22	9.86	9.30	**8.94**	10.86	17.36	22.58	27.70	31.20	33.35	14.11
bonn 06	4.53	4.41	**4.33**	4.39	4.94	6.23	8.31	19.87	35.46	48.16	5.58
bonn 07	7.23	6.96	6.28	5.87	32.78	5.00	33.36	**4.68**	34.29	9.09	7.10
bonn 08	23.76	**23.75**	24.54	29.24	50.11	56.65	57.17	57.88	59.59	61.42	53.38
bonn 09	27.96	28.00	**16.34**	17.47	29.24	30.62	34.26	36.69	39.08	41.19	29.93
bonn 10	**26.15**	26.27	26.59	27.34	29.65	33.16	54.20	54.09	54.52	58.75	31.41
bonn 11	32.54	32.80	33.12	33.17	32.82	32.28	**3.54**	24.50	31.32	33.30	32.67
bonn 12	8.05	7.76	7.47	**7.45**	8.74	12.44	18.89	20.45	30.84	33.85	10.59
bonn 13	5.48	5.40	**5.35**	5.87	7.23	31.02	28.79	37.54	38.28	40.68	18.01
bonn 14	6.53	6.32	5.79	**5.34**	5.39	23.77	25.60	27.38	28.96	32.13	15.15
bonn 15	4.79	4.72	4.57	**4.48**	4.58	4.82	5.47	6.48	8.59	19.65	4.80
bonn 16	**6.82**	6.84	6.97	7.26	7.81	8.56	9.96	11.90	26.07	45.37	8.18
bonn 17	7.13	6.93	6.37	**6.17**	7.12	9.58	13.96	18.06	26.18	28.30	8.36
bonn 18	**21.31**	23.07	25.45	26.49	27.30	27.80	46.47	47.74	48.15	48.79	27.55
bonn 19	26.86	26.30	24.36	21.84	19.04	**10.14**	28.82	36.68	40.26	41.87	26.58
bonn 20	29.38	29.25	**28.68**	34.25	33.93	33.40	33.11	35.01	35.49	34.99	33.66
bonn 21	10.86	10.78	10.63	**10.53**	11.11	12.45	14.94	18.21	19.72	21.32	11.78
bonn 22	16.03	**15.88**	16.26	16.63	17.87	20.73	35.08	36.63	34.06	46.44	19.30
bonn 23	8.90	27.86	27.19	26.61	**7.93**	8.00	30.25	29.79	14.43	18.31	22.46
bonn 24	21.92	22.40	**13.42**	13.52	23.22	22.99	22.15	20.94	18.73	19.01	21.43
bonn 25	**15.03**	15.13	15.71	16.91	23.20	23.30	23.49	24.15	37.42	44.51	23.25

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 26	5.97	5.93	5.79	**5.78**	25.89	24.24	8.09	24.24	28.45	37.11	16.17
bonn 27	9.70	9.58	9.28	**9.00**	9.20	9.93	22.45	24.72	27.85	31.76	9.82
bonn 28	**28.85**	28.94	28.86	29.05	29.78	30.21	35.27	37.51	39.82	43.62	30.00
bonn 29	4.86	4.71	4.38	4.11	**3.83**	4.14	6.72	11.52	18.35	20.64	4.79
bonn 30	**25.85**	29.16	30.88	31.88	29.33	34.01	34.61	34.20	46.50	40.20	32.95
bonn 31	3.67	3.56	3.45	**3.36**	3.71	19.20	20.43	36.82	39.42	40.78	11.46
bonn 32	12.29	**12.26**	12.29	13.17	19.83	26.19	32.13	34.81	38.32	47.00	23.01
bonn 33	6.21	6.06	5.88	**5.79**	22.39	22.39	21.86	19.31	18.92	16.56	17.74
bonn 34	9.54	9.33	**9.19**	11.45	23.01	21.04	33.94	35.77	37.38	38.77	22.02
bonn 35	5.34	5.19	4.88	4.61	4.30	**4.26**	4.87	7.42	18.25	20.75	5.04
bonn 36	7.05	**6.95**	7.09	7.81	9.83	14.01	31.91	34.69	37.85	42.55	11.92
bonn 37	3.49	3.38	3.18	**3.15**	3.35	3.74	4.39	8.04	21.37	58.97	3.61
bonn 38	**7.86**	23.93	8.03	8.41	10.40	14.45	15.48	16.39	19.01	32.46	14.97
bonn 39	2.37	2.29	2.11	1.99	1.81	**1.79**	1.87	3.02	34.47	42.10	2.20
bonn 40	33.53	4.41	3.99	3.71	**3.44**	32.60	32.64	32.45	31.00	28.64	29.82
bonn 41	**14.45**	14.50	15.00	33.40	18.49	21.20	25.10	31.82	47.87	50.29	23.15
bonn 42	22.29	22.80	23.43	23.35	23.50	25.73	24.80	**22.05**	25.08	22.35	23.39
bonn 43	8.42	8.25	**8.21**	9.04	11.77	30.28	31.43	33.54	49.83	50.75	21.03
bonn 44	**9.91**	9.93	10.33	11.06	15.01	17.18	19.92	23.69	27.29	46.60	16.09
bonn 45	13.70	13.59	**13.41**	13.54	14.67	17.16	25.09	39.33	40.24	41.38	15.92
bonn 46	**27.63**	28.20	29.25	31.50	29.10	30.35	33.51	34.99	35.58	36.50	30.92
bonn 47	6.60	**6.48**	6.55	31.83	8.10	10.68	19.58	24.78	35.74	39.02	15.13
bonn 48	21.93	22.87	23.63	23.43	22.72	22.12	21.23	19.64	**17.36**	37.30	22.42
bonn 49	8.38	**8.29**	8.44	9.20	12.20	15.83	34.56	37.71	46.37	46.95	14.01
bonn 50	9.12	9.02	**8.81**	8.95	9.59	28.05	30.54	37.55	41.49	44.32	18.82
bonn 51	4.63	4.53	4.28	**4.17**	4.23	4.98	6.62	16.31	25.97	31.01	4.81
bonn 52	25.24	25.13	4.60	**4.38**	4.55	6.57	16.89	27.58	37.70	39.32	21.01
bonn 53	34.69	34.61	**34.59**	34.69	34.86	35.14	35.84	36.79	38.49	37.88	35.00
bonn 54	7.85	7.81	**7.65**	7.87	9.34	13.65	16.14	17.10	18.30	19.82	11.50
bonn 55	6.51	6.34	6.10	**5.85**	5.98	6.45	27.15	28.84	31.37	51.54	6.48
bonn 56	39.70	38.95	**37.90**	38.57	40.25	42.55	48.12	48.90	49.73	51.02	41.40
bonn 57	**5.82**	5.88	6.13	6.38	8.02	16.49	30.26	46.35	54.59	59.34	12.26
bonn 58	8.76	**8.58**	8.74	9.05	9.83	11.55	16.40	22.11	21.55	25.41	10.69
bonn 59	10.78	**10.55**	10.58	10.60	10.92	12.51	26.54	32.46	29.27	35.39	11.72
bonn 60	23.97	24.40	22.82	23.12	22.80	25.65	**22.22**	27.22	32.94	46.58	24.18
bonn 61	6.51	6.29	**5.91**	22.13	21.94	21.54	12.18	19.26	18.25	17.63	17.94
bonn 62	7.56	**7.48**	8.02	17.60	23.74	30.19	26.82	29.58	48.26	49.54	25.28
bonn 63	6.36	6.29	**6.24**	6.24	7.03	8.97	13.64	16.62	16.87	31.21	8.00
bonn 64	11.96	11.88	**11.79**	12.17	13.27	17.12	26.22	33.57	31.61	33.21	15.19
bonn 65	4.72	4.61	4.37	4.14	3.84	**3.73**	17.14	16.77	15.77	15.92	4.66
bonn 66	2.67	2.58	2.42	2.29	**2.26**	2.34	2.81	3.68	16.02	28.09	2.62
bonn 67	10.13	**10.07**	10.39	23.43	24.13	16.15	24.88	25.13	25.51	25.77	23.78
bonn 68	10.72	10.68	**10.59**	27.40	28.56	29.43	30.17	30.87	31.44	31.71	28.99
bonn 69	**7.56**	7.63	8.69	8.96	40.99	51.00	49.37	57.60	57.03	59.20	45.18
bonn 70	**7.91**	8.00	8.39	9.18	11.35	15.75	22.27	28.27	35.48	41.22	13.55
bonn 71	8.01	7.89	7.59	7.39	**7.31**	8.08	12.20	13.17	23.27	17.83	8.05
bonn 72	4.59	4.46	4.26	**4.17**	4.23	4.96	7.07	9.86	29.02	31.87	4.77
bonn 73	**15.28**	16.02	25.31	23.63	35.80	37.16	39.70	41.89	42.91	56.62	36.48
bonn 74	5.37	5.21	4.86	**4.66**	5.09	7.50	17.60	37.59	40.14	41.81	6.43
bonn 75	4.33	4.29	**4.24**	4.31	5.00	5.87	7.24	16.27	29.92	48.80	5.44
bonn 76	**7.06**	7.14	34.76	34.70	34.84	17.46	23.29	36.15	39.21	37.73	34.73
bonn 77	32.41	**32.40**	32.93	32.66	33.44	34.70	36.54	36.82	37.40	38.00	34.07
bonn 78	23.61	23.62	5.89	5.58	**5.33**	16.95	15.67	16.80	18.25	20.18	16.87
bonn 79	**12.02**	12.36	13.52	15.86	20.02	23.57	26.45	29.89	35.69	37.15	21.79
bonn 80	5.62	5.42	5.18	**5.01**	5.06	5.78	8.43	17.60	17.72	16.63	5.70
bonn 81	25.82	7.57	**7.29**	22.07	20.72	18.09	21.12	34.88	45.06	56.30	21.60
bonn 82	32.64	33.01	37.96	34.43	**21.33**	33.75	46.74	45.78	46.73	46.72	36.20
bonn 83	6.05	5.87	5.53	**5.25**	5.33	23.17	7.38	9.17	12.03	18.15	6.72
bonn 84	7.56	7.29	6.65	6.32	**5.91**	7.16	9.93	12.67	15.56	16.45	7.42
bonn 85	9.53	**9.45**	9.95	11.96	16.80	22.22	28.10	50.36	52.09	52.20	19.51
bonn 86	21.23	21.53	22.14	22.14	22.19	18.38	17.46	**16.93**	17.83	35.93	21.38
bonn 87	31.65	31.39	30.98	30.74	**12.11**	33.09	36.02	36.54	36.22	33.41	32.37
bonn 88	7.08	6.88	6.62	**6.40**	23.72	7.48	22.37	26.04	24.35	22.94	14.93
bonn 89	5.52	5.26	4.85	4.42	**4.34**	4.46	4.99	5.88	8.56	17.07	5.12
bonn 90	4.74	4.60	4.47	4.39	**4.35**	5.11	9.07	29.61	36.95	38.05	4.92
bonn 91	**12.32**	12.74	26.08	25.60	24.35	22.45	22.37	22.20	22.06	22.65	22.41
bonn 92	4.68	4.56	4.41	**4.33**	4.35	12.36	14.96	15.17	15.92	18.07	8.52
bonn 93	**7.61**	7.80	7.93	8.50	10.16	12.48	33.28	44.49	45.86	48.41	11.32
bonn 94	6.30	6.15	6.00	**5.84**	6.54	8.65	32.43	35.60	39.93	44.18	7.59
bonn 95	**12.86**	13.00	13.84	14.79	15.54	22.02	22.76	26.21	31.32	34.97	18.78
bonn 96	22.31	22.19	16.86	**16.44**	16.62	18.84	17.77	17.16	20.18	26.62	18.30
bonn 97	4.13	4.00	3.78	3.65	**3.41**	3.65	4.08	4.93	15.03	33.26	4.04
bonn 98	7.57	7.44	**7.35**	7.68	10.75	18.58	28.86	38.38	50.79	49.60	14.66
bonn 99	4.36	4.21	3.94	3.69	**3.48**	17.08	15.73	16.60	36.39	36.94	10.05
Median	8.76	8.29	8.39	9.18	12.20	18.38	23.29	28.27	34.06	37.53	17.37

B.3 Modified K-Means

Table B.9. Summary of segmentation error rates for modified K-Means, DWT depth 3. The transform uses the Kingsbury filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	0.17	0.16	0.13	**0.06**	0.07	0.06	0.06	0.06	0.09	0.15	0.08
D5-D92	**0.38**	0.40	0.46	0.48	0.46	0.45	0.49	0.51	0.49	0.49	0.47
D8-D84	0.35	0.38	0.38	0.39	0.38	0.33	0.34	0.33	0.32	**0.24**	0.35
D12-D17	0.33	0.34	0.31	**0.28**	0.28	0.34	0.35	0.39	0.34	0.29	0.34
My 5a	4.71	20.59	**4.38**	22.00	20.20	6.33	7.30	8.58	11.29	15.75	9.94
My 5b	3.81	3.64	3.42	3.10	3.10	3.08	3.00	2.73	2.78	**2.70**	3.09
Nat 10	3.60	3.77	3.60	3.45	2.85	2.73	2.66	2.70	**2.20**	2.47	2.79
Nat 10v	1.97	1.93	1.79	1.72	1.71	1.67	**1.61**	1.62	2.18	1.77	1.75
Nat 16b	5.65	5.59	**5.56**	5.70	5.80	8.24	7.98	7.03	15.41	17.13	6.41
Nat 5b	4.30	4.23	**4.14**	4.15	4.36	4.70	5.30	5.83	7.35	8.91	4.53
Nat 5c	4.55	4.41	4.13	**4.09**	4.17	4.38	4.46	4.74	5.11	5.64	4.43
Nat 5m	4.19	4.11	**4.06**	4.14	4.30	4.57	4.97	7.38	12.91	19.91	4.43
Nat 5v2	6.65	6.53	**6.45**	6.74	7.03	7.95	8.77	10.42	8.97	12.99	7.49
Nat 5v3	5.43	5.75	**4.75**	4.92	5.63	6.34	13.96	8.78	17.47	34.94	6.05
Nat 5v	3.45	3.40	**3.24**	3.30	3.47	3.75	3.91	4.29	5.37	6.13	3.61
bonn 00	17.86	17.75	5.91	17.59	23.47	**2.27**	14.34	20.87	2.31	2.39	15.97
bonn 01	3.70	3.50	3.82	3.81	**3.06**	3.37	4.31	5.52	9.52	20.43	3.82
bonn 02	24.87	4.15	3.60	3.41	3.41	3.36	6.07	**3.20**	3.33	3.49	3.45
bonn 03	3.28	3.23	3.04	2.90	2.90	**2.80**	2.88	25.26	3.44	4.14	3.13
bonn 04	3.71	3.98	3.41	3.36	3.24	3.19	3.22	3.28	**3.04**	26.98	3.32
bonn 05	4.16	3.64	3.56	3.56	3.41	3.36	**3.36**	3.54	22.67	27.66	3.56
bonn 06	3.52	3.43	3.23	21.50	2.97	**2.95**	2.97	23.29	3.81	4.74	3.48
bonn 07	3.12	3.05	3.00	2.77	2.58	2.44	2.29	2.22	**2.17**	2.30	2.51
bonn 08	22.36	19.82	**4.74**	23.82	21.27	23.39	9.20	11.19	29.30	30.02	21.81
bonn 09	18.24	5.43	17.03	18.12	25.32	**5.23**	24.46	13.75	11.94	23.76	17.57
bonn 10	**3.48**	17.51	17.35	16.95	21.83	3.58	3.63	18.65	18.60	3.76	17.15
bonn 11	3.32	3.40	3.19	3.09	2.92	2.72	**2.64**	2.72	3.13	3.42	3.11
bonn 12	2.81	2.82	2.83	**2.78**	2.97	2.89	3.02	3.50	4.60	7.26	2.93
bonn 13	17.94	17.93	**2.25**	2.31	2.43	2.51	2.81	2.96	3.34	3.96	2.89
bonn 14	2.63	2.59	2.34	**2.14**	2.28	2.18	2.22	2.37	2.64	3.17	2.36
bonn 15	23.36	21.83	1.59	**1.52**	1.59	2.48	21.83	1.82	24.02	2.56	2.52
bonn 16	2.29	2.24	2.17	2.06	1.90	1.77	1.65	1.57	**1.50**	1.53	1.83
bonn 17	3.31	3.28	3.25	2.95	2.71	2.55	2.40	**2.33**	2.33	2.81	2.76
bonn 18	4.35	4.77	5.69	**4.19**	4.77	5.56	5.35	4.93	5.93	8.52	5.14
bonn 19	4.86	19.56	21.53	24.63	2.78	**2.34**	2.83	2.43	4.16	2.98	3.57
bonn 20	5.37	30.69	4.42	**4.14**	30.12	29.97	29.75	29.69	4.19	23.19	26.44
bonn 21	4.85	4.81	4.63	4.49	4.28	**4.14**	4.66	5.03	24.44	4.38	4.64
bonn 22	**6.84**	13.02	30.08	13.20	9.75	10.58	11.62	30.60	30.69	12.07	12.55
bonn 23	6.30	3.34	3.61	3.30	2.94	**2.42**	2.53	2.56	2.63	2.84	2.89
bonn 24	3.23	3.06	2.81	2.52	2.56	2.29	2.63	**2.26**	22.45	2.60	2.62
bonn 25	3.68	3.91	3.85	19.64	**3.27**	3.69	15.58	3.66	3.58	25.32	3.77
bonn 26	2.79	2.72	2.52	2.86	22.49	2.32	**2.27**	2.31	21.64	24.23	2.76
bonn 27	4.72	4.66	27.94	29.16	28.28	27.27	26.71	26.81	4.14	**3.91**	26.76
bonn 28	18.47	18.38	10.30	17.85	**3.99**	25.90	4.69	7.22	25.83	26.79	18.12
bonn 29	3.04	2.92	2.78	2.63	2.34	2.25	2.28	**2.16**	2.70	2.42	2.53
bonn 30	3.86	3.67	3.93	3.56	3.18	3.56	3.06	**2.69**	3.52	3.48	3.54
bonn 31	3.72	**3.66**	3.72	29.66	3.67	30.08	31.57	28.88	28.61	4.73	16.67
bonn 32	4.41	4.39	4.24	24.48	4.08	4.16	**3.88**	4.05	10.56	32.48	4.32
bonn 33	17.96	2.77	2.85	3.62	2.47	2.24	22.61	22.85	**2.21**	5.93	3.23
bonn 34	7.30	7.25	7.13	6.93	6.59	**5.22**	6.46	6.38	5.81	6.99	6.76
bonn 35	2.98	2.62	2.55	2.30	2.12	**2.12**	2.23	2.22	2.46	2.62	2.38
bonn 36	7.24	2.32	2.13	2.00	**1.99**	2.03	2.74	3.11	2.79	3.18	2.53
bonn 37	1.95	1.90	1.77	1.67	1.62	**1.59**	1.68	1.73	1.82	2.01	1.75
bonn 38	20.75	22.64	3.81	5.41	3.26	3.06	**2.81**	2.87	2.98	17.21	3.54
bonn 39	2.38	2.28	2.14	1.92	1.76	**1.71**	1.76	1.90	2.15	2.22	2.03
bonn 40	2.64	2.49	2.33	2.16	1.95	1.90	**1.85**	1.85	1.90	1.93	1.94
bonn 41	4.38	4.30	**3.86**	3.94	24.66	26.28	4.08	4.82	7.23	7.76	4.60
bonn 42	17.97	17.72	17.29	14.65	**1.90**	22.49	24.16	24.87	24.49	2.75	17.85
bonn 43	3.14	3.14	**3.08**	3.14	3.26	3.60	3.88	4.72	5.51	7.68	3.43
bonn 44	3.40	22.96	3.14	2.72	2.50	**2.48**	2.50	2.67	2.93	3.23	2.82
bonn 45	13.88	13.56	12.84	11.74	6.91	**6.71**	7.07	8.11	8.69	10.61	9.65
bonn 46	5.74	22.31	22.49	22.94	22.64	21.37	4.41	**4.08**	27.10	27.50	22.40
bonn 47	3.27	3.22	2.95	3.23	2.70	**2.62**	3.52	3.92	2.92	3.19	3.20
bonn 48	18.20	19.31	23.74	1.80	**1.61**	1.63	24.12	24.38	2.11	23.60	18.76
bonn 49	3.34	3.22	**3.11**	3.60	3.57	3.58	4.69	4.19	5.65	4.97	3.59
bonn 50	5.19	5.01	4.82	4.53	4.35	3.92	3.76	3.66	**3.58**	3.67	4.13
bonn 51	2.26	2.13	1.97	1.87	1.67	**1.59**	1.70	1.81	1.89	2.00	1.88
bonn 52	2.87	2.81	2.70	2.43	**2.23**	2.25	2.34	24.33	2.87	24.29	2.76
bonn 53	3.88	3.78	3.79	3.56	3.58	3.55	**3.23**	3.27	3.66	4.93	3.62
bonn 54	3.58	3.50	3.27	3.17	3.14	**2.93**	3.28	21.70	3.83	3.16	3.27
bonn 55	2.65	2.73	2.87	2.44	2.24	2.20	2.34	**1.97**	6.69	2.40	2.42
bonn 56	7.13	6.22	5.27	6.41	5.67	5.33	**5.00**	5.87	7.04	9.40	6.04
bonn 57	4.62	**4.35**	4.51	5.10	4.46	5.02	5.26	5.84	8.06	7.75	5.06
bonn 58	3.32	3.28	3.19	3.01	3.07	2.73	**2.64**	2.64	3.42	3.96	3.13
bonn 59	2.30	2.22	2.06	2.01	**1.98**	1.99	2.14	2.48	2.69	3.28	2.18
bonn 60	20.53	20.33	**2.95**	20.01	23.15	23.54	3.00	3.17	5.83	4.62	12.92

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 61	3.43	3.79	19.24	3.11	23.51	**2.76**	20.38	2.98	20.63	20.78	11.52
bonn 62	3.34	21.62	**2.93**	2.97	22.03	3.29	24.39	24.49	29.33	30.28	21.83
bonn 63	3.84	2.90	3.25	2.57	2.23	2.95	**2.09**	2.29	2.78	3.48	2.84
bonn 64	5.90	5.81	5.68	**5.55**	5.62	5.76	5.80	5.68	6.05	5.93	5.78
bonn 65	20.47	16.63	3.17	20.77	**2.98**	3.01	3.03	3.03	3.19	23.69	3.18
bonn 66	2.11	2.09	2.00	1.92	1.81	**1.65**	21.77	22.00	2.31	2.10	2.09
bonn 67	6.43	3.75	18.85	18.66	19.44	2.80	**2.63**	20.06	3.80	3.99	5.21
bonn 68	3.63	19.69	19.70	19.32	19.03	19.53	18.92	19.24	24.87	**2.65**	19.28
bonn 69	3.41	3.60	3.08	**2.91**	3.16	3.58	3.56	5.58	7.99	10.36	3.57
bonn 70	2.40	2.28	2.22	2.11	2.02	2.03	**1.99**	2.05	2.20	2.40	2.15
bonn 71	3.75	3.38	3.24	3.00	**2.93**	3.06	3.17	3.05	3.00	22.47	3.12
bonn 72	3.19	3.19	2.95	2.91	2.95	2.87	**2.86**	3.05	20.56	21.25	3.00
bonn 73	4.66	5.16	5.00	5.03	5.29	**3.62**	3.71	4.19	4.88	4.66	4.77
bonn 74	2.53	2.44	2.40	**2.35**	2.80	2.97	3.48	4.10	5.57	5.18	2.88
bonn 75	2.64	3.02	24.86	2.34	2.29	**2.26**	4.31	2.61	2.78	25.16	2.71
bonn 76	3.39	3.34	3.06	3.00	**2.95**	3.08	3.59	4.75	5.97	8.16	3.37
bonn 77	5.29	23.77	25.32	25.69	25.31	**5.12**	27.80	5.21	27.56	28.08	25.31
bonn 78	16.41	17.88	16.78	16.49	3.05	3.06	**2.97**	28.75	28.63	29.41	16.64
bonn 79	3.86	4.19	3.78	5.80	5.85	**3.33**	3.49	4.02	6.46	27.04	4.10
bonn 80	3.07	2.97	2.78	2.64	2.45	2.36	**2.29**	2.52	23.12	23.73	2.71
bonn 81	2.44	2.37	2.28	2.49	2.28	**2.27**	2.62	3.16	24.15	26.06	2.47
bonn 82	3.26	21.92	3.30	2.64	**2.25**	24.35	25.25	25.40	2.55	24.60	12.61
bonn 83	6.23	19.42	6.41	3.99	3.50	3.60	21.03	2.84	**2.80**	3.06	3.79
bonn 84	3.33	3.15	3.00	2.79	2.63	2.42	**2.26**	2.31	2.44	2.70	2.66
bonn 85	**3.05**	3.20	3.74	3.65	3.54	3.41	3.70	4.08	4.71	5.45	3.68
bonn 86	3.00	21.13	21.71	23.52	23.49	3.12	6.09	**2.94**	22.64	3.11	13.61
bonn 87	6.23	6.13	6.50	6.87	6.21	27.23	5.58	**5.31**	6.70	30.35	6.36
bonn 88	4.64	4.50	4.36	4.19	4.13	**3.56**	4.16	22.83	3.72	4.27	4.23
bonn 89	4.25	4.09	3.96	3.70	3.09	2.89	**2.78**	2.88	3.19	3.80	3.45
bonn 90	2.84	2.82	2.82	2.73	**2.65**	2.72	2.91	3.32	3.84	4.95	2.83
bonn 91	4.02	3.66	3.83	21.30	21.41	**3.08**	3.50	3.44	23.94	24.68	3.93
bonn 92	3.04	18.47	18.60	19.14	22.23	2.37	2.36	**2.33**	2.43	2.93	2.98
bonn 93	4.20	4.21	3.96	4.26	4.82	4.63	**3.87**	5.26	5.85	4.06	4.24
bonn 94	3.19	3.27	2.89	2.90	**2.87**	2.98	2.98	3.27	3.63	5.15	3.08
bonn 95	5.68	5.62	5.91	6.30	4.55	4.24	4.59	**4.01**	4.05	24.38	5.11
bonn 96	3.11	3.08	**2.95**	3.02	3.08	22.71	23.76	3.36	3.49	5.29	3.23
bonn 97	1.85	1.75	1.57	**1.50**	1.55	1.51	1.55	1.53	1.59	1.82	1.56
bonn 98	2.25	2.20	2.10	1.99	**1.84**	1.84	1.86	2.31	2.48	3.06	2.15
bonn 99	4.10	4.02	4.05	4.03	4.53	2.67	2.50	**2.39**	23.71	23.78	4.04
Median	3.71	3.75	3.56	3.45	3.14	3.06	3.49	3.54	4.05	4.93	3.54

Table B.10. Summary of segmentation error rates for modified K-Means, DT-CWT depth 3. The transform uses the Kingsbury filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	0.31	0.28	0.12	**0.04**	0.06	0.09	0.11	0.12	0.11	0.15	0.12
D5-D92	0.43	0.45	0.49	0.51	0.41	0.37	**0.35**	0.38	0.39	0.41	0.41
D8-D84	0.40	0.40	0.43	0.50	0.48	0.48	0.41	0.33	0.26	**0.13**	0.41
D12-D17	**0.15**	0.15	0.18	0.23	0.31	0.34	0.34	0.32	0.23	0.16	0.23
My 5a	9.58	12.93	15.20	14.91	12.38	9.49	**4.66**	17.34	5.46	5.96	10.98
My 5b	3.46	3.26	3.01	2.87	2.72	2.56	2.29	2.32	2.30	**2.14**	2.64
Nat 10	2.37	2.33	2.33	2.14	2.06	1.79	1.51	1.47	**1.45**	1.45	1.92
Nat 10v	1.78	1.73	1.59	1.56	1.66	1.51	1.25	1.17	1.11	**1.10**	1.54
Nat 16b	5.31	5.49	5.31	5.16	**5.11**	13.60	5.43	6.20	15.11	15.50	5.46
Nat 5b	2.48	2.41	2.22	**2.12**	2.18	2.25	2.42	2.72	3.13	3.83	2.42
Nat 5c	3.12	2.94	2.79	2.66	2.53	**2.46**	2.51	2.62	2.80	2.95	2.73
Nat 5m	3.14	3.10	3.06	2.98	3.33	**2.97**	3.61	3.05	3.08	4.17	3.09
Nat 5v2	4.53	4.37	4.29	**4.25**	4.86	4.85	4.99	5.43	5.75	7.00	4.86
Nat 5v3	5.32	5.25	5.24	5.19	**5.11**	5.52	7.75	6.79	23.82	17.16	5.42
Nat 5v	3.94	3.83	3.67	3.53	**3.44**	3.60	3.90	4.52	5.07	6.04	3.86
bonn 00	17.86	17.71	17.30	17.14	17.94	13.66	14.34	2.52	**2.25**	2.25	15.74
bonn 01	4.08	3.56	12.05	3.08	3.56	**2.99**	3.70	4.92	3.22	7.03	3.63
bonn 02	3.72	3.61	3.30	3.13	17.95	17.30	22.09	**2.75**	3.25	17.10	3.67
bonn 03	2.96	2.86	2.76	2.58	2.23	2.19	2.13	2.08	**2.06**	2.22	2.23
bonn 04	27.25	4.52	26.31	26.40	26.75	2.77	2.65	2.62	3.53	**2.59**	4.03
bonn 05	3.98	3.81	3.72	3.52	3.30	**3.24**	3.45	3.67	26.76	4.10	3.70
bonn 06	2.87	2.84	2.73	2.70	2.58	**2.29**	2.43	2.58	24.82	25.84	2.72
bonn 07	2.51	2.45	2.36	2.03	1.77	1.65	1.52	1.46	1.37	**1.28**	1.71
bonn 08	22.27	20.77	22.79	23.87	23.64	**6.62**	23.28	12.02	26.35	28.72	23.03
bonn 09	17.44	4.26	4.29	**3.82**	22.53	23.32	23.42	23.43	23.68	23.07	22.80
bonn 10	17.05	6.74	6.39	5.61	25.18	15.91	15.99	3.23	23.93	**3.20**	11.33
bonn 11	3.14	3.08	2.98	5.68	2.69	5.49	5.45	**2.35**	2.43	5.89	3.11
bonn 12	2.94	2.88	2.78	2.78	2.76	**2.72**	2.84	2.99	3.58	5.56	2.86
bonn 13	1.97	1.89	1.89	1.82	**1.80**	21.11	21.97	20.95	20.88	21.60	11.42
bonn 14	2.25	2.17	2.08	1.91	1.76	1.64	**1.62**	1.63	26.61	27.01	1.99
bonn 15	22.66	2.29	24.07	21.97	**2.06**	22.27	22.33	23.74	24.20	24.35	22.49
bonn 16	2.33	2.31	2.22	2.03	1.92	1.76	1.69	1.61	1.61	**1.59**	1.84
bonn 17	2.60	2.50	2.48	2.37	2.26	2.10	2.03	2.01	**2.01**	2.02	2.18
bonn 18	4.36	4.36	4.74	4.71	5.37	4.26	4.36	**4.08**	4.52	4.88	4.44
bonn 19	16.71	2.38	18.44	18.04	18.56	**2.29**	22.21	22.18	2.45	22.36	18.24
bonn 20	23.54	22.75	22.35	21.66	28.08	29.05	8.39	**3.90**	3.96	27.97	22.55

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 21	4.79	4.72	4.62	4.51	4.83	5.11	**4.02**	4.07	4.07	5.83	4.67
bonn 22	**5.87**	5.87	28.46	5.87	28.50	6.70	28.61	28.35	9.73	9.70	9.71
bonn 23	6.88	25.85	4.03	3.62	3.28	2.91	2.82	**2.64**	7.57	3.04	3.45
bonn 24	3.31	3.53	3.46	3.19	2.67	2.53	2.47	**2.35**	2.59	2.55	2.63
bonn 25	2.92	15.52	15.34	2.78	**2.62**	23.81	24.52	24.07	24.01	24.58	19.67
bonn 26	3.10	2.95	2.86	2.73	2.58	2.53	22.69	**2.36**	21.86	6.78	2.91
bonn 27	4.84	4.99	16.06	4.52	4.22	26.83	26.10	**4.14**	4.33	5.53	4.91
bonn 28	17.63	17.56	17.31	16.97	24.61	25.24	**3.30**	20.59	26.14	4.41	17.60
bonn 29	1.82	1.75	1.61	1.61	1.54	1.42	**1.31**	1.31	1.35	1.43	1.49
bonn 30	4.32	4.69	4.24	4.00	2.98	2.86	3.36	**2.58**	2.59	3.81	3.59
bonn 31	4.05	3.99	3.64	3.28	**2.93**	30.11	4.12	28.91	3.83	28.47	4.02
bonn 32	4.74	5.33	4.68	26.14	26.29	4.44	**4.32**	4.55	6.76	29.72	5.04
bonn 33	3.08	4.46	4.22	3.07	21.23	**2.23**	22.17	23.27	22.33	21.84	12.84
bonn 34	5.40	4.62	4.85	4.39	4.28	4.17	3.99	**3.94**	4.00	5.08	4.34
bonn 35	2.85	2.83	2.57	2.53	3.57	2.25	2.24	**2.19**	2.22	2.23	2.39
bonn 36	15.39	2.78	2.75	2.61	**2.26**	24.02	24.48	23.71	23.10	2.56	9.09
bonn 37	1.82	1.78	1.65	1.62	1.61	**1.57**	1.66	1.64	1.64	1.72	1.64
bonn 38	3.41	3.77	3.03	2.77	2.36	**2.12**	3.49	2.17	23.99	2.42	2.90
bonn 39	2.41	2.35	2.23	2.17	1.95	1.80	1.68	1.62	**1.56**	1.59	1.87
bonn 40	2.78	2.72	2.58	2.28	2.00	1.82	1.67	1.62	**1.60**	1.63	1.91
bonn 41	3.29	5.00	3.60	3.30	2.92	**2.89**	3.14	28.80	29.03	29.24	3.45
bonn 42	1.56	1.55	1.43	1.30	1.23	20.85	15.32	**1.18**	1.28	1.48	1.46
bonn 43	2.91	2.86	2.75	**2.62**	2.67	2.68	2.81	3.35	3.62	4.51	2.84
bonn 44	2.72	2.86	2.41	2.31	2.11	**1.95**	2.08	1.97	2.04	3.40	2.21
bonn 45	6.62	6.38	6.16	5.79	5.56	**5.46**	5.65	5.96	6.26	7.16	6.06
bonn 46	19.81	5.27	23.76	4.62	22.34	3.76	**3.45**	26.00	26.29	26.41	21.08
bonn 47	2.85	2.70	2.50	2.44	2.14	1.97	1.84	**1.75**	1.76	28.31	2.29
bonn 48	**1.76**	2.02	18.99	21.66	1.89	21.83	1.79	22.69	2.06	1.87	2.04
bonn 49	3.05	3.02	2.96	2.76	2.56	**2.44**	2.44	2.94	26.31	5.12	2.95
bonn 50	5.43	5.36	5.09	4.91	4.55	4.39	4.15	3.83	3.81	**3.53**	4.47
bonn 51	1.97	1.89	1.84	1.84	1.70	1.65	1.56	**1.53**	1.56	1.81	1.75
bonn 52	3.20	2.98	3.05	2.76	2.50	2.30	2.31	**2.29**	2.36	2.36	2.43
bonn 53	3.91	3.81	3.77	3.60	3.40	3.12	3.05	3.12	**2.95**	27.74	3.50
bonn 54	3.60	3.11	2.99	3.80	2.68	2.47	2.40	21.31	**2.28**	2.36	2.84
bonn 55	3.03	2.99	2.92	2.82	2.66	2.52	2.35	2.19	**2.12**	2.15	2.59
bonn 56	4.69	4.68	4.31	5.19	4.39	4.39	**3.57**	4.59	5.47	5.06	4.63
bonn 57	3.80	3.72	3.61	**3.50**	3.67	3.68	22.35	22.19	4.17	4.26	3.76
bonn 58	2.28	2.28	2.33	2.25	2.20	2.16	2.13	2.15	**2.11**	2.23	2.21
bonn 59	2.15	2.12	2.06	2.01	1.90	1.83	**1.81**	1.83	1.93	1.98	1.96
bonn 60	25.31	23.21	**3.12**	22.82	23.18	26.20	26.10	23.89	24.35	24.85	24.12
bonn 61	3.60	3.16	2.89	2.70	2.56	2.48	2.47	2.40	**2.37**	22.55	2.63
bonn 62	21.63	21.30	3.36	3.22	**3.04**	3.06	3.08	24.38	25.51	27.00	12.33
bonn 63	2.81	**2.71**	3.34	27.26	24.65	23.12	3.24	23.08	3.40	23.49	13.24
bonn 64	6.23	6.02	5.85	5.91	5.78	5.02	**4.70**	5.21	5.22	5.30	5.54
bonn 65	15.04	14.58	3.06	19.04	19.42	14.58	19.20	2.80	**2.48**	2.81	14.58
bonn 66	2.67	2.70	2.52	22.33	18.66	1.73	1.62	**1.57**	23.62	23.76	2.68
bonn 67	6.84	17.99	18.04	18.15	2.46	21.79	2.55	**2.18**	18.79	2.43	12.41
bonn 68	18.46	18.74	3.04	2.99	20.61	**2.17**	17.60	15.20	17.63	23.83	17.61
bonn 69	3.11	3.02	2.88	2.67	**2.50**	2.50	2.71	2.87	3.64	5.19	2.88
bonn 70	2.95	2.85	2.76	2.69	2.50	2.56	**2.45**	2.46	2.47	2.52	2.54
bonn 71	3.08	3.00	2.66	2.45	2.29	2.14	**2.09**	2.09	2.51	19.41	2.48
bonn 72	3.89	16.90	16.65	16.13	3.16	3.14	3.02	2.97	**2.73**	18.05	3.52
bonn 73	2.85	**2.81**	3.45	3.30	3.38	2.93	2.83	4.00	2.98	3.32	3.14
bonn 74	2.27	2.17	2.11	2.13	**2.09**	2.23	2.31	2.59	3.19	3.77	2.25
bonn 75	2.91	24.29	23.82	23.52	23.22	2.13	1.97	1.87	**1.84**	1.93	2.52
bonn 76	2.55	2.51	2.48	2.44	2.45	**2.36**	2.70	3.01	3.36	4.28	2.53
bonn 77	22.20	22.49	22.17	22.67	4.41	**3.86**	17.24	4.91	26.60	27.66	22.19
bonn 78	3.09	3.05	17.00	16.13	2.40	2.36	24.38	**2.27**	23.27	23.32	9.61
bonn 79	3.75	3.60	4.08	3.50	3.67	**3.08**	5.54	22.85	4.60	25.46	3.92
bonn 80	3.26	3.18	3.08	2.86	2.62	2.39	**2.22**	20.51	21.98	19.69	3.13
bonn 81	21.84	21.84	21.88	3.42	**1.84**	23.20	1.87	2.02	23.97	2.72	12.63
bonn 82	22.90	22.61	22.09	22.19	24.91	24.40	5.69	24.68	24.44	**4.57**	22.75
bonn 83	3.26	3.16	2.87	2.62	2.40	2.23	**2.15**	16.45	2.61	2.25	2.61
bonn 84	2.73	2.67	2.53	2.25	2.09	1.86	**1.73**	1.75	1.75	21.94	2.17
bonn 85	3.39	3.44	3.30	3.13	**3.04**	6.53	8.93	23.72	3.30	25.31	3.42
bonn 86	4.79	3.44	15.40	10.19	2.83	20.25	24.48	**2.61**	2.67	2.88	4.11
bonn 87	5.15	5.07	4.94	4.79	5.80	4.90	5.97	4.82	**4.67**	26.67	5.00
bonn 88	19.71	19.37	4.90	4.22	18.94	3.58	3.43	22.47	**3.16**	3.33	4.56
bonn 89	4.03	3.88	3.43	3.12	2.70	**2.48**	2.53	2.58	2.77	3.00	2.88
bonn 90	2.34	2.28	20.30	2.03	1.91	23.40	**1.80**	1.98	1.88	2.33	2.15
bonn 91	4.03	3.95	3.77	3.63	17.61	**3.42**	8.28	3.47	3.73	6.56	3.86
bonn 92	2.92	2.88	2.73	18.31	2.21	2.17	1.79	23.52	**1.67**	1.76	2.47
bonn 93	4.89	4.16	3.00	2.92	3.36	**2.76**	5.39	21.84	21.30	20.67	4.53
bonn 94	2.76	2.70	2.61	2.50	2.08	1.97	**1.96**	2.04	2.23	2.56	2.37
bonn 95	5.21	5.12	4.64	4.56	4.66	5.29	4.66	4.95	**4.32**	4.37	4.66
bonn 96	4.89	15.00	3.14	2.91	2.83	2.70	2.65	**2.61**	22.85	22.78	3.02
bonn 97	1.64	1.57	1.43	1.30	1.23	1.17	**1.13**	1.18	1.24	1.29	1.27
bonn 98	2.42	2.21	2.14	2.08	2.04	**1.97**	1.98	2.25	2.20	2.42	2.17
bonn 99	4.02	4.07	3.65	2.28	2.02	1.93	1.89	**1.85**	19.75	20.98	2.96
Median	3.39	3.26	3.30	3.13	2.72	2.77	3.02	2.94	3.40	4.41	3.14

Table B.11. Summary of segmentation error rates for modified K-Means, DWT depth 2. The transform uses the Daubechies 9-7 filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	0.15	0.15	**0.01**	0.02	0.05	0.07	0.09	0.15	0.16	0.17	0.12
D5-D92	0.62	0.62	0.62	0.62	0.66	0.59	0.55	0.57	0.56	**0.51**	0.60
D8-D84	0.44	0.45	0.45	0.46	0.46	0.46	0.38	0.38	0.35	**0.31**	0.45
D12D17	**0.35**	0.39	0.44	0.45	0.43	0.41	0.43	0.46	0.46	0.43	0.43
My 5a	**4.97**	13.86	21.58	21.62	5.54	8.83	22.01	25.06	11.58	12.82	13.34
My 5b	3.00	2.94	2.83	2.72	2.65	2.53	**2.43**	2.53	2.60	2.84	2.69
Nat 10	4.23	3.33	3.99	3.72	3.18	3.05	3.42	2.86	**2.82**	2.82	3.25
Nat 10v	1.95	2.16	1.97	1.93	1.82	1.78	**1.42**	1.43	1.57	1.69	1.80
Nat 16b	6.52	6.32	6.09	**5.96**	10.91	7.51	7.12	8.00	9.39	17.92	7.31
Nat 5b	4.02	3.95	3.80	**3.65**	3.75	4.28	4.82	6.07	8.58	11.32	4.15
Nat 5c	5.89	5.75	5.47	5.12	4.81	**4.71**	4.91	5.24	5.68	6.65	5.36
Nat 5m	5.02	4.85	4.64	4.51	4.78	**4.28**	4.47	7.73	12.68	22.88	4.81
Nat 5v2	**6.20**	6.96	6.90	8.15	7.88	7.98	8.37	9.46	11.11	11.58	8.07
Nat 5v3	5.33	5.29	8.72	**5.08**	5.17	13.51	13.06	13.85	8.50	9.63	8.61
Nat 5v	4.15	3.99	3.83	3.69	3.49	**3.39**	3.80	4.10	6.20	8.20	3.91
bonn 00	18.13	17.95	17.66	17.35	17.98	14.06	**2.47**	23.85	2.53	2.67	17.50
bonn 01	22.22	22.20	**3.13**	3.49	14.50	3.34	3.64	3.66	4.44	7.53	4.05
bonn 02	3.45	3.37	3.51	3.19	3.05	3.04	**2.80**	2.80	2.94	3.19	3.12
bonn 03	2.79	2.73	2.66	**2.56**	24.27	2.83	4.89	3.10	3.42	4.43	2.96
bonn 04	3.60	3.67	3.28	3.09	25.81	2.55	29.37	2.52	**2.45**	30.61	3.44
bonn 05	5.40	5.43	5.04	**4.05**	4.51	4.29	4.49	4.07	4.35	29.24	4.50
bonn 06	19.16	2.84	2.64	2.48	**2.30**	3.10	2.46	3.12	3.49	25.73	2.97
bonn 07	2.86	2.81	2.70	2.59	2.52	2.48	2.46	**2.40**	2.51	2.58	2.55
bonn 08	**4.69**	7.20	22.38	23.24	24.46	28.81	24.27	29.25	30.12	29.17	24.37
bonn 09	17.80	4.65	4.72	4.19	23.72	4.06	**4.00**	24.63	24.51	24.55	11.26
bonn 10	17.86	17.58	4.55	4.47	4.17	9.41	22.79	**3.51**	3.52	25.70	6.98
bonn 11	2.51	2.47	2.36	2.14	2.14	**2.07**	2.20	2.37	2.57	3.04	2.37
bonn 12	2.79	2.58	2.49	2.41	2.39	**2.37**	2.73	2.82	4.60	11.63	2.65
bonn 13	18.88	18.80	3.63	**2.44**	21.53	2.75	23.07	25.01	24.39	27.93	20.21
bonn 14	3.16	23.02	23.01	2.72	2.71	2.73	**2.64**	3.58	3.17	3.67	3.17
bonn 15	2.75	2.51	2.34	2.03	**1.89**	1.89	1.92	2.04	2.25	2.37	2.15
bonn 16	2.53	2.45	2.35	2.20	2.08	1.93	1.84	**1.72**	1.72	1.74	2.00
bonn 17	2.40	13.35	2.28	4.19	2.25	2.25	**2.20**	2.31	2.26	2.72	2.29
bonn 18	**5.10**	5.24	6.96	5.15	5.32	5.98	6.70	6.70	5.69	9.78	5.83
bonn 19	21.01	3.34	4.57	**3.07**	3.93	28.62	3.69	3.69	3.80	27.96	3.87
bonn 20	24.73	24.30	4.08	4.03	**3.69**	3.81	3.76	3.72	3.81	4.23	3.92
bonn 21	4.50	4.44	25.50	4.20	3.99	3.98	**3.93**	25.49	23.62	4.23	4.33
bonn 22	8.07	**8.01**	8.02	8.39	8.88	9.20	10.36	11.29	12.63	15.17	9.04
bonn 23	2.99	4.08	2.42	2.28	2.03	**1.73**	1.80	1.95	2.06	2.62	2.17
bonn 24	4.26	4.72	2.89	3.14	2.50	2.37	2.28	**2.18**	23.71	2.56	2.73
bonn 25	3.31	21.13	3.03	2.84	23.25	**2.64**	2.84	2.92	3.33	24.28	3.17
bonn 26	2.11	2.05	1.94	1.93	22.29	1.95	**1.79**	1.81	21.63	22.06	2.00
bonn 27	4.86	4.69	4.52	4.46	**4.38**	4.42	25.29	10.35	5.05	28.61	4.77
bonn 28	19.38	7.29	19.02	4.88	4.82	23.55	**4.68**	28.03	27.23	27.92	19.20
bonn 29	3.00	2.84	2.69	2.51	2.34	2.15	3.85	**2.00**	3.60	22.32	2.77
bonn 30	4.79	3.80	3.66	4.26	3.80	3.71	3.86	**3.12**	4.00	3.98	3.83
bonn 31	3.42	3.25	3.02	3.22	**2.95**	27.48	4.55	3.93	8.73	10.52	3.67
bonn 32	4.44	4.49	4.37	4.14	4.05	5.08	**3.96**	4.13	4.22	5.05	4.30
bonn 33	3.14	4.41	2.73	5.04	19.30	2.44	22.34	23.32	**2.40**	6.24	4.73
bonn 34	6.39	6.40	6.12	5.90	5.57	**5.49**	5.61	5.83	5.59	6.77	5.87
bonn 35	2.74	2.70	2.51	2.32	2.19	**2.13**	2.19	2.39	2.55	2.90	2.45
bonn 36	3.39	3.35	19.02	3.16	2.98	**2.93**	3.15	3.48	27.59	27.64	3.37
bonn 37	2.47	2.40	2.33	2.28	2.18	2.08	**2.04**	2.32	2.43	2.58	2.33
bonn 38	19.87	4.10	19.53	19.19	18.88	3.12	18.71	**2.95**	21.30	23.39	19.03
bonn 39	2.37	2.27	2.14	1.95	1.80	**1.69**	1.71	1.84	1.87	2.10	1.91
bonn 40	2.49	2.45	2.31	2.20	2.05	1.91	1.88	1.88	**1.87**	1.97	2.01
bonn 41	4.53	4.24	3.58	**3.49**	3.92	4.00	4.19	4.94	6.40	6.06	4.22
bonn 42	2.64	2.58	2.41	**2.24**	24.26	2.38	24.92	2.31	23.30	24.17	2.61
bonn 43	4.96	4.94	4.68	4.27	4.15	4.59	**3.91**	4.36	6.60	8.15	4.63
bonn 44	8.70	5.38	21.54	3.36	3.64	3.51	23.03	**3.02**	3.38	4.30	3.97
bonn 45	13.17	7.46	7.41	**6.92**	7.75	7.18	7.36	7.75	8.38	9.81	7.61
bonn 46	5.39	5.26	21.96	4.37	4.77	23.18	4.16	**3.99**	4.06	4.16	4.57
bonn 47	5.18	4.06	4.79	4.82	4.21	4.13	**3.73**	3.86	4.11	5.33	4.17
bonn 48	17.97	19.04	17.47	2.17	**1.97**	23.66	23.61	24.08	23.81	25.41	21.33
bonn 49	3.35	3.23	**3.11**	3.17	3.94	3.61	4.58	4.60	5.49	5.60	3.78
bonn 50	4.42	4.35	4.09	3.77	3.23	2.91	2.67	2.53	**2.47**	2.50	3.07
bonn 51	2.64	2.53	2.39	2.23	2.06	**1.92**	1.92	1.96	2.09	2.27	2.16
bonn 52	2.73	2.64	2.56	2.39	2.18	**2.11**	2.29	2.46	23.83	23.66	2.51
bonn 53	3.27	3.44	3.22	3.31	3.92	3.00	**2.76**	2.95	3.17	28.47	3.25
bonn 54	3.74	3.56	3.91	3.71	3.33	**3.03**	3.05	3.44	5.63	3.19	3.50
bonn 55	2.89	2.84	2.79	2.58	2.52	2.53	2.47	**2.42**	2.53	2.94	2.55
bonn 56	10.08	10.02	10.14	8.06	8.14	**4.49**	6.90	5.11	9.67	10.80	8.91
bonn 57	5.24	5.22	**5.16**	5.21	5.29	5.44	6.16	7.38	8.74	12.32	5.36
bonn 58	3.77	20.89	4.46	3.38	3.17	3.34	**2.98**	4.33	3.08	5.33	3.58
bonn 59	3.29	3.24	3.09	2.91	3.34	**2.84**	3.02	3.42	3.38	3.94	3.27
bonn 60	19.96	3.32	4.24	**2.95**	23.36	24.45	4.27	3.08	5.12	25.93	4.69
bonn 61	23.71	21.41	2.48	21.22	2.42	**2.30**	2.41	2.38	23.89	2.58	2.53
bonn 62	3.42	3.54	**3.19**	24.22	4.66	3.55	25.93	5.32	28.86	28.29	4.99
bonn 63	4.62	3.38	3.12	5.69	2.68	**2.49**	8.59	9.47	2.74	3.31	3.34
bonn 64	6.57	6.13	6.20	5.97	6.15	5.65	**5.55**	5.65	5.71	5.87	5.92
bonn 65	19.60	4.41	4.22	4.03	4.10	8.82	26.76	**3.99**	4.14	5.58	4.32

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 66	2.23	2.17	2.03	1.99	1.91	**1.84**	1.89	1.96	2.16	2.76	2.01
bonn 67	20.11	3.48	20.03	20.20	20.28	3.39	**3.13**	3.53	3.36	3.78	3.65
bonn 68	3.83	3.80	25.15	3.62	3.29	3.11	**3.09**	3.16	3.49	3.49	3.49
bonn 69	4.15	3.98	3.91	**3.70**	3.70	3.85	4.27	4.95	9.41	6.20	4.06
bonn 70	2.49	2.43	2.13	1.95	1.79	**1.64**	1.64	1.73	1.97	2.65	1.96
bonn 71	3.47	16.56	3.33	3.27	**2.97**	3.00	3.27	4.06	3.25	3.44	3.30
bonn 72	17.74	3.75	3.61	3.49	**3.34**	3.34	3.67	4.21	25.12	26.34	3.71
bonn 73	4.05	4.28	**3.66**	4.09	4.16	4.82	3.92	4.41	5.20	16.33	4.22
bonn 74	2.26	**2.16**	2.17	2.28	2.36	2.46	3.02	4.06	5.58	10.67	2.41
bonn 75	3.33	3.26	3.12	3.02	2.93	**2.82**	2.91	3.20	4.00	4.85	3.16
bonn 76	3.58	3.53	3.33	3.12	**2.91**	9.37	3.97	5.32	7.48	11.15	3.78
bonn 77	5.21	**4.25**	5.13	5.14	24.13	24.98	5.50	5.19	28.77	6.52	5.36
bonn 78	3.34	3.29	3.19	3.16	**3.07**	3.42	6.66	26.84	28.41	29.14	3.38
bonn 79	3.97	3.66	3.56	3.33	**3.24**	3.52	3.52	6.68	3.89	5.25	3.61
bonn 80	2.47	2.34	2.17	2.11	**1.87**	1.89	1.91	23.75	24.08	24.32	2.26
bonn 81	2.51	2.45	21.98	22.02	22.03	22.19	**2.40**	23.38	24.82	27.23	22.02
bonn 82	3.07	2.95	2.86	2.59	2.22	2.41	24.31	**2.16**	24.19	2.49	2.73
bonn 83	4.11	6.77	3.72	3.34	22.67	22.72	22.80	2.81	**2.80**	3.15	3.91
bonn 84	3.45	3.72	3.08	2.97	2.73	2.74	2.70	**2.69**	3.42	3.89	3.03
bonn 85	3.13	3.07	3.50	**3.00**	3.21	3.14	3.41	3.71	4.33	5.40	3.31
bonn 86	15.72	14.63	15.77	**2.45**	15.08	16.09	2.51	2.79	24.23	3.58	14.86
bonn 87	5.66	5.52	26.33	5.46	4.89	**4.83**	24.83	4.98	27.91	28.84	5.59
bonn 88	3.86	3.77	3.58	3.31	3.17	6.55	**2.87**	2.93	3.31	4.63	3.45
bonn 89	3.75	3.64	3.39	3.17	2.93	**2.89**	3.00	3.09	3.43	3.97	3.28
bonn 90	18.44	17.83	2.83	2.79	2.80	2.87	**2.75**	2.83	3.64	4.65	2.85
bonn 91	20.43	3.30	3.30	13.55	**2.83**	23.14	4.66	3.09	3.17	24.83	3.98
bonn 92	2.68	2.61	2.35	2.23	**2.19**	2.39	2.29	2.35	2.38	5.47	2.37
bonn 93	5.47	20.53	20.37	21.07	23.03	21.12	19.98	5.15	**4.95**	5.43	20.18
bonn 94	2.64	2.61	2.61	**2.45**	2.52	2.65	2.90	3.19	4.63	6.45	2.65
bonn 95	5.58	4.27	5.21	4.30	5.87	6.63	4.60	**4.27**	4.36	6.20	4.91
bonn 96	16.11	15.25	2.70	2.61	**2.61**	2.97	3.25	26.32	3.79	27.06	3.52
bonn 97	2.15	2.12	2.00	1.87	1.82	1.86	1.79	**1.74**	1.83	2.01	1.86
bonn 98	2.47	2.44	2.47	2.36	2.22	**2.03**	2.11	2.37	3.19	3.36	2.40
bonn 99	3.61	3.32	14.79	15.16	2.70	**2.58**	2.70	24.15	26.46	24.23	9.20
Median	3.97	3.77	3.56	3.31	3.34	3.12	3.64	3.53	4.14	5.60	3.58

Table B.12. Summary of segmentation error rates for modified K-Means, DT-CWT depth 2. The transform uses the Kingsbury filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	0.12	0.09	**0.00**	0.00	0.00	0.06	0.09	0.10	0.10	0.15	0.09
D5-D92	0.50	0.52	0.61	0.59	0.57	0.55	0.50	0.49	0.45	**0.43**	0.51
D8-D84	0.39	0.40	0.51	0.55	0.61	0.60	0.49	0.44	0.37	**0.28**	0.46
D12-D17	0.22	0.23	0.27	0.33	0.34	0.35	0.34	0.34	0.33	**0.17**	0.33
My 5a	11.91	13.36	11.47	11.29	5.62	**3.90**	12.03	5.06	5.84	13.85	11.38
My 5b	2.87	2.84	2.61	2.43	2.32	2.26	2.17	2.11	**2.03**	2.10	2.29
Nat 10	3.46	3.26	3.45	3.09	2.77	2.91	2.84	2.66	2.39	**2.09**	2.87
Nat 10v	1.73	1.70	1.66	1.69	1.56	1.36	1.22	1.18	1.19	**1.04**	1.46
Nat 16b	5.75	5.70	5.84	**5.44**	5.92	10.31	14.58	16.47	16.26	16.77	8.11
Nat 5b	3.18	3.09	2.91	2.77	**2.76**	2.89	3.19	4.17	5.49	6.74	3.13
Nat 5c	4.05	3.94	3.66	3.42	3.27	3.18	**3.14**	3.30	3.56	3.88	3.49
Nat 5m	3.95	3.86	3.69	3.49	**3.28**	3.31	3.56	3.34	4.90	5.04	3.63
Nat 5v2	5.48	5.10	**4.71**	4.93	5.16	5.47	5.66	7.13	7.42	8.28	5.48
Nat 5v3	5.50	8.08	5.29	**5.18**	5.27	5.65	6.55	7.09	12.96	18.01	6.10
Nat 5v	3.82	3.71	3.55	3.49	3.39	**3.38**	3.60	4.09	4.38	5.94	3.66
bonn 00	17.90	17.77	17.58	17.23	**2.42**	14.65	21.09	2.92	22.47	21.66	17.67
bonn 01	3.34	21.91	21.84	3.11	3.06	**2.55**	3.87	3.31	3.76	3.09	3.32
bonn 02	3.36	3.20	22.01	23.00	3.69	2.81	**2.47**	22.17	2.59	2.62	3.28
bonn 03	2.26	2.21	2.01	1.85	**1.78**	1.81	1.85	1.98	2.26	2.08	1.99
bonn 04	3.46	2.82	3.49	2.73	26.64	2.19	2.10	2.06	26.80	**2.02**	2.77
bonn 05	3.93	3.81	3.87	3.42	3.11	2.72	**2.56**	4.09	3.71	3.52	3.62
bonn 06	4.35	4.24	2.34	**2.09**	2.13	2.09	2.23	2.56	2.70	27.50	2.45
bonn 07	2.55	2.45	2.25	2.11	2.02	**1.93**	2.01	2.01	2.09	2.07	2.08
bonn 08	4.89	21.69	**4.24**	5.01	4.99	24.58	4.41	4.37	26.05	12.66	5.00
bonn 09	17.11	16.75	4.00	**3.92**	21.73	23.86	24.06	24.19	23.71	23.35	22.54
bonn 10	4.80	4.79	4.38	4.19	16.21	21.53	18.18	6.64	**2.95**	2.97	4.79
bonn 11	2.80	2.70	2.51	2.43	2.23	2.08	**2.00**	2.01	2.04	2.17	2.20
bonn 12	2.95	2.91	2.80	2.69	2.55	**2.42**	2.50	2.90	3.75	5.80	2.85
bonn 13	18.01	2.00	1.98	**1.97**	1.98	2.12	2.27	23.31	2.54	16.60	2.20
bonn 14	2.86	2.76	2.56	2.48	2.37	2.25	**2.23**	2.26	27.36	2.91	2.52
bonn 15	2.08	1.98	1.84	1.68	1.50	1.43	1.44	**1.35**	1.49	1.57	1.54
bonn 16	2.64	2.57	2.43	2.19	2.02	1.92	1.82	1.72	**1.70**	1.76	1.97
bonn 17	21.78	21.82	2.26	2.06	2.06	2.07	2.03	2.07	2.08	**1.98**	2.07
bonn 18	4.14	4.13	3.99	3.89	**3.86**	4.08	3.99	4.93	6.43	13.34	4.11
bonn 19	3.28	3.14	18.08	**2.47**	2.87	2.67	23.96	25.26	24.58	24.26	10.68
bonn 20	5.02	4.97	4.75	4.83	3.98	4.27	4.10	3.94	**3.84**	5.37	4.51
bonn 21	4.40	4.36	4.22	4.13	5.06	3.92	**3.60**	25.41	4.73	4.04	4.29
bonn 22	6.51	6.49	**6.21**	6.23	6.60	7.55	8.31	8.67	9.27	10.47	7.08
bonn 23	3.09	2.99	2.72	2.53	2.16	2.01	5.06	1.84	**1.81**	2.27	2.40
bonn 24	17.98	3.02	3.10	7.11	2.82	2.34	23.30	**2.15**	2.16	2.23	2.92
bonn 25	14.45	13.86	2.54	2.39	2.28	23.82	**2.17**	2.25	15.57	22.95	8.20

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 26	2.39	2.34	2.20	2.09	1.98	19.42	**1.82**	21.04	23.07	22.17	2.37
bonn 27	4.64	4.58	4.39	4.20	**4.08**	26.59	4.08	4.25	4.52	28.55	4.46
bonn 28	5.24	**5.00**	17.83	17.45	17.32	26.34	26.25	10.26	5.63	26.39	17.39
bonn 29	2.94	2.86	2.65	2.48	2.22	1.97	1.82	1.72	**1.65**	1.87	2.09
bonn 30	3.72	3.63	3.91	3.31	3.25	3.05	2.96	**2.86**	2.91	3.83	3.28
bonn 31	3.28	3.22	28.58	2.51	**2.49**	27.22	3.67	3.78	4.48	5.07	3.73
bonn 32	5.24	5.85	5.40	5.27	7.06	6.63	6.87	7.60	**4.24**	11.74	6.24
bonn 33	3.10	4.16	2.86	2.65	2.47	20.97	2.19	22.73	**2.15**	2.39	2.75
bonn 34	4.65	4.57	4.18	3.98	**3.49**	3.96	3.98	4.28	4.60	6.90	4.23
bonn 35	2.60	2.65	2.33	2.13	2.11	2.10	**1.96**	1.98	2.04	2.06	2.11
bonn 36	3.44	2.69	2.60	**2.40**	2.44	22.87	2.98	3.10	25.68	5.37	3.04
bonn 37	2.20	2.18	2.03	1.85	1.67	1.59	1.54	**1.50**	1.67	1.79	1.73
bonn 38	21.11	4.44	3.70	3.45	3.11	2.91	3.99	**2.76**	23.94	9.03	3.84
bonn 39	2.03	1.96	1.85	1.69	1.57	1.51	1.42	1.35	**1.33**	1.46	1.54
bonn 40	2.58	2.51	2.39	2.28	2.18	1.99	1.91	1.97	1.82	**1.81**	2.08
bonn 41	4.65	4.50	4.24	3.36	**2.83**	3.33	28.08	4.08	4.82	28.33	4.37
bonn 42	14.66	2.09	2.04	24.18	15.46	23.48	**1.56**	1.58	2.17	1.75	2.13
bonn 43	3.83	3.62	3.58	3.50	**3.33**	3.39	3.45	3.60	5.48	7.17	3.59
bonn 44	3.56	3.06	2.96	3.09	2.80	2.62	2.42	**2.17**	3.76	6.32	3.01
bonn 45	6.95	6.93	6.56	6.14	**5.87**	6.71	6.79	6.89	7.44	8.14	6.84
bonn 46	20.29	20.28	5.44	19.58	**4.68**	25.08	26.68	35.25	39.51	26.43	22.68
bonn 47	2.42	2.38	2.29	2.20	2.32	2.33	2.29	**2.20**	2.23	2.41	2.31
bonn 48	2.17	2.17	17.05	18.07	22.87	1.76	**1.71**	1.98	13.29	2.26	2.21
bonn 49	3.14	3.06	2.98	2.85	2.66	**2.55**	3.22	4.08	5.88	5.95	3.10
bonn 50	4.93	4.91	4.37	3.98	3.55	3.16	3.06	2.86	3.06	**2.57**	3.36
bonn 51	2.58	2.55	2.39	2.25	2.00	1.91	1.86	1.76	**1.75**	1.81	1.96
bonn 52	2.92	2.88	2.67	2.44	2.28	1.89	**1.84**	2.11	2.19	2.29	2.28
bonn 53	3.41	3.38	3.34	3.16	3.08	2.80	**2.61**	24.55	3.54	28.56	3.36
bonn 54	3.22	3.22	3.08	3.05	2.95	2.77	21.94	**2.63**	4.96	6.88	3.15
bonn 55	2.90	2.75	2.71	2.61	2.51	2.42	2.28	2.24	**2.23**	2.25	2.47
bonn 56	6.42	6.04	5.07	5.83	**4.48**	5.06	5.44	7.85	10.15	11.15	5.94
bonn 57	3.48	3.42	**3.14**	3.49	3.15	3.28	3.31	3.61	5.18	5.76	3.45
bonn 58	2.76	3.64	2.90	2.89	2.81	**2.72**	3.05	3.01	3.08	3.34	2.95
bonn 59	2.75	2.69	2.61	2.48	2.39	**2.38**	2.42	2.51	2.68	2.85	2.56
bonn 60	2.96	2.88	**2.67**	18.72	19.32	24.51	25.05	24.99	25.16	25.39	21.91
bonn 61	2.58	2.50	2.40	2.29	2.11	**2.10**	18.66	24.27	2.28	24.63	2.45
bonn 62	3.42	3.31	3.13	3.02	**2.93**	3.14	3.28	25.32	25.86	26.37	3.30
bonn 63	3.42	3.34	2.53	2.36	2.06	22.54	22.88	**1.78**	24.42	24.22	3.38
bonn 64	5.93	5.98	5.87	5.69	5.43	5.20	4.98	4.66	**4.66**	4.67	5.32
bonn 65	2.69	2.56	2.32	17.10	15.34	18.44	18.50	**1.62**	19.86	2.61	9.01
bonn 66	1.89	1.87	1.78	1.70	1.65	**1.58**	23.16	23.57	23.61	7.05	1.88
bonn 67	3.81	3.82	3.81	3.09	20.09	22.58	2.97	**2.90**	24.82	3.26	3.81
bonn 68	3.47	3.69	3.61	3.06	22.31	3.07	2.83	2.83	**2.78**	18.59	3.27
bonn 69	3.27	3.20	3.05	2.91	**2.79**	2.98	3.96	3.53	4.06	7.37	3.23
bonn 70	2.92	2.84	2.78	2.48	2.27	2.11	1.89	**1.73**	1.93	5.29	2.38
bonn 71	3.70	3.65	3.45	3.07	2.94	2.78	2.67	**2.63**	22.19	21.81	3.26
bonn 72	3.57	3.59	3.23	3.17	2.91	2.88	**2.75**	2.78	2.78	3.22	3.04
bonn 73	3.52	3.17	3.44	3.33	5.38	3.03	**2.75**	3.57	4.44	5.44	3.48
bonn 74	2.15	**2.04**	2.04	2.05	2.11	2.26	2.52	3.07	4.57	6.40	2.21
bonn 75	2.58	2.54	2.42	2.32	2.28	2.09	**2.00**	2.05	2.26	2.51	2.30
bonn 76	2.58	2.55	2.54	2.50	**2.48**	2.52	2.74	10.17	4.80	7.92	2.56
bonn 77	4.99	4.93	4.75	4.66	4.69	**4.10**	26.06	26.99	5.50	28.34	4.96
bonn 78	3.03	2.98	2.79	3.37	**2.53**	15.13	2.54	23.61	23.46	24.49	3.20
bonn 79	3.49	3.33	3.11	2.93	3.85	3.14	**2.80**	2.87	3.14	25.92	3.14
bonn 80	2.75	2.62	2.52	2.19	1.95	1.80	1.76	**1.74**	1.85	1.95	1.95
bonn 81	22.30	2.29	2.14	2.01	**1.88**	21.84	22.05	2.13	24.51	3.31	2.80
bonn 82	3.47	3.38	3.10	2.93	7.71	2.39	20.65	**1.95**	2.19	8.07	3.24
bonn 83	3.52	5.96	3.04	2.84	2.59	2.44	21.75	22.33	23.32	**2.24**	3.28
bonn 84	3.05	2.95	2.86	2.61	2.37	2.22	2.12	**2.09**	2.11	20.02	2.49
bonn 85	3.14	3.14	2.85	3.03	2.87	**2.76**	3.00	3.17	3.51	3.81	3.09
bonn 86	2.51	16.38	16.13	19.40	20.72	21.90	19.71	**1.88**	2.04	2.10	16.26
bonn 87	5.71	24.36	24.82	24.66	24.04	**4.39**	5.02	5.35	4.75	5.51	5.61
bonn 88	4.26	4.02	3.92	3.65	3.59	3.32	19.21	3.57	**3.17**	3.22	3.62
bonn 89	3.66	3.57	3.09	2.73	2.82	2.53	**2.47**	2.57	2.73	3.00	2.77
bonn 90	2.52	2.45	2.40	2.33	2.30	**2.21**	2.33	2.26	2.75	3.18	2.37
bonn 91	3.82	16.24	3.60	20.69	17.61	**2.67**	2.73	2.81	5.49	17.30	4.66
bonn 92	2.80	2.70	2.51	2.29	21.39	20.90	**1.82**	22.96	1.84	1.92	2.61
bonn 93	4.66	4.53	4.46	4.32	4.64	5.05	4.44	**3.85**	4.15	4.68	4.50
bonn 94	2.58	2.47	2.48	2.31	**2.12**	2.15	2.29	2.44	2.58	3.55	2.45
bonn 95	4.50	4.35	4.19	24.16	4.70	3.99	3.69	**3.68**	24.50	24.78	4.42
bonn 96	3.14	3.00	2.80	2.47	**2.39**	22.84	23.64	21.12	4.01	21.91	3.58
bonn 97	2.00	1.88	1.71	1.61	1.54	1.46	**1.37**	1.38	1.39	1.57	1.56
bonn 98	3.34	2.29	2.31	2.26	2.12	2.02	**2.01**	2.01	2.19	2.36	2.23
bonn 99	14.42	14.45	2.63	2.50	**2.40**	2.40	23.60	23.41	23.96	22.28	14.43
Median	3.42	3.22	3.04	2.93	2.82	2.81	2.97	2.92	3.76	5.04	3.15

B.4 Modified Fuzzy K-Means

Table B.13. Summary of segmentation error rates for modified fuzzy K-Means, DWT depth 3. The transform uses the Kingsbury filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	**0.00**	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.00
D5-D92	**0.00**	0.00	0.00	0.00	0.00	0.04	0.06	0.12	0.17	0.20	0.02
D8-D84	0.18	0.16	**0.13**	0.13	0.13	0.20	0.23	0.22	0.21	0.23	0.19
D12-D17	0.41	0.41	0.41	0.44	0.40	0.37	0.34	0.33	0.33	**0.26**	0.38
My 5a	11.78	**11.71**	14.41	14.37	14.80	15.33	14.77	14.26	14.13	14.62	14.39
My 5b	3.36	3.33	3.18	3.06	2.79	2.57	2.36	2.15	2.06	**1.95**	2.68
Nat 10	2.96	2.93	2.88	2.81	2.68	2.61	2.51	2.42	2.29	**2.22**	2.64
Nat 10v	1.60	1.59	1.53	1.55	1.64	1.60	1.58	1.56	1.50	**1.49**	1.57
Nat 16b	**6.84**	6.86	6.99	7.37	9.31	11.42	13.76	15.96	17.95	19.75	10.37
Nat 5b	4.32	4.25	4.13	**4.10**	4.40	5.37	6.69	9.32	30.72	28.90	4.89
Nat 5c	4.58	4.63	4.55	**4.55**	4.66	4.97	5.68	7.13	9.15	12.91	4.82
Nat 5m	4.73	4.61	**4.56**	4.88	6.25	8.39	12.33	16.88	21.26	25.29	7.32
Nat 5v2	**8.26**	8.42	9.61	9.03	10.70	15.30	27.40	35.22	39.78	41.20	13.00
Nat 5v3	**7.05**	7.28	7.49	8.00	13.38	16.24	20.78	29.49	41.09	39.52	14.81
Nat 5v	3.56	3.50	**3.44**	3.55	3.97	4.48	6.32	9.35	14.17	17.05	4.22
bonn 00	3.75	3.67	3.51	3.33	3.11	2.84	2.79	2.72	**2.70**	2.90	3.00
bonn 01	4.97	4.90	4.57	**4.42**	4.71	5.47	8.16	10.17	13.36	19.36	5.22
bonn 02	3.52	3.45	3.28	3.19	3.05	**2.96**	3.02	3.23	3.70	4.78	3.26
bonn 03	4.02	3.99	3.84	3.80	**3.75**	4.59	16.06	16.25	16.67	17.12	4.31
bonn 04	3.16	3.08	2.88	2.64	2.45	2.41	**2.38**	2.94	3.86	5.33	2.91
bonn 05	3.52	3.37	3.26	3.13	**2.98**	3.64	4.77	6.73	12.19	13.89	3.58
bonn 06	3.08	2.94	**2.84**	2.86	3.06	3.20	3.43	3.87	4.81	6.28	3.14
bonn 07	4.60	4.55	4.36	4.19	3.91	3.72	**3.72**	3.81	4.16	4.75	4.17
bonn 08	**6.38**	6.45	33.72	33.58	27.35	13.65	30.69	30.98	40.47	30.43	30.56
bonn 09	4.56	**4.55**	4.75	5.26	7.64	5.18	5.51	7.44	10.93	12.40	5.38
bonn 10	23.31	23.20	22.38	20.17	9.05	**8.88**	11.81	10.55	11.72	13.71	12.76
bonn 11	3.44	3.34	3.18	3.15	**3.09**	3.11	3.72	4.58	4.93	5.24	3.39
bonn 12	5.92	5.82	5.66	5.66	**5.45**	5.83	6.95	8.99	11.55	13.42	5.88
bonn 13	2.93	2.94	**2.88**	2.91	3.13	3.37	3.60	4.27	14.38	21.43	3.25
bonn 14	3.50	3.47	3.36	3.30	3.25	3.25	**3.22**	3.22	3.31	4.06	3.31
bonn 15	2.77	2.72	2.45	2.37	**2.29**	2.29	2.32	2.43	2.72	3.38	2.44
bonn 16	2.61	2.59	2.52	2.48	2.41	2.38	2.33	**2.28**	2.31	2.33	2.40
bonn 17	4.86	4.85	4.80	4.81	4.90	4.86	**4.74**	5.32	7.12	9.52	4.86
bonn 18	**4.36**	5.83	6.04	6.77	22.64	16.82	20.31	19.81	21.18	22.52	18.31
bonn 19	3.44	3.39	3.05	2.93	2.84	**2.80**	3.31	3.85	3.36	3.74	3.33
bonn 20	2.64	2.58	2.39	2.23	**2.19**	2.33	2.47	2.60	2.93	3.49	2.52
bonn 21	4.02	3.95	3.83	3.78	**3.78**	3.80	3.86	4.16	4.53	5.22	3.91
bonn 22	10.77	10.53	**10.28**	10.91	11.66	13.56	14.94	17.19	20.37	26.76	12.61
bonn 23	3.94	3.83	3.45	3.20	2.80	2.55	**2.36**	3.08	3.44	4.05	3.32
bonn 24	3.42	3.37	3.21	3.09	**3.00**	3.12	3.37	3.89	4.44	5.19	3.37
bonn 25	4.13	4.08	**4.05**	4.06	4.22	4.42	4.81	5.29	5.74	6.48	4.32
bonn 26	2.63	2.60	2.47	2.37	2.14	**2.13**	2.20	2.28	2.43	2.69	2.40
bonn 27	5.13	5.05	4.96	**4.94**	4.99	5.02	5.00	4.98	5.37	19.75	5.01
bonn 28	24.65	24.63	27.68	26.78	25.36	23.72	**22.91**	23.63	25.37	24.19	24.64
bonn 29	4.54	4.47	4.24	3.99	3.78	**3.66**	3.67	3.81	4.06	4.60	4.03
bonn 30	3.19	3.15	**3.12**	3.12	3.50	3.99	4.30	4.81	6.03	7.71	3.74
bonn 31	5.24	5.19	5.12	**5.05**	5.17	5.51	6.24	7.40	15.79	18.32	5.37
bonn 32	4.53	4.46	4.25	4.14	3.88	**3.73**	3.91	4.13	5.16	13.70	4.20
bonn 33	3.63	3.53	3.38	3.29	3.16	3.04	**3.03**	3.06	3.12	3.22	3.19
bonn 34	7.23	7.12	**6.90**	7.15	7.40	12.52	14.81	15.33	17.13	18.72	9.96
bonn 35	3.91	3.81	3.69	3.67	3.46	**3.33**	3.34	3.39	3.45	3.81	3.57
bonn 36	4.07	4.08	3.97	3.78	3.83	**3.68**	3.77	3.88	15.33	6.63	3.92
bonn 37	2.56	2.45	2.38	2.33	2.29	**2.22**	2.23	2.23	2.34	2.51	2.33
bonn 38	5.15	5.05	4.74	4.47	**4.35**	4.96	6.26	7.68	8.67	9.66	5.10
bonn 39	2.55	2.49	2.36	2.28	**2.19**	2.26	2.47	4.26	4.57	4.88	2.48
bonn 40	3.25	3.18	3.08	2.97	2.84	2.76	**2.70**	2.76	2.81	3.01	2.91
bonn 41	7.67	7.60	**6.32**	6.76	7.70	9.84	12.10	30.80	28.25	25.33	8.77
bonn 42	22.31	23.45	3.66	**3.65**	3.77	4.63	7.32	21.53	12.08	19.12	9.70
bonn 43	3.16	3.14	**3.09**	3.19	5.63	14.79	15.18	15.21	15.62	16.86	10.21
bonn 44	4.28	4.19	4.15	4.05	**3.91**	3.96	4.16	4.52	5.11	6.16	4.17
bonn 45	**7.89**	8.14	8.75	8.95	9.23	11.59	15.58	19.29	22.77	22.44	10.41
bonn 46	10.59	10.47	10.30	**8.83**	11.27	13.45	12.49	26.73	25.93	24.96	11.88
bonn 47	4.15	4.11	4.06	4.05	**4.03**	4.28	4.98	6.79	8.70	9.61	4.21
bonn 48	3.33	3.23	3.00	2.88	2.76	**2.70**	2.91	4.50	14.92	16.40	3.12
bonn 49	3.47	3.40	3.38	3.32	**3.31**	3.58	19.20	17.53	17.58	17.68	3.52
bonn 50	4.35	4.33	4.25	4.09	3.92	**3.78**	3.83	3.93	4.24	4.58	4.16
bonn 51	2.39	2.35	2.25	2.17	**2.12**	2.16	2.26	2.54	3.44	4.22	2.31
bonn 52	3.58	3.50	3.35	3.23	3.11	**3.05**	3.23	3.53	4.14	4.94	3.42
bonn 53	5.10	4.86	4.49	4.10	3.69	3.46	**3.41**	7.75	14.29	16.09	4.67
bonn 54	4.77	4.72	4.68	4.64	4.47	4.31	4.34	4.18	**4.16**	4.47	4.47
bonn 55	3.28	3.25	3.15	3.00	2.93	**2.86**	2.89	2.94	3.14	3.75	3.07
bonn 56	**5.59**	5.68	5.88	6.04	6.48	11.54	19.83	20.28	21.11	20.45	9.01
bonn 57	4.87	4.86	**4.77**	4.85	5.07	5.47	5.93	6.93	8.46	11.90	5.27
bonn 58	4.67	4.62	**4.52**	4.54	4.80	5.18	7.21	17.58	17.76	17.36	4.99
bonn 59	2.95	2.92	2.81	**2.75**	2.78	2.84	3.04	3.34	3.64	4.44	2.94
bonn 60	3.64	3.75	3.55	3.47	**3.45**	3.50	3.64	4.11	19.77	20.46	3.64

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 61	3.20	3.19	3.09	3.02	2.90	2.77	**2.76**	2.78	2.84	2.83	2.87
bonn 62	4.00	3.94	**3.80**	3.86	4.44	19.71	25.04	25.16	25.93	27.60	12.07
bonn 63	3.72	3.74	**3.61**	3.63	4.02	4.74	6.34	7.74	9.50	11.91	4.38
bonn 64	6.71	6.56	6.26	5.99	5.74	5.47	5.26	5.04	**4.89**	4.97	5.60
bonn 65	2.69	2.61	2.54	2.51	2.50	**2.41**	2.47	2.73	3.16	3.83	2.57
bonn 66	2.36	2.28	2.22	2.15	**2.11**	2.14	2.22	2.53	2.91	3.53	2.25
bonn 67	4.36	4.36	4.24	4.12	3.93	3.86	**3.80**	4.04	4.39	5.25	4.18
bonn 68	3.95	3.87	3.67	3.48	3.27	**3.15**	12.72	7.48	5.54	6.52	3.91
bonn 69	**4.96**	4.98	5.19	5.85	6.74	8.04	10.05	10.74	13.42	18.68	7.39
bonn 70	2.58	2.59	2.57	2.51	2.49	**2.48**	2.72	2.82	2.84	3.97	2.58
bonn 71	4.53	4.45	4.29	4.19	4.02	3.89	3.83	**3.75**	3.77	3.78	3.96
bonn 72	3.72	3.68	3.52	3.44	3.45	3.51	3.40	**3.34**	3.46	3.67	3.49
bonn 73	7.37	7.15	**6.56**	10.12	13.23	18.80	19.43	23.00	23.93	26.17	16.02
bonn 74	3.34	**2.94**	3.06	3.22	6.59	14.52	15.23	17.90	20.04	22.86	10.55
bonn 75	3.09	3.05	2.92	**2.84**	2.92	3.00	3.17	3.45	4.25	6.92	3.07
bonn 76	3.57	3.48	3.22	**3.17**	3.53	3.86	4.50	5.35	6.56	8.53	3.72
bonn 77	**3.72**	3.72	3.75	3.74	3.86	3.89	3.92	3.83	3.91	5.31	3.85
bonn 78	4.87	4.57	4.33	4.26	**4.21**	4.38	16.37	15.46	15.22	15.34	4.72
bonn 79	3.99	3.90	3.86	**3.82**	3.87	4.11	4.69	5.51	6.45	8.80	4.05
bonn 80	2.87	2.87	2.80	2.70	**2.62**	2.67	2.62	2.72	2.87	3.16	2.76
bonn 81	4.10	**4.09**	4.10	4.16	4.40	4.86	5.43	7.73	15.12	18.59	4.63
bonn 82	2.77	2.73	2.62	**2.56**	2.67	2.65	3.05	3.22	4.21	20.27	2.75
bonn 83	4.20	4.08	3.85	3.61	3.36	**3.22**	3.26	3.48	4.00	4.78	3.73
bonn 84	3.58	3.49	3.27	3.13	**2.96**	3.01	3.48	4.58	6.41	6.90	3.48
bonn 85	6.20	6.11	5.98	6.05	**5.97**	6.22	7.09	18.28	22.55	24.11	6.21
bonn 86	5.43	5.35	5.08	4.90	**4.82**	4.83	5.19	5.62	6.09	6.93	5.27
bonn 87	6.51	**6.43**	6.45	6.60	7.00	8.01	14.14	15.53	14.52	16.25	7.50
bonn 88	4.74	4.68	4.45	4.15	**3.99**	4.04	4.41	16.00	18.48	23.83	4.56
bonn 89	3.20	3.12	2.95	2.80	**2.70**	2.78	3.07	3.54	4.24	5.60	3.09
bonn 90	3.12	3.13	**3.10**	3.12	3.23	3.48	4.10	4.99	6.42	8.89	3.35
bonn 91	3.20	3.20	3.16	**3.15**	3.16	3.17	3.26	3.37	3.52	3.81	3.20
bonn 92	3.43	3.36	3.18	3.10	2.94	**2.93**	3.26	3.55	4.35	8.52	3.31
bonn 93	3.92	3.89	**3.81**	3.82	3.91	3.99	3.98	4.10	4.33	4.52	3.95
bonn 94	4.71	4.67	4.68	4.63	4.58	**4.50**	4.79	8.14	8.46	8.98	4.69
bonn 95	10.82	10.73	**10.60**	10.69	10.89	10.75	11.11	11.67	12.52	13.44	10.86
bonn 96	3.34	3.30	**3.25**	3.26	3.30	3.39	3.63	5.68	4.60	15.23	3.37
bonn 97	2.64	2.59	2.47	2.45	2.33	**2.29**	2.33	2.51	3.17	5.07	2.49
bonn 98	2.98	2.95	**2.92**	2.95	3.22	3.75	3.86	3.80	4.33	6.00	3.49
bonn 99	2.99	2.97	2.85	2.75	2.69	2.68	**2.66**	2.84	3.03	4.12	2.84
Median	3.92	3.83	3.67	3.63	3.77	3.75	3.91	4.52	5.37	6.93	3.92

Table B.14. Summary of segmentation error rates for modified fuzzy K-Means, DT-CWT depth 3. The transform uses the Kingsbury filter pair, at a depth of 3 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	**0.00**	0.00	0.00	0.00	0.04	0.10	0.12	0.13	0.15	0.15	0.07
D5-D92	**0.22**	0.22	0.22	0.28	0.31	0.37	0.41	0.43	0.49	0.52	0.34
D8-D84	0.12	0.13	0.15	0.23	0.26	0.29	0.27	0.26	0.21	**0.10**	0.22
D12-D17	0.88	0.88	0.88	0.90	0.90	0.90	0.88	0.84	0.73	**0.67**	0.88
My 5a	14.34	14.28	13.81	13.57	13.24	12.83	12.51	**12.41**	12.64	13.65	13.41
My 5b	3.23	3.09	2.87	2.66	2.31	1.96	1.74	1.53	1.38	**1.26**	2.13
Nat 10	2.61	2.60	2.58	2.48	2.35	2.23	**2.21**	2.35	2.71	3.50	2.53
Nat 10v	1.50	1.45	1.42	1.41	**1.39**	1.43	1.51	1.55	1.68	1.89	1.47
Nat 16b	5.55	**5.52**	5.57	5.87	6.65	7.99	9.95	12.01	14.10	31.03	7.32
Nat 5b	2.78	2.72	2.64	2.50	**2.36**	2.39	2.97	4.57	8.68	20.75	2.75
Nat 5c	3.06	2.95	2.75	**2.62**	2.63	2.95	3.61	4.55	6.04	8.68	3.01
Nat 5m	**3.11**	3.23	3.41	3.82	4.58	6.21	9.71	17.71	23.43	27.75	5.40
Nat 5v2	**5.92**	5.92	6.21	6.51	8.17	13.60	36.52	37.51	41.64	41.33	10.89
Nat 5v3	**6.13**	6.26	6.70	7.55	12.50	18.34	26.03	31.21	31.63	36.18	15.42
Nat 5v	3.25	3.22	**3.12**	3.13	3.78	5.46	8.89	14.01	14.64	25.16	4.62
bonn 00	2.95	2.93	2.84	2.69	2.55	2.51	2.47	**2.45**	2.45	2.95	2.62
bonn 01	4.57	4.47	**4.45**	4.74	6.20	6.71	7.87	10.56	15.57	20.63	6.45
bonn 02	3.35	3.26	3.06	2.87	2.72	**2.64**	2.71	3.33	5.01	6.78	3.16
bonn 03	2.96	2.88	2.73	2.66	**2.61**	3.12	15.52	14.97	15.44	16.14	3.04
bonn 04	2.61	2.44	2.25	2.14	2.10	**2.00**	2.03	18.67	17.54	16.58	2.34
bonn 05	3.16	3.07	2.87	**2.68**	2.85	3.47	4.36	5.75	10.55	12.96	3.31
bonn 06	2.85	2.75	2.60	2.45	**2.39**	2.44	2.61	2.97	3.56	4.48	2.68
bonn 07	2.68	2.64	2.54	2.46	2.39	2.36	2.31	**2.26**	2.29	2.40	2.40
bonn 08	**5.28**	5.52	6.40	7.80	9.73	17.52	33.33	35.09	28.75	29.11	13.62
bonn 09	**4.25**	4.68	4.65	4.69	5.08	5.47	5.96	10.53	12.27	14.31	5.27
bonn 10	20.96	19.95	10.49	8.31	**8.15**	9.89	11.38	13.09	15.15	16.41	12.24
bonn 11	3.45	3.39	3.31	3.16	**3.12**	3.33	3.64	3.56	3.49	3.41	3.40
bonn 12	5.29	5.21	5.01	**4.92**	5.54	6.95	8.62	10.34	11.95	13.10	6.24
bonn 13	2.11	**2.08**	2.08	2.11	2.14	2.30	2.55	3.14	4.83	21.51	2.22
bonn 14	2.94	2.86	2.69	2.55	2.32	2.27	**2.25**	2.33	2.57	10.88	2.56
bonn 15	2.37	2.29	2.18	**2.14**	2.21	2.34	2.48	2.63	2.95	4.02	2.36
bonn 16	2.57	2.51	2.48	2.39	2.34	**2.30**	2.31	2.44	2.57	2.49	2.46
bonn 17	3.83	3.81	**3.72**	3.77	3.96	4.16	4.49	5.31	6.56	8.69	4.06
bonn 18	**4.72**	4.78	5.02	10.97	13.38	17.72	19.38	21.44	21.92	23.04	15.55
bonn 19	3.05	2.97	2.76	2.48	2.31	**2.19**	2.24	2.25	2.62	13.49	2.55
bonn 20	3.05	3.05	3.06	3.05	**3.04**	3.14	3.31	3.61	4.00	4.47	3.10

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 21	3.71	3.67	3.52	3.47	3.39	**3.34**	3.35	3.53	3.83	4.27	3.52
bonn 22	8.70	8.70	**8.40**	8.41	9.10	10.47	12.74	16.33	23.30	27.51	9.79
bonn 23	**4.00**	4.05	4.11	4.11	4.18	4.27	4.38	4.38	4.64	5.25	4.23
bonn 24	3.25	3.20	3.12	**3.05**	3.09	3.35	3.87	4.24	4.76	5.26	3.30
bonn 25	3.89	3.87	**3.80**	3.89	4.06	4.57	5.29	6.33	7.88	9.48	4.32
bonn 26	2.77	2.71	2.61	2.59	2.54	2.46	2.40	**2.37**	2.37	2.38	2.50
bonn 27	4.39	4.38	4.20	4.09	4.04	3.95	**3.92**	4.14	4.80	6.00	4.17
bonn 28	17.68	17.18	16.36	16.13	15.30	**14.74**	14.94	15.31	16.64	18.29	16.24
bonn 29	2.86	2.81	2.73	2.65	2.43	2.37	2.32	**2.28**	2.47	3.13	2.56
bonn 30	2.98	2.95	2.91	2.95	2.94	2.94	2.89	**2.81**	3.11	4.55	2.94
bonn 31	4.47	4.38	4.28	4.14	**4.00**	4.02	4.16	5.29	14.47	16.85	4.33
bonn 32	3.92	3.86	3.58	3.41	3.10	**2.86**	3.39	4.53	10.02	18.79	3.72
bonn 33	3.46	3.39	3.22	3.08	2.94	2.83	2.75	**2.72**	2.90	3.04	2.99
bonn 34	6.04	5.99	5.89	**5.83**	6.34	8.37	10.66	12.12	14.21	16.68	7.35
bonn 35	2.72	2.67	2.67	2.43	2.26	2.20	2.09	**2.00**	2.08	13.38	2.35
bonn 36	4.61	4.55	4.41	4.31	4.24	4.08	**3.77**	14.68	15.85	16.60	4.48
bonn 37	1.96	1.91	1.85	1.92	1.93	1.90	1.89	**1.81**	1.84	2.11	1.90
bonn 38	5.34	5.26	5.10	**4.93**	5.15	5.63	5.98	6.28	7.59	9.77	5.48
bonn 39	2.47	2.43	2.32	2.25	2.19	2.12	2.09	1.90	**1.84**	2.09	2.16
bonn 40	2.89	2.85	2.72	2.55	2.38	2.25	2.21	**2.16**	2.19	2.26	2.32
bonn 41	**6.40**	6.43	6.77	7.22	11.22	20.15	23.85	21.25	22.85	24.64	15.69
bonn 42	2.76	2.72	2.69	**2.67**	2.72	2.98	4.00	5.31	6.52	7.59	2.87
bonn 43	2.86	**2.84**	2.89	3.00	5.33	9.69	12.61	14.28	15.31	16.56	7.51
bonn 44	3.55	3.44	3.33	**3.18**	3.19	3.20	3.22	3.47	3.98	4.82	3.38
bonn 45	10.20	10.06	9.77	**9.39**	10.38	13.15	15.65	21.58	22.23	22.72	11.76
bonn 46	6.49	6.67	6.69	7.13	23.32	**5.81**	26.51	25.88	25.21	24.22	15.23
bonn 47	3.99	3.91	3.86	3.73	**3.70**	3.91	5.63	5.64	5.78	6.21	3.95
bonn 48	2.64	2.58	2.39	2.32	2.19	**2.17**	2.19	2.20	2.42	14.24	2.36
bonn 49	3.59	3.55	3.43	**3.31**	3.35	4.50	11.10	13.04	14.75	15.87	4.05
bonn 50	4.82	4.77	4.58	4.44	4.27	**4.17**	4.24	4.64	7.23	13.36	4.61
bonn 51	2.17	2.13	2.06	**1.99**	2.00	2.08	2.17	2.46	2.36	2.25	2.15
bonn 52	3.43	3.34	3.13	2.95	2.83	2.79	**2.78**	2.99	4.68	7.40	3.06
bonn 53	4.75	4.58	4.20	3.80	3.30	**3.07**	18.33	17.73	16.31	15.72	4.66
bonn 54	3.57	3.55	3.47	3.44	3.36	**3.33**	3.49	3.68	3.94	4.42	3.52
bonn 55	3.17	3.17	3.22	3.16	3.06	3.00	2.96	**2.85**	2.96	3.38	3.11
bonn 56	5.72	5.68	**5.60**	5.68	5.90	6.35	7.59	11.63	17.16	17.97	6.12
bonn 57	4.35	4.24	4.10	**4.02**	4.03	4.06	4.17	4.46	5.63	8.47	4.21
bonn 58	3.93	3.89	3.79	**3.75**	4.10	4.03	5.08	13.51	17.80	18.04	4.07
bonn 59	3.12	3.08	2.94	2.84	2.71	**2.58**	2.67	3.03	3.65	5.10	2.99
bonn 60	3.68	3.57	3.36	3.15	**2.96**	3.14	4.82	16.15	19.27	17.37	3.63
bonn 61	3.22	3.13	2.98	2.79	2.67	2.58	2.48	2.40	**2.31**	2.35	2.62
bonn 62	**3.72**	3.81	3.88	4.35	15.58	17.28	20.58	21.56	20.78	23.73	16.43
bonn 63	4.38	4.37	**4.33**	4.42	4.80	6.15	6.82	7.68	8.69	10.09	5.47
bonn 64	5.91	5.80	5.54	5.33	5.02	4.75	4.42	4.22	3.96	**3.95**	4.88
bonn 65	2.46	2.40	2.27	2.17	2.03	1.91	**1.89**	1.92	1.89	2.06	2.04
bonn 66	2.19	2.17	2.12	2.08	1.95	1.88	1.78	1.74	**1.70**	14.11	2.01
bonn 67	4.20	4.09	3.93	3.80	**3.63**	3.66	3.83	4.04	4.58	5.55	3.99
bonn 68	3.50	3.39	3.20	3.08	**2.98**	3.16	18.50	18.57	18.69	18.78	3.45
bonn 69	3.98	3.89	3.83	**3.78**	3.79	4.36	5.83	7.71	9.89	12.37	4.17
bonn 70	2.84	2.77	2.69	2.58	2.50	**2.49**	2.67	2.92	3.09	4.13	2.73
bonn 71	3.68	3.61	3.42	3.27	3.08	3.09	**3.06**	3.33	3.50	3.83	3.38
bonn 72	3.39	3.32	3.19	3.02	2.70	2.86	2.51	2.42	**2.42**	2.44	2.78
bonn 73	**5.19**	5.45	6.91	7.97	11.15	16.56	18.34	20.43	21.02	22.56	13.86
bonn 74	**2.17**	2.20	2.23	2.44	9.88	13.71	15.75	17.36	19.67	22.38	11.79
bonn 75	2.83	2.73	2.52	2.34	**2.22**	2.22	2.27	2.47	3.03	5.44	2.50
bonn 76	2.63	2.66	2.62	**2.62**	2.78	2.86	2.89	6.42	7.40	6.85	2.82
bonn 77	3.43	3.42	3.32	3.26	3.11	3.00	**2.87**	2.91	3.25	5.55	3.26
bonn 78	5.04	4.78	4.42	4.25	4.30	**4.16**	14.73	14.20	14.06	13.98	4.91
bonn 79	3.47	3.36	3.22	3.16	**3.11**	3.34	3.92	5.05	7.12	7.26	3.41
bonn 80	2.56	2.55	2.46	2.29	2.22	2.13	**2.06**	2.15	2.21	2.69	2.25
bonn 81	2.97	2.97	**2.94**	2.98	3.20	3.64	4.36	5.59	9.41	19.46	3.42
bonn 82	3.13	3.08	2.88	2.95	**2.68**	9.70	5.87	21.30	20.18	19.45	4.50
bonn 83	3.63	3.53	3.31	3.10	2.81	**2.60**	2.65	3.03	3.52	4.66	3.21
bonn 84	3.32	3.25	3.06	2.85	**2.79**	2.89	4.16	6.82	6.60	6.62	3.28
bonn 85	3.85	3.73	3.56	**3.50**	3.62	4.64	13.76	16.71	19.56	22.14	4.25
bonn 86	5.27	5.15	4.88	4.66	4.42	4.32	**4.24**	4.44	5.32	6.13	4.77
bonn 87	3.96	3.88	3.70	3.57	**3.33**	4.64	6.76	14.07	14.18	14.67	4.30
bonn 88	4.67	4.61	4.40	4.21	3.99	**3.88**	4.46	18.85	22.13	21.85	4.53
bonn 89	3.60	3.50	3.32	3.15	**3.11**	3.24	3.31	3.79	4.46	5.80	3.41
bonn 90	**2.91**	2.91	2.92	2.94	3.14	3.62	4.68	6.29	9.05	13.71	3.38
bonn 91	3.55	3.50	3.47	3.41	3.41	3.42	**3.39**	3.45	3.51	3.94	3.46
bonn 92	3.03	2.96	2.74	2.60	2.42	**2.26**	2.47	3.47	9.09	9.05	2.85
bonn 93	3.22	3.21	3.17	3.14	3.08	**3.06**	3.14	3.18	3.18	3.96	3.17
bonn 94	4.02	3.97	3.89	3.82	**3.79**	5.39	6.85	6.79	7.01	7.46	4.71
bonn 95	9.09	8.94	8.95	9.17	**8.28**	8.61	9.74	11.36	12.41	13.73	9.13
bonn 96	3.10	3.04	2.92	2.80	2.80	**2.77**	2.89	14.21	3.31	13.38	2.98
bonn 97	2.15	2.11	2.06	2.00	1.89	**1.85**	2.02	2.57	2.73	2.84	2.08
bonn 98	2.85	2.87	2.85	2.94	3.17	3.31	2.87	**2.81**	4.00	11.23	2.91
bonn 99	2.92	2.89	2.78	2.70	2.58	2.45	**2.36**	2.42	2.61	3.34	2.66
Median	3.43	3.34	3.22	3.13	3.11	3.24	3.77	4.46	5.63	8.69	3.40

Table B.15. Summary of segmentation error rates for modified fuzzy K-Means, DWT depth 2. The transform uses the Daubechies 9-7 filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	**0.00**	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.00
D5-D92	**0.00**	0.00	0.00	0.00	0.00	0.02	0.05	0.11	0.16	0.17	0.01
D8-D84	0.04	0.01	**0.00**	0.00	0.00	0.00	0.02	0.12	0.13	0.17	0.02
D12-D17	**0.29**	0.33	0.35	0.37	0.40	0.39	0.39	0.46	0.46	0.44	0.39
My 5a	13.88	13.72	13.60	13.49	**13.34**	13.77	14.94	16.06	16.24	16.13	13.82
My 5b	2.51	2.48	2.37	2.28	2.23	**2.17**	2.17	2.21	2.26	2.29	2.27
Nat 10	2.92	2.85	2.71	2.65	2.59	2.56	2.56	2.56	2.57	**2.54**	2.58
Nat 10v	1.58	1.56	1.54	1.51	1.51	1.47	1.41	1.39	**1.36**	1.39	1.49
Nat 16b	7.79	**7.73**	7.77	9.74	10.23	12.01	16.21	19.22	21.20	22.92	11.12
Nat 5b	4.00	3.92	3.79	**3.72**	3.92	4.64	7.32	7.79	34.34	42.35	4.32
Nat 5c	5.29	5.18	5.02	4.82	**4.75**	4.91	5.19	6.30	8.33	13.38	5.18
Nat 5m	4.55	4.40	**4.33**	4.63	5.83	6.99	8.83	11.22	16.59	21.90	6.41
Nat 5v2	13.54	13.18	12.44	11.51	10.54	**10.35**	15.19	13.99	34.83	35.46	13.36
Nat 5v3	5.68	**5.62**	5.73	6.07	6.61	8.26	11.81	16.20	26.29	22.77	7.43
Nat 5v	4.44	4.40	4.23	4.13	**3.92**	4.71	4.71	7.29	8.95	13.11	4.57
bonn 00	3.38	3.31	3.12	2.97	2.72	2.61	2.42	2.36	**2.33**	2.40	2.66
bonn 01	3.91	3.82	3.75	3.42	**3.11**	3.18	3.92	5.99	9.43	11.53	3.86
bonn 02	3.39	3.28	3.08	2.91	2.81	2.81	**2.76**	2.80	2.91	3.38	2.91
bonn 03	3.34	3.27	3.20	**3.09**	3.09	3.23	3.51	4.42	14.62	21.86	3.31
bonn 04	2.85	2.86	2.61	2.40	2.18	2.02	**1.97**	2.33	3.00	14.29	2.51
bonn 05	4.77	4.60	4.27	4.03	**3.91**	3.99	10.86	13.02	15.25	15.67	4.68
bonn 06	2.75	2.61	2.42	2.29	**2.22**	2.39	2.71	3.31	4.33	5.71	2.66
bonn 07	3.63	3.48	3.29	3.05	2.85	**2.73**	2.78	2.82	3.06	3.70	3.06
bonn 08	5.49	5.32	5.15	**4.93**	5.27	6.80	14.53	36.89	34.57	27.39	6.14
bonn 09	3.94	**3.93**	3.96	3.96	3.96	3.96	4.22	4.65	6.37	7.57	3.96
bonn 10	25.01	5.05	4.91	4.16	**4.05**	4.62	4.10	4.43	4.90	6.33	4.76
bonn 11	2.17	2.12	1.98	1.91	**1.90**	1.98	2.11	2.70	3.49	5.11	2.12
bonn 12	4.68	4.41	4.12	**4.03**	4.05	4.13	5.13	6.45	8.87	11.44	4.54
bonn 13	2.28	2.23	**2.19**	4.80	2.48	3.03	15.59	4.34	9.17	26.32	3.68
bonn 14	3.45	3.42	3.31	3.14	2.97	**2.89**	2.89	2.93	3.29	4.46	3.22
bonn 15	2.29	2.22	2.02	1.94	1.82	1.79	**1.74**	1.86	2.08	3.04	1.98
bonn 16	2.25	2.19	2.10	2.03	1.87	1.77	1.73	**1.67**	1.67	1.75	1.82
bonn 17	2.87	**2.76**	2.77	2.87	2.86	2.94	3.17	3.57	3.86	4.17	2.91
bonn 18	4.58	4.48	4.23	**4.13**	4.53	5.82	19.71	20.34	22.00	23.38	5.20
bonn 19	2.86	2.74	2.38	**2.15**	2.29	2.74	4.35	2.96	3.86	6.84	2.80
bonn 20	2.70	2.69	2.48	2.44	2.40	2.35	**2.26**	2.34	3.26	2.76	2.46
bonn 21	3.40	3.38	**3.17**	3.27	3.34	3.35	3.39	3.54	3.86	4.54	3.38
bonn 22	11.85	11.49	10.85	10.37	**10.12**	10.83	14.39	15.56	29.55	34.70	11.67
bonn 23	4.14	3.89	3.46	3.15	2.70	2.33	2.08	1.94	**1.89**	2.14	2.52
bonn 24	3.12	3.03	2.76	2.62	2.37	2.28	**2.19**	2.24	2.46	3.11	2.54
bonn 25	3.49	3.41	3.35	**3.06**	3.11	3.45	3.83	4.95	9.93	18.68	3.47
bonn 26	1.96	1.95	1.90	1.85	1.75	1.68	**1.66**	1.79	2.25	2.42	1.87
bonn 27	4.11	4.14	4.13	4.03	**3.94**	3.97	4.03	4.04	4.33	4.77	4.07
bonn 28	30.75	30.80	30.66	30.53	30.49	30.19	29.57	29.11	**26.19**	27.86	30.34
bonn 29	3.08	2.98	2.78	2.58	2.40	2.24	**2.23**	2.40	3.09	4.48	2.68
bonn 30	2.59	2.53	2.37	**2.21**	2.22	2.50	2.89	3.58	3.95	4.34	2.56
bonn 31	4.70	4.64	4.51	**4.49**	4.64	5.44	6.49	8.15	10.24	12.72	5.07
bonn 32	4.63	4.58	4.35	4.25	4.07	4.07	**3.95**	4.12	4.25	4.33	4.25
bonn 33	3.24	3.16	3.00	2.85	2.65	2.58	2.57	**2.53**	2.62	2.80	2.73
bonn 34	6.77	6.58	**6.38**	6.55	8.51	11.32	14.89	20.83	18.21	17.47	9.92
bonn 35	3.01	2.93	2.74	2.61	2.50	**2.47**	2.48	2.53	2.77	3.34	2.67
bonn 36	3.78	3.69	3.45	3.19	3.04	**3.01**	3.11	4.33	8.39	31.28	3.57
bonn 37	2.53	2.47	2.37	2.37	**2.27**	2.28	2.38	2.45	2.65	3.53	2.42
bonn 38	4.46	3.94	3.73	3.39	3.15	**3.03**	3.25	4.04	8.53	7.26	3.83
bonn 39	2.34	2.23	2.11	1.92	1.79	**1.70**	1.76	1.89	2.37	4.64	2.02
bonn 40	2.44	2.37	2.28	2.13	2.00	1.87	1.82	1.82	**1.79**	1.95	1.97
bonn 41	6.63	4.22	3.99	**3.98**	4.25	4.54	5.42	6.65	8.21	10.90	4.98
bonn 42	**21.48**	22.05	23.42	24.26	24.84	23.83	23.54	24.06	22.03	23.45	23.50
bonn 43	3.23	3.18	**3.11**	3.16	3.33	4.96	5.65	10.56	11.15	16.57	4.15
bonn 44	4.22	4.11	3.83	3.63	3.40	3.31	**3.15**	3.50	4.47	6.38	3.73
bonn 45	9.79	9.66	9.43	8.92	8.39	**8.22**	8.49	10.69	13.60	16.94	9.54
bonn 46	6.02	5.91	**5.58**	7.48	26.45	26.50	7.27	26.49	25.82	24.60	16.04
bonn 47	4.65	4.45	4.25	4.21	4.13	**4.09**	4.27	4.58	5.79	8.04	4.36
bonn 48	2.91	2.84	2.61	2.49	**2.36**	2.55	10.85	10.84	17.79	20.64	2.87
bonn 49	2.95	2.94	2.87	**2.76**	2.94	3.05	3.27	3.70	5.13	17.88	3.00
bonn 50	3.97	3.86	3.56	3.36	3.10	2.89	**2.71**	2.78	2.92	3.36	3.23
bonn 51	2.62	2.55	2.32	2.22	1.99	**1.89**	1.96	2.01	2.09	2.80	2.15
bonn 52	3.14	3.02	2.75	2.51	2.33	**2.29**	2.38	2.56	2.95	3.87	2.65
bonn 53	4.31	23.83	3.70	3.13	**2.37**	22.97	2.64	3.29	4.41	19.05	4.00
bonn 54	4.04	3.99	3.81	3.74	3.47	3.38	3.30	**3.29**	3.36	3.45	3.46
bonn 55	2.87	2.82	2.71	2.67	2.63	2.57	**2.56**	2.57	2.61	2.72	2.65
bonn 56	**7.23**	7.69	8.31	7.84	7.53	10.57	15.94	23.38	28.12	27.95	9.44
bonn 57	4.93	4.85	**4.70**	4.75	5.15	5.69	5.79	6.62	7.71	23.58	5.42
bonn 58	4.05	3.96	3.72	3.63	**3.58**	3.64	4.61	6.05	9.69	12.41	4.00
bonn 59	3.69	3.61	3.42	3.30	**3.28**	3.34	3.50	3.76	4.56	6.72	3.56
bonn 60	2.85	2.89	**2.73**	2.85	2.77	2.73	3.14	5.17	10.16	15.06	2.87
bonn 61	2.83	2.76	2.66	2.55	2.53	2.48	**2.48**	2.50	2.67	5.18	2.61
bonn 62	3.36	3.34	**3.17**	3.19	3.35	4.23	7.32	24.60	26.86	27.75	3.80
bonn 63	4.75	**4.72**	4.75	4.83	4.90	5.16	5.32	6.13	7.13	10.23	5.03
bonn 64	6.37	6.30	6.04	5.96	5.77	5.62	5.29	**5.19**	5.19	5.31	5.69
bonn 65	3.70	3.61	3.49	3.62	3.52	**3.38**	3.69	4.66	5.90	6.85	3.65

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 66	2.00	1.97	1.89	1.82	1.73	1.70	**1.68**	1.80	2.27	3.14	1.86
bonn 67	3.56	3.44	3.30	3.28	3.27	3.25	**3.21**	3.39	3.67	4.96	3.34
bonn 68	4.78	4.61	4.24	3.92	3.64	3.56	**3.55**	3.60	3.69	4.02	3.80
bonn 69	4.36	**4.35**	4.41	4.97	5.76	6.98	11.23	14.23	18.59	20.94	6.37
bonn 70	2.17	2.09	1.95	1.85	1.73	**1.68**	1.77	1.88	2.00	2.55	1.92
bonn 71	3.41	3.32	3.12	3.01	2.86	**2.76**	2.88	3.27	4.32	6.20	3.19
bonn 72	4.11	4.07	4.05	3.94	**3.80**	3.91	4.16	4.79	5.81	7.34	4.09
bonn 73	4.67	4.69	4.41	4.11	**3.88**	4.68	10.33	24.57	19.71	24.21	4.68
bonn 74	2.52	2.46	2.37	**2.30**	2.33	2.56	4.37	9.85	12.60	15.89	2.54
bonn 75	3.44	3.35	3.17	3.09	**3.02**	3.31	3.18	3.72	5.65	8.95	3.33
bonn 76	3.80	3.64	3.38	**3.27**	3.39	3.61	4.05	5.04	6.32	8.47	3.72
bonn 77	**3.43**	3.46	3.57	3.74	4.02	4.10	4.50	4.84	5.32	6.15	4.06
bonn 78	3.55	3.47	3.45	3.27	3.17	**3.16**	3.29	4.91	18.54	17.08	3.46
bonn 79	4.11	4.03	3.87	**3.78**	4.00	4.44	5.05	5.73	6.82	9.13	4.28
bonn 80	2.14	2.08	1.97	1.85	1.76	**1.72**	1.73	1.94	2.45	3.12	1.95
bonn 81	23.82	23.71	**3.30**	3.33	3.85	6.42	3.80	6.59	20.50	16.65	6.50
bonn 82	2.61	2.54	2.37	2.27	2.19	2.18	**2.08**	2.40	3.23	4.31	2.39
bonn 83	3.80	3.71	3.69	3.38	2.94	2.87	**2.72**	2.85	3.32	4.11	3.35
bonn 84	3.13	2.99	2.81	2.50	2.31	2.25	**2.24**	2.55	3.49	5.07	2.68
bonn 85	4.33	4.19	3.86	3.72	3.24	3.10	3.06	**3.00**	4.57	12.95	3.79
bonn 86	3.74	3.60	3.31	3.06	2.82	**2.79**	3.06	4.11	3.89	5.35	3.45
bonn 87	6.04	5.97	5.84	5.77	**5.68**	5.91	7.18	10.28	12.73	12.57	6.00
bonn 88	3.45	3.39	3.19	2.97	2.80	2.61	**2.50**	2.95	3.99	16.15	3.08
bonn 89	3.08	2.93	2.78	2.61	2.38	**2.36**	2.40	2.70	3.19	4.50	2.74
bonn 90	2.72	2.67	2.58	**2.56**	2.58	2.70	3.21	4.46	6.85	10.91	2.71
bonn 91	3.27	3.20	3.02	**2.94**	3.00	3.12	3.54	3.74	4.13	4.88	3.24
bonn 92	2.80	2.72	2.49	**2.40**	2.45	2.52	2.69	3.00	3.64	6.45	2.70
bonn 93	4.71	4.61	4.36	4.19	**4.06**	4.13	4.25	4.50	4.86	5.37	4.43
bonn 94	2.93	2.90	2.80	**2.75**	2.79	2.83	3.08	3.55	4.33	8.48	2.91
bonn 95	6.32	**6.21**	6.39	7.31	9.37	10.38	11.07	12.16	10.88	11.23	9.87
bonn 96	3.30	3.24	3.08	**3.04**	3.19	3.42	20.73	5.88	7.69	12.05	3.36
bonn 97	2.05	1.97	1.88	1.76	1.68	1.63	**1.61**	1.68	1.77	1.94	1.76
bonn 98	2.65	2.50	2.41	2.34	2.30	**2.29**	2.63	3.28	4.47	5.51	2.57
bonn 99	2.61	2.62	2.56	2.52	2.39	**2.37**	2.38	2.39	2.47	2.98	2.50
Median	3.45	3.42	3.30	3.15	3.04	3.10	3.27	3.74	4.47	6.72	3.36

Table B.16. Summary of segmentation error rates for modified fuzzy K-Means, DT-CWT depth 2. The transform uses the Kingsbury filter pair, at a depth of 2 levels.

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
D4-D84	**0.06**	0.07	0.09	0.07	0.07	0.06	0.09	0.12	0.16	0.16	0.08
D5-D92	0.27	**0.21**	0.21	0.26	0.32	0.33	0.39	0.41	0.49	0.49	0.32
D8-D84	**0.09**	0.10	0.10	0.17	0.29	0.27	0.31	0.28	0.27	0.20	0.23
D12-D17	0.89	0.89	0.89	0.90	0.92	0.88	0.85	0.84	**0.81**	0.84	0.89
My 5a	9.84	**9.72**	12.05	13.64	14.42	15.09	15.83	15.13	15.00	15.43	14.71
My 5b	2.64	2.56	2.44	2.28	2.09	1.93	1.74	1.56	1.48	**1.43**	2.01
Nat 10	3.01	2.98	2.87	2.71	2.52	**2.49**	2.51	2.51	2.51	2.63	2.57
Nat 10v	1.40	1.37	1.31	1.33	1.34	1.32	1.28	1.21	**1.19**	1.26	1.31
Nat 16b	6.14	**6.07**	6.11	6.30	7.20	9.29	12.65	16.47	19.20	21.26	8.24
Nat 5b	2.94	**2.84**	2.85	2.89	3.01	3.36	4.24	9.43	26.69	37.54	3.18
Nat 5c	3.75	3.60	3.47	3.36	**3.25**	3.26	3.39	3.99	5.20	7.71	3.54
Nat 5m	3.77	3.65	3.55	**3.43**	3.71	4.49	6.99	11.54	17.05	22.45	4.13
Nat 5v2	7.09	6.95	6.64	**6.45**	6.46	7.29	9.41	25.22	36.14	39.62	7.19
Nat 5v3	**5.85**	5.96	5.98	6.16	6.95	9.18	13.87	15.90	19.23	28.02	8.06
Nat 5v	3.18	3.12	**3.06**	3.10	3.17	3.44	4.52	6.85	10.07	14.97	3.31
bonn 00	2.96	2.84	2.70	2.54	2.33	2.26	2.28	2.27	**2.21**	2.28	2.30
bonn 01	3.97	3.82	3.53	3.05	**2.75**	3.48	4.90	6.21	8.64	15.42	3.90
bonn 02	3.27	3.17	3.00	2.73	2.56	2.34	2.27	**2.25**	2.46	3.14	2.65
bonn 03	2.45	2.43	2.33	2.19	2.09	**2.08**	2.42	13.14	19.06	19.83	2.43
bonn 04	2.58	2.50	2.36	2.15	2.00	1.78	**1.69**	1.69	14.49	12.95	2.26
bonn 05	3.63	3.60	3.42	**3.31**	3.34	3.64	7.01	12.47	14.18	15.07	3.64
bonn 06	2.42	2.33	2.12	2.06	**2.03**	2.04	2.19	2.64	3.56	5.01	2.26
bonn 07	3.07	2.99	2.88	2.75	2.59	2.47	2.42	2.42	**2.34**	2.36	2.53
bonn 08	4.48	**4.47**	4.54	4.74	5.93	7.24	9.50	32.54	27.86	31.88	6.59
bonn 09	3.78	3.78	3.81	3.82	**3.75**	3.83	4.40	5.41	6.63	9.52	3.82
bonn 10	4.24	4.30	3.92	3.64	3.52	**3.48**	3.64	3.91	4.66	6.36	3.92
bonn 11	2.48	2.42	2.32	2.23	**2.15**	2.15	2.23	2.59	3.27	3.47	2.37
bonn 12	5.00	4.82	4.50	4.25	**4.05**	4.21	5.19	6.97	9.27	10.33	4.91
bonn 13	2.01	1.92	1.83	**1.75**	2.20	2.17	2.53	3.47	19.48	23.38	2.19
bonn 14	2.91	2.81	2.62	2.44	2.27	2.25	**2.23**	2.29	2.34	2.70	2.39
bonn 15	1.81	1.75	1.68	1.57	1.33	**1.32**	1.45	1.51	1.70	2.69	1.63
bonn 16	2.27	2.22	2.07	1.95	1.89	1.84	**1.81**	1.84	1.87	1.95	1.92
bonn 17	2.89	2.84	2.79	**2.76**	2.80	2.94	3.28	3.28	3.41	4.28	2.91
bonn 18	4.54	4.47	4.34	**4.13**	4.34	5.61	15.33	18.21	20.43	22.61	5.08
bonn 19	2.81	2.65	2.50	2.50	2.33	2.25	**2.19**	2.38	3.16	19.57	2.50
bonn 20	3.00	2.94	2.82	2.62	**2.56**	2.65	2.58	2.64	3.17	4.27	2.73
bonn 21	3.69	3.63	3.47	3.23	3.12	**3.00**	3.03	3.07	3.21	3.71	3.22
bonn 22	8.84	8.60	8.16	**7.93**	8.36	9.27	10.79	13.69	30.10	31.59	9.05
bonn 23	3.89	3.75	3.37	3.02	2.67	2.41	2.21	**2.03**	2.48	3.03	2.84
bonn 24	2.92	2.86	2.67	2.47	2.27	2.17	**2.13**	2.25	3.03	4.11	2.57
bonn 25	**3.34**	3.34	3.36	3.44	3.83	4.81	5.62	5.72	5.55	11.03	4.32

Image \ β	0	1	2	3	5	8	13	21	34	55	Median
bonn 26	2.16	2.13	2.09	1.97	1.87	1.78	1.76	1.76	**1.75**	1.87	1.87
bonn 27	4.14	4.16	4.11	3.99	3.90	3.87	3.75	**3.63**	3.80	4.70	3.95
bonn 28	23.85	24.99	25.17	**4.88**	22.52	22.27	21.75	22.78	23.57	20.31	22.65
bonn 29	2.90	2.83	2.66	2.50	2.22	2.01	**1.85**	1.94	2.36	3.41	2.43
bonn 30	2.35	2.26	2.16	2.05	**2.01**	2.07	2.34	2.42	2.50	2.65	2.30
bonn 31	3.44	3.32	3.22	**3.10**	3.22	3.56	4.14	12.37	6.99	17.66	3.50
bonn 32	4.85	4.72	4.55	4.35	4.26	3.98	3.55	**3.47**	3.86	5.98	4.30
bonn 33	2.75	2.71	2.58	2.49	2.32	2.20	**2.16**	2.17	2.20	2.39	2.35
bonn 34	5.21	5.08	**4.90**	4.98	5.63	5.12	13.85	14.75	14.81	15.71	5.42
bonn 35	2.79	2.65	2.47	2.37	2.27	2.20	2.11	**2.00**	2.08	2.43	2.32
bonn 36	2.93	2.89	2.83	2.69	2.65	2.65	**2.56**	3.22	7.06	11.93	2.86
bonn 37	2.17	2.11	1.94	1.93	1.85	**1.81**	1.81	1.84	1.96	2.24	1.94
bonn 38	3.83	3.74	3.64	3.54	**3.34**	3.51	3.75	4.59	6.55	10.93	3.74
bonn 39	2.00	1.93	1.80	1.67	1.53	1.51	**1.49**	1.57	2.13	2.61	1.74
bonn 40	2.36	2.32	2.25	2.15	2.05	1.93	1.92	**1.90**	1.93	1.95	2.00
bonn 41	5.43	**5.33**	5.35	5.51	5.48	6.10	7.05	9.09	17.44	20.09	5.81
bonn 42	22.29	22.80	23.25	3.00	**2.16**	2.35	9.13	15.47	18.05	18.57	16.76
bonn 43	2.35	2.28	2.22	**2.17**	2.37	3.36	9.25	11.88	14.14	15.41	2.87
bonn 44	3.55	3.45	3.28	3.01	2.76	2.58	2.47	**2.43**	3.33	5.57	3.15
bonn 45	8.54	8.28	7.88	7.53	7.06	**6.83**	8.72	12.81	16.32	20.81	8.41
bonn 46	5.53	5.40	**5.32**	7.89	11.01	9.27	23.93	19.73	19.73	23.39	10.14
bonn 47	2.61	2.52	2.34	**2.20**	2.26	2.57	2.86	3.18	4.39	7.29	2.59
bonn 48	2.60	2.48	2.34	2.18	**2.03**	2.06	11.35	14.16	2.18	10.36	2.41
bonn 49	3.05	3.00	2.91	2.80	**2.62**	2.62	2.84	4.13	14.57	15.47	2.95
bonn 50	3.34	3.33	3.23	3.14	3.06	**2.95**	2.96	3.09	3.45	4.14	3.19
bonn 51	2.54	2.48	2.33	2.17	1.98	1.92	**1.88**	1.88	2.09	2.97	2.13
bonn 52	2.91	2.81	2.61	2.40	2.24	**2.13**	2.15	2.23	2.47	3.09	2.44
bonn 53	4.57	4.32	3.83	3.43	22.52	22.06	**2.39**	3.26	19.40	18.30	4.44
bonn 54	3.25	3.19	3.13	3.05	2.94	2.83	2.83	**2.78**	2.79	3.39	2.99
bonn 55	2.49	2.44	2.47	2.48	2.47	2.44	2.38	2.36	**2.32**	2.81	2.45
bonn 56	5.49	5.38	5.05	**4.79**	4.85	5.25	15.52	20.64	21.31	19.80	5.44
bonn 57	3.55	3.51	3.31	**3.30**	3.44	3.58	3.82	4.28	5.49	21.61	3.56
bonn 58	3.38	3.36	**3.34**	3.40	3.73	4.30	3.90	5.05	10.96	12.69	3.81
bonn 59	2.94	2.87	2.77	2.67	**2.57**	2.57	2.60	2.67	3.36	4.75	2.72
bonn 60	2.68	2.64	2.51	2.42	**2.40**	2.53	3.78	14.69	17.29	19.14	2.66
bonn 61	2.89	2.81	2.64	2.55	2.36	2.28	2.25	**2.22**	2.27	2.34	2.35
bonn 62	3.55	3.45	**3.24**	3.25	3.52	4.67	8.07	20.15	22.16	23.94	4.11
bonn 63	3.58	3.59	**3.51**	3.53	3.54	3.62	3.74	4.10	4.85	6.95	3.60
bonn 64	6.19	6.12	5.91	5.66	5.26	4.90	4.63	4.39	4.24	**4.11**	5.08
bonn 65	2.71	2.63	2.58	2.33	2.05	1.89	**1.78**	1.84	2.12	2.76	2.22
bonn 66	1.85	1.81	1.75	1.69	1.61	1.52	**1.50**	1.57	1.61	6.89	1.65
bonn 67	3.45	3.42	3.28	3.16	3.07	**3.05**	3.16	3.33	3.67	4.32	3.30
bonn 68	3.99	3.88	3.66	3.48	3.27	3.21	**3.11**	3.11	3.41	9.22	3.44
bonn 69	4.04	3.93	3.81	**3.77**	4.22	5.85	8.12	11.96	17.04	21.00	5.04
bonn 70	1.84	1.86	1.84	1.79	1.70	**1.65**	1.67	1.82	2.18	2.56	1.83
bonn 71	3.22	3.11	2.94	2.75	2.57	2.30	**2.23**	2.28	2.51	4.17	2.66
bonn 72	3.65	3.60	3.42	3.29	3.17	**3.09**	3.23	3.61	4.01	4.29	3.51
bonn 73	4.53	4.50	4.42	4.14	**3.77**	5.70	9.99	14.03	17.51	20.39	5.12
bonn 74	2.32	2.25	2.14	2.11	**2.07**	2.27	4.47	7.98	11.60	15.71	2.29
bonn 75	2.36	2.30	2.14	2.03	**1.95**	2.07	2.12	2.22	2.81	6.63	2.18
bonn 76	2.58	2.54	2.43	2.31	**2.19**	2.28	2.85	3.33	3.84	6.07	2.56
bonn 77	**3.10**	3.16	3.12	3.25	3.38	3.52	3.63	3.78	3.91	4.65	3.45
bonn 78	3.73	3.56	3.33	3.12	2.97	**2.91**	2.94	3.99	6.70	13.76	3.45
bonn 79	3.59	3.50	3.34	3.22	**3.20**	3.47	3.99	4.63	5.63	7.13	3.55
bonn 80	2.10	2.03	1.89	1.78	1.63	1.51	1.40	1.34	**1.27**	1.30	1.57
bonn 81	2.91	2.79	2.60	**2.46**	2.52	2.80	3.81	6.54	15.22	21.02	2.85
bonn 82	2.70	2.65	2.53	2.39	2.19	2.13	**2.02**	2.19	3.02	20.48	2.46
bonn 83	3.15	3.07	2.88	2.70	2.37	2.17	**2.08**	2.14	2.39	3.05	2.55
bonn 84	3.07	2.94	2.73	2.46	2.19	**2.11**	2.17	2.75	4.45	5.16	2.74
bonn 85	3.50	3.41	3.23	3.06	2.80	2.62	2.42	**2.36**	17.62	21.20	3.14
bonn 86	3.41	3.30	3.12	2.87	2.66	**2.58**	2.69	2.82	3.91	6.36	2.99
bonn 87	3.68	3.55	3.50	3.41	3.28	**3.13**	4.28	6.97	12.88	11.39	3.62
bonn 88	3.99	3.96	3.74	3.45	3.19	3.06	**3.01**	3.33	18.37	20.44	3.60
bonn 89	2.97	2.92	2.73	2.59	2.36	2.24	**2.23**	2.40	2.84	3.53	2.66
bonn 90	2.53	2.48	2.42	**2.34**	2.44	2.59	3.10	4.05	7.58	11.54	2.56
bonn 91	3.31	3.19	3.11	3.02	2.89	**2.87**	2.98	3.12	3.41	3.53	3.11
bonn 92	2.70	2.61	2.42	2.27	2.11	2.01	**2.00**	2.22	3.07	7.53	2.35
bonn 93	4.19	4.17	4.00	3.85	3.60	3.48	3.38	3.34	**3.27**	4.45	3.72
bonn 94	2.79	2.78	2.78	2.70	**2.59**	2.64	2.69	4.01	7.26	8.11	2.78
bonn 95	6.07	**6.03**	6.14	7.03	8.76	9.90	10.30	10.98	12.26	12.58	9.33
bonn 96	3.10	3.08	2.94	**2.74**	2.75	2.84	3.41	5.83	15.12	9.68	3.09
bonn 97	1.79	1.75	1.63	1.57	1.46	1.42	**1.41**	1.45	1.52	1.72	1.55
bonn 98	2.51	2.48	2.40	2.36	**2.31**	2.36	2.71	2.86	3.36	3.93	2.50
bonn 99	2.38	2.29	2.15	2.06	2.09	2.10	2.01	1.89	**1.84**	1.97	2.08
Median	3.07	3.00	2.88	2.75	2.59	2.62	2.85	3.26	3.86	6.63	2.86

This page is blank

Appendix C

List of Publications

- Lin, Wen-Kuo, Ng, Brian Wai-him, Burgess, Neil and Bouzerdoum, Abdesselam. Reduced Memory Zerotree Coding Algorithm for Hardware Implementation, IEEE International Conference on Multimedia Computing and System, Florence, Italy, June 1999

- Ng, Brian and Bouzerdoum, Abdesselam. Supervised Texture Segmentation Using DWT and a Modified k-NN Classifier Proceedings of 15th International Conference on Pattern Recognition, September 3-7 Barcelona 2000, Volume 2, pp 545-548 Editors: A. Sanfeliu, J.J. Villanueva, M. Vanrell, R. Alquezar, A.K. Jain, J. Kittler

- Ng, Brian and Bouzerdoum, Abdesselam. Supervised Texture Segmentation Using DT-CWT and a Modified k-NN Classifier Visual Communications and Image Processing 2000, 20-23 June 2000, Perth, Australia. Proceedings of SPIE Vol. 4067 (2000). Part three, pp 1176-1184. Editors: King N. Ngan, Thomas Sikora, Ming-Ting Sun

- Ng, Brian and Bouzerdoum, Abdesselam. K-Means Clustering Applied to Texture Segmentation with Complex Wavelet Features. The 5th International Conference on Optimization: Techniques and Applications. December 15–17, 2001,q Hong Kong

This page is blank

Bibliography

[1] Roberto H. Bamberger, Steven L. Eddins, and Veyis Nuri. Generalized symmetric extensions for size-limited multirate filter banks. *IEEE Transactions on Image Processing*, 3(1):82–87, January 1994.

[2] J. Bigün. Unsupervised feature reduction in image segmentation by local transforms. *Pattern Recognition Letters*, 14:573–583, July 1993.

[3] Christopher M. Brislawn. Preservation of subband symmetry in multirate signal coding. *Signal Processing*, 43(12):1–4, December 1995.

[4] Christopher M. Brislawn. Classification of nonexpansive symmetric extension transforms for multirate filter banks. *Applied and Computational Harmonic Analysis*, 3:337–357, 1996.

[5] P. Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover, 1966.

[6] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[7] A. R. Calderbank, Ingrid Daubechies, Wim Sweldens, and Boon-Lock Yeo. Wavelet transforms that map integers to integers. *Appl. Comput. Harmon. Anal.*, 5(4):332–369, 1998.

[8] Kenneth R. Castleman. *Digital Image Processing*. Prentice Hall, 1996.

[9] Tianhorng Chang and C.-C. Jay Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on Image Processing*, 2(4):429–441, Oct 1993.

[10] B. B. Chaudhuri and Nirupam Sarkar. Texture segmentation using fractal dimension. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):72–77, January 1995.

[11] Patrick C. Chen and Theodosios Pavlidis. Segmentation by texture using a co-occurrence matrix and a split-and-merge algorithm. *Computer Graphics and Image Processing*, 10:172–182, 1979.

[12] Albert Cohen and Ingrid Daubechies. Non-separable bidimensional wavelet bases. *Revista Matemática Iberoamericana*, 9(1):51–137, 1993.

[13] Albert Cohen, Ingrid Daubechies, and Pierre Vial. Wavelets on the interval and fast wavelet transforms. *Applied and Computational Harmonic Analysis*, 1:54–81, 1993.

[14] University of Bonn Computer Vision Group, Computer Science. Texture database. http://www-dbv.informatik.uni-bonn.de/image/segmentation.

[15] Richard W. Conners and Charles A. Harlow. A theoretical comparison of texture algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(3):204–222, May 1980.

[16] George R. Cross and Anil K. Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(1):25–39, Jan 1983.

[17] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, XLI:909–996, 1988.

[18] Ingrid Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, 1992.

[19] Ingrid Daubechies and Wim Sweldens. Factoring wavelet transforms into lifting steps. *J. Fourier Anal. Appl.*, 4(3):245–267, 1998.

[20] Peter de Rivaz and Nick Kingsbury. Complex wavelet features for fast texture image retrieval. In *Proc. IEEE International Conference on Image Processing 99*, 1999.

[21] Ronald A. DeVore. Adaptive wavelet bases for image compression. In P.J. Laurent, A. Le Méhauté, and L. L. Schumaker, editors, *Wavelets, Images and Surface Fitting*. A. K. Peters, 1994.

[22] R. Dubes and A.K. Jain. Random field models in image analysis. *Journal of Applied Statistics*, 6(2):131–164, 1989.

[23] Dennis Dunn, William E. Higgins, and Joseph Wakeley. Texture segmentation using 2-d gabor elementary functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):130–149, February 1994.

[24] Navid Fatemi-Ghomi. *Performance Measures for Wavelet-based Segmentation Algorithms*. Phd thesis, University of Surrey, 1997.

[25] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice-Hall, Inc., Upper Saddle River, New Jersey 07458, 2nd edition edition, 2002.

[26] D.J. Hand. *Discrimination and Classification*. John Wiley & Sons Ltd., 1981.

[27] Robert M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, May 1979.

[28] Thomas Hofmann, Jan Puzicha, and Joachim M. Buhmann. Unsupervised segmentation of textured images by pairwise data clustering. In *Proc. of IEEE International Conference on Image Processing, Lausanne*, volume III, pages 137–140, 1996.

[29] John Y. Hsiao and Alexander A. Sawchuk. Supervised textured image segmentation using feature smoothing and probabilistic relaxation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1279–1292, Dec 1989.

[30] John Y. Hsiao and Alexander A. Sawchuk. Unsupervised textured image segmentation using feature smoothing and probabilistic relaxation techniques. *Computer Vision, Graphics, and Image Processing*, 48(3):1–21, 1989.

[31] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. Technical Report 19, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001.

[32] Anil K. Jain and Farshid Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

[33] Anil K. Jain and Kalle Karu. Learning texture discrimination masks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):195–205, Feb 1996.

[34] Warren J. Jasper, Stephen J. Garnier, and Harsh Potlapalli. Texture characterization and defect detection using adaptive wavelets. *Optical Engineering*, 35(11):3140–3149, Nov 1996.

[35] Alvin Kam and William Fitzgerald. Unsupervised multiscale image segmentation. In *Proc. of the 10th International Conference on Image Analysis and Processing (ICIAP'99)*, pages 315–321, Venice, Italy, September 1999.

[36] Nick Kingsbury. Image processing with complex wavelets. In *Phil. Trans. Royal Society London A, Discussion meeting on "Wavelets: the key to intermittent information?"*, February 1999.

[37] Nick Kingsbury. A dual-tree complex wavelet transform with improved orthogonality and symmetry properties. In *Proc. IEEE Conf. on Image Processing, Vancouver, September 11-13, 2000*, page paper 1429, 2000.

[38] Hitoshi Kiya, Kiyoshi Nishikawa, and Masahiro Iwahashi. A development of symmetric extension method for subband image coding. *IEEE Transactions on Image Processing*, 3(1):78–81, January 1994.

[39] G. Koepfler, C. Lopez, and J.M. Morel. A multiscale algorithm for image segmentation by variational method. *SIAM J. Numer. Anal.*, 31(1):282–299, February 1994.

[40] Erwin Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, Inc., 7th edition, 1993.

[41] Andrew Laine and Jian Fan. An adaptive approach for texture segmentation by multi-channel wavelet frames. Technical Report TR-93-025, Centre for Computer Vision and Visualization, August 1993.

[42] Andrew Laine and Jian Fan. Texture classification by wavelet packet signatures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1186–1191, Nov 1993.

[43] K.I. Laws. Rapid texture identification. *Proc. SPIE*, 238:376–380, 1980.

[44] J. F. A. Magarey and N. G. Kingsbury. Motion estimation using complex wavelets. In *Proc. ICASSP-96, Atlanta*, 1996.

[45] Stephane G. Mallat. A theory of multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, Jul 1989.

[46] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, August 1996.

[47] Yves Meyer. *Wavelets and Operators*. Cambridge University Press, 1992.

[48] Yves Meyer. *Wavelets Algorithms & Applications*. Society for Industrial and Applied Mathematics, 1993.

[49] E. Monjoux and J.P. Rudant. Texture segmentation in aerial images. In *Image Porcessing Algorithms and Techniques II, San Jose, Ca*, SPIE, pages 310–318, 1991.

[50] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. in Pure Appl. Math.*, 42:577–684, 1989.

[51] Philippe P. Ohanian and Richard C. Dubes. Performance evaluation for four classes of textural features. *Pattern Recognition*, 25(8):819–833, 1992.

[52] Jeng-Shyang Pan and Jing-Wein Wang. Texture segmentation using separable and non-separable wavelet frames. *IEICE Trans. Fundamentals*, E82-A(8):1463–1474, August 1999.

[53] D. Pelleg and A. Moore. Accelerating exact k-means algorithms with exact reasoning. In *Fifth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 15-18, 1999, San Diego, CA, USA*, 1999.

[54] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Seventeenth International Conference on Machine Learning, Stanford University, June 29-July 2, 2000*, 2000.

[55] Olaf Pichler, Andreas Teuner, and Bedrich J. Hosticka. An unsupervised texture segmentation algorithm with feature space reduction and knowledge feedback. *IEEE Transactions on Image Processing*, 7(1):53–61, January 1998.

[56] Robert Porter and Nishan Canagarajah. A robust automatic clustering scheme for image segmentation using wavelets. *IEEE Transactions on Image Processing*, 5(4):662–665, April 1996.

[57] Jan Puzicha and Joachim M. Buhmann. Multiscale annealing for real-time unsupervised texture segmentation. Technical report, Institut fr Informatik III, University of Bonn, April 1997. Technical Report IAI-TR-97-4.

[58] Kannan Ramchandran, Martin Vetterli, and Cormac Herly. Wavelets, subband coding, and best bases. *Proceedings of the IEEE*, 84(4):541–560, Apr 1996.

[59] Trygve Randen. Personal homepage. http://www.ux.his.no/ ∼tranden/data.html.

[60] Trygve Randen. *Filter and Filter Bank Design for Image Texture Recognition*. Ph.d. thesis, Norwegian University of Science and Technology, Stavanger College, 1997. Obtained from WWW.

[61] Trygve Randen and John Håkon Husøy. Novel approaches to multichannel filtering for image texture segmentation. In *Proc. Visual Communication and Signal Processing, Boston*, pages 626–636, November 1993.

[62] Trygve Randen and John Håkon Husøy. Multichannel filtering for image texture segmentation. *Optical Engineering*, 33(8):2617–2625, August 1994.

[63] Trygve Randen and John Håkon Husøy. Optimal texture filtering. In *Int. Conf. on Image Proc., Washington D.C.*, volume 1, pages 374–377, October 1995.

[64] Trygve Randen and John Håkon Husøy. Filtering for texture classification: A comparative study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):291–310, April 1999.

[65] Todd R. Reed and Harry Wechsler. Segmentation of texxtured images and gestalt organization using spatial/spatial-frequency representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):1–12, January 1990.

[66] De Backer S., Naud A., and Scheunders P. Nonlinear dimensionality reduction techniques for unsupervised feature extraction. *Pattern Recognition Letters*, 19:711–720, 1998.

[67] M. Sharma and S. Singh. Evaluation of texture methods for image analysis. In *Proc. 7th Australian and New Zealand Intelligent Information Systems Conference, Perth*, November 2001.

[68] Eero P. Simoncelli and Javier Portilla. Texture characterization via joint satistics of wavelet coefficient magnitudes. In *Proceedings of Fifth International Conference on Image Processing*, volume I. IEEE Computer Society, October 1998.

[69] Gilbert Strang and Truong Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.

[70] Wim Sweldens. The lifting scheme: A new philosophy in biorthogonal wavelet construction. In *Wavelet Applications in Signal and Image Processing III*, pages 68–79. Proc. SPIE 2569, 1995.

[71] Wim Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.

[72] Wim Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1997.

[73] Wim Sweldens and Schröder. Building your own wavelets at home. Technical report, University of South Carolina, 1995. Obtained from WWW.

[74] Andreas Teuner, Olaf Pichler, and Bedrich J. Hosticka. Unsupervised texture segmentation of images using tuned matched gabor filters. *IEEE Transactions on Image Processing*, 4(6):863–870, Jun 1995.

[75] D. L. Toulson and J. F. Boyce. Segmentation of mr images using neural nets. *Image and Vision Computing*, 10(5):324–328, 1992.

[76] Michael Unser. Texture classification and segmentation using wavelet frames. *IEEE Transactions on Image Processing*, 11(4):1549–1560, Nov 1995.

[77] Michael Unser and Murray Eden. Multiresolution feature extraction and selection for feature segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):717–728, 1989.

[78] Michael Unser and Murray Eden. Nonlinear operators for improving texture segmentation based on features extracted by spatial filtering. *IEEE Transactions on Systems, Man, And Cybernetics*, 20(4):804–815, July 1990.

[79] G. Van de Wouwer, P. Sheunders, S. Livens, and D. Van Dyck. Wavelet correlation signatures for color texture classification. *Pattern Recognition*, 32(3):443–451, 1999.

[80] G. Van de Wouwer, P. Sheunders, and D. Van Dyck. Statistical texture characterization from discrete wavelet representations. *IEEE Transactions on Image Processing*, 8(4):592–598, 1999.

[81] Martin Vetterli and Jelena Kovačević. *Wavelets and Subband Coding*. Prentice-Hall PTR, 1995.

[82] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, Egham, TW20 0EX, UK, 1998.

[83] Mladen Victor Wickerhauser. *Adapted Wavelet Analysis from Theory to Software*. A.K. Peters, 1994.

[84] Zhi-Yan Xie and Michael Brady. Texture segmentation using local energy in wavelet scale space. In *Computer Vision, ECCV '96: fourth European Conference on Computer Vision, Cambridge, UK, April 15-18, 1996: Proceedings*, pages 304–313, 1996.