MULTI-CHANNEL FILTERING TECHNIQUES FOR TEXTURE SEGMENTATION AND SURFACE QUALITY INSPECTION

By

Farshid Farrokhnia

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical Engineering

1990

Abstract

Multi-Channel Filtering Techniques for Texture Segmentation and Surface Quality Inspection

By

Farshid Farrokhnia

This dissertation focuses on the multi-channel filtering approach to texture analysis. We combine this biologically motivated approach with analytical and signal analysis considerations to develop powerful, generally applicable texture analysis techniques. First, a detailed texture segmentation algorithm is proposed that uses a bank of even-symmetric Gabor filters to represent the channels. This representation is augmented with a systematic filter selection scheme based on an intuitive least squares error criterion. By introducing a nonlinear stage following the linear filtering operations, a multi-scale 'blob detection' mechanism is created. 'Feature images' are then obtained by computing the "energy" in a small neighborhood around each pixel, in each 'response image'. These energy features capture the attributes of the blobs without the need for extracting them. The texture segmentation experiments show that these features can discriminate among a large number of textures, including some artificially generated texture pairs with identical second- and third-order statistics. Both *unsupervised* and *supervised* texture segmentation experiments are reported. In the supervised segmentation experiments a feed-forward neural network is used, in addition to several other classifiers.

We also develop a new technique to obtain an edge-based segmentation by combining the magnitude responses of Canny edge detectors to the feature images. The region-based and edge-based segmentation techniques each have certain weaknesses. To eliminate these weaknesses we propose an integrated approach that combines the region- and edge-based segmentations to produce a new, improved segmentation. The integrated approach results in a truly unsupervised segmentation technique by eliminating the need for knowing the "true" number of texture categories.

Finally, we address a practical problem involving automated visual inspection of the textural appearance of automotive metallic finishes. We address imaging and preprocessing requirements and demonstrate that a multi-channel filtering technique can be used to successfully characterize the finish texture. Two alternative methods for grading the degree of uniformity of the finish texture are developed. The 'texture grading' experiments show that there is a high correlation between the texture uniformity grade and the visual scaling of the finish samples by finish inspection experts.

To my parents Parikh-Niaz and Doursun-Bibi,
and my wife Katy

Acknowledgements

I would like to express my sincere gratitude to my advisor Prof. Anil K. Jain. His guidance and constant encouragement were crucial for keeping research objectives in perspective and keeping me motivated. I also would like to thank Professors Mihran Tüceryan, John Deller Jr., Hassan Khalil, and V. Mandrekar for serving on the guidance committee. Their criticisms and recommendations have significantly improved the content and presentation of this dissertation. I am particularly grateful for the many constructive discussions I had with Prof. Tüceryan.

I also consider myself very fortunate for being part of the Pattern Recognition and Image Processing (PRIP) Group. I would like to thank many current and former graduate students in the PRIP Group for their support. Thanks to Dr. Chaur-Chin Chen, Dr. Sei-Wang Chen, Dr. Patrick Flynn, Dr. Joseph Miller, Debby Trytten, Satish Nadabar, Narayan Raja, Sally Howden, Greg Lee, Tim Newman, Jian-Chang Mao, Sushil Battacharjee, and David Marks, our lab manager, for making PRIP Laboratory a friendly place to work and a great environment in which to do research. During my graduate studies at Michigan State University, I enjoyed and benefited from the friendship, help, and encouragement of my long time friends, Dr. Farzad Esfandiari, Dr. Mani Azimi, Carl Erickson, Javad Kalantar, Dr. Mahmud Khodadad, and many others.

I am truly grateful to Dr. David Alman, of Automotive Products Division at E.I. Du Pont De Nemours & Company Inc., for his support throughout my Ph.D. program. The content of Chapter 5 on quality inspection of metallic finish texture is part of an ongoing research program supported by a grant from Du Pont. The support from the National Science Foundation infrastructure grant CDA-8806599 was also critical. The computing facilities acquired through this grant, were instrumental for carrying out this research.

Last, but not least, I would like to thank my wife, Katy, whose love and support were crucial for completing this dissertation successfully.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In computer vision or image analysis, an important goal is to "summarize" meaningful information in an image, that is otherwise distributed among a large number of pixels. For example, significant research effort is directed toward extracting segments of an image that correspond to objects or other physical entities. For intensity images, differences in average gray value alone are not always sufficient to obtain a segmentation. Rather, one has to rely on differences in the spatial distribution of gray levels of neighboring pixels — that is, on differences in *texture.*

This dissertation focuses on a particular approach to texture analysis known as the *multi-channel filtering approach.* This approach is inspired by the multi-channel filtering theory of visual information processing in the early stages of the human visual system. First proposed by Campbell and Robson [9], the theory holds that the visual system decomposes the retinal image into a number of filtered images, each of which contains intensity variations over a narrow range of frequency (size) and/or orientation. In texture analysis, such a decomposition is intuitively appealing, because it allows us to exploit differences in dominant frequencies and orientations of different textures. When combined with analytical and signal analysis considerations, this biologically motivated approach has the potential to produce powerful, generally applicable techniques for texture analysis. In this dissertation, we develop and evaluate several such techniques for segmenting images on the basis of texture. We model the 'channels' by a bank of even-symmetric Gabor filters, and propose an intuitive least squares error criterion for filter selection.

Texture analysis also plays an important role in industrial quality inspection problems. In many cases, the quality of a surface is best characterized by its texture. Texture analysis techniques have been used for controlling the quality of paper in paper-rolling mills [17], for detecting and classifying common surface defects in wood [23], and for determining the degree of carpet wear [84]. In this dissertation, we also address a practical problem involving automated visual inspection of automotive metallic finishes. We demonstrate that a multi-channel filtering technique can be used to successfully characterize the finish texture, and develop two alternative methods for grading the degree of uniformity of the finish texture.

The remainder of this chapter discusses visual information processing in both biological and artificial vision systems. We emphasize the role of texture in image analysis or computer vision and give a taxonomy of texture analysis techniques. We conclude the chapter by providing an outline of the dissertation.

1.1 Vision in Man and Machine

It has been said that "a picture is worth a thousand words". An estimated 75% of information about our environment is obtained through our visual system [17]. With increased reliance on visual information has come the need for visual information processing systems that can 'look at' and 'interpret' various types of imagery. For example, thousands and thousands of aerial reconnaissance images are taken everyday. These images can not be screened and analyzed by human experts alone. In medical applications, x-ray, ultrasound, or other kinds of imagery need to be processed and analyzed quickly and reliably. Increased availability and affordability of electronic imaging systems has made it possible to use image analysis techniques to address these problems.

Another reason for the increasing interest in building machines that can process and analyze images is for automation of visual inspection tasks. For industrial products that require visual inspection, increased automation of production lines has turned the inspection stage into a significant bottleneck. Within the next decade, as much as 90% of all industrial visual inspection tasks might be performed by computer vision systems [17]. Using human inspectors does have certain advantages. Humans can adapt to changes in the inspection requirements or to new inspection tasks very quickly. Human versatility and judgement, therefore, make strict and detailed specification of product requirements and tolerances unnecessary. However, humans are usually affected by fatigue, psychological state, and monotony. Machine vision systems, on the other hand, can provide reliable decisions based on objective criteria that are not expected to change. Also, many tasks that involve working in dangerous environments, such as mines, can be done safely and more efficiently by robots that are capable of analyzing visual information.

Although visual information is not necessary for all automated tasks, integrating visual data does have advantages. Most mechanical systems — for example, gauging and surface inspection systems — are being replaced by optical systems that are much faster. In other applications, integration of visual data with data that is sensed through other sensors, such as thermal or range imagery allows for building more robust systems.

A distinction must be made between image processing and image analysis (or computer vision.) Digital image processing, in general terms, refers to the processing of a two-dimensional picture (or any two-dimensional data) by a digital computer. In image processing operations, such as image restoration or enhancement, the output is another image. Edge detection is another common image processing operation. Such operations are also referred to as 'low-level' processing. This is because the information contained in the image is still distributed among individual pixels. For example, in an edge image each pixel is labeled as an edge or non-edge pixel. The aim of image analysis, on the other hand, goes beyond such operations and involves interpreting the *content* of the image. When edge pixels in an edge image are grouped together as line segments, for example, one obtains a more compact and meaningful representation. Similarly, grouping pixels into regions to obtain a meaningful segmentation of an image results in a compact description of the image. Sophisticated computer vision systems are expected to interpret or assign labels to regions or surfaces in images. Image analysis therefore involves tasks such as feature extraction, segmentation, classification, and recognition. In these cases the output of the vision system is usually a symbolic description of the input image.

Computer vision techniques invoke concepts from diverse fields such as optics, digital signal processing, estimation theory, information theory, stochastic processes, visual perception, pattern recognition, artificial intelligence, and computer graphics. Computer vision is a rapidly evolving field with growing applications in science and technology. This area holds the possibility of developing machines that can perform many of the visual functions of human beings. While many theoretical and technological breakthroughs are required before we could build such sophisticated vision systems, there is an abundance of vision applications that can be realized through available algorithms and hardware.

In recent years, a major trend in computer vision research has been the integration of biological and psychological studies of vision and image analysis techniques.

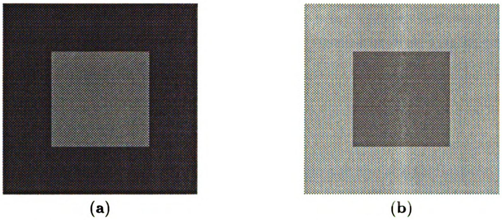


Figure 1.1: Simultaneous contrast phenomenon in human visual system. The perceived luminance, i.e. the *brightness*, of an object is influenced by the luminance of its background. A square block of constant gray-level of 128 (**a**) surrounded by gray-level of 32, (**b**) surrounded by gray-level of 192.

Clearly, computer vision systems are not restricted to the limited range of the electromagnetic spectrum, called visible light, for input. Neither are they restricted to use hardware or processing architectures similar to that of biological systems. However, study of psychological and neurophysiological properties of visual systems in humans and other living beings holds great potential for developing sophisticated algorithms for image analysis. After all, many applications of computer vision involve developing vision systems that can imitate human performance.

For example, it is known that two objects with different surroundings may have identical *luminance,* but different *brightness* values [62, Sec. 6.2]. (See Figure 1.1.) In other words, the perceived luminance, which we call brightness, depends on the luminance of the surround. This phenomenon is called *simultaneous contrast* and is caused by the fact that our visual system is sensitive to luminance contrast rather than to absolute luminance values themselves. Such perceptual phenomena need to be considered and incorporated in any computer vision system that is to behave like humans. Psychophysical experiments, such as those by Campbell and Robson [9], and Julesz and his co-workers [53, 56] have contributed a great deal to our understanding of the mechanisms and properties of the human visual system.

As noted before, computer vision or image analysis involves tasks such as feature extraction, image segmentation, and recognition that require interpreting the content of an image. Image texture, which is the main subject of this dissertation, is one of the richest sources of information for analyzing many images. Following sections discuss different characteristics of image textures, texture analysis approaches, and major texture analysis tasks.

1.2 What Is Texture?

Texture is an intrinsic property of all surfaces. Humans use texture features in analyzing retinal images of a scene. This implies that texture is an easily understood concept. However, it is very difficult to define texture in a concise mathematical sense. A common definition of texture describes a basic local pattern that is repeated in a nearly periodic manner over some image region. This definition is appropriate for 'macrostructure' textures — textures whose underlying patterns can be easily detected. This definition, however, does not apply to 'microstructure' textures — textures whose underlying patterns are not obvious. Surprisingly, even the more random looking textures seem to possess a distinctive property that is readily identified by the human observer.

The lack of a comprehensive definition of texture really stems from the lack of good understanding of texture and texture models. The proliferation of texture analysis techniques over the last two decades was stimulated by lack of agreement as to how texture should be measured.

Figure 1.2 shows some natural and artificial textures. Some textures, such as that of a rough wall surface or canvas, are perceived because of the underlying physical structure of the surface. Others, such as the texture of a checkerboard or ruled paper, are perceived because of the design patterns or marks on the surface. In some cases, a collection of objects is viewed as a single textural entity, as in the case of grass or a brick wall. An important characteristic of texture is its dependence on spatial resolution. For example, a tiled floor is perceived to have a nearly regular (cellular) texture whose elements are the individual tiles. But, when attention is focused on a single tile, one perceives the random texture of the tile.

We often correlate visual textures with tactile sensations such as smoothness, coarseness, graininess. We also describe them with adjectives such as regular, directional, line-like, etc. Intuitively, developing computational measures that correspond

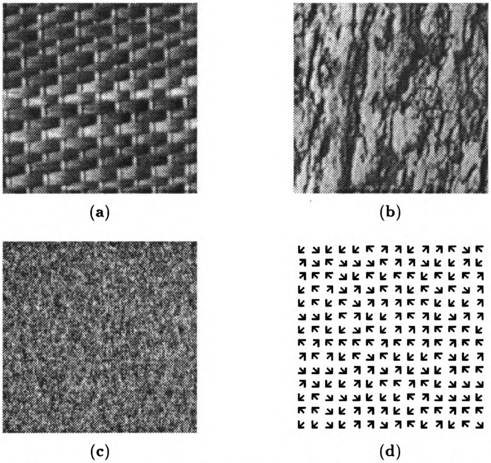


Figure 1.2: Some natural and artificial textures. (**a**) 'Straw matting' (D55) from the Brodatz album [7]. (**b**) 'Bark of tree' (D12) from the Brodatz album. (**c**) An artificial texture generated from a Gaussian Markov random field model [14]. (**d**) A synthetic texture [53].

to such perceptual attributes of textures is very appealing. Tamura *et al.* [86] developed six computational measures corresponding to coarseness, contrast, directionality, line-likeness, regularity, and roughness. They compared their measures with psychological data on 16 natural textures. The rank correlations between their measures and the visual judgements by human subjects were between 0.65 and 0.90.

Psychological experiments have shown that human beings are capable of discriminating certain types of texture preattentively; that is, by viewing the texture for a very short period of time so as to avoid detailed scrutiny by higher-level processes. Most of these experiments made use of computer-generated patterns or textures which were void of familiar cues. For example, Julesz and his co-workers used a large number of computer-generated texture pairs in their texture discrimination experiments [52, 55, 57]. Such experiments have greatly contributed to our understanding of texture discrimination processes in the human visual system.

Haralick [43] emphasizes the interaction between tone (i.e., gray-level) and texture in an image. He points out that tone and texture are both present in the image at the same time, but depending on circumstances one or the other may dominate. When there is a large variation in the tonal primitives in a relatively small area in an image, texture becomes the dominant property. Obviously, the amount of perceived variation in the tonal primitives depends strongly on the size of the image area being investigated. For example, in an extreme case where the area consists of a single pixel, the texture property is completely suppressed. What characterizes image texture is, therefore, the tonal primitives as well as the spatial relationships between them.

What is important about texture, therefore, is that it is the property of regions or neighborhoods rather than individual pixels. The interaction between gray-levels of neighboring pixels can therefore be used to characterize textures.

1.3 Texture Analysis: A Taxonomy

In this section we present a general classification of texture analysis approaches, and describe them briefly. Several surveys of texture analysis techniques have appeared in the literature. Haralick [43] compiled a comprehensive survey of *statistical* and *structural* approaches to texture. Van Gool *et al.* [91] have also published a survey of texture analysis techniques. Their survey emphasizes texture classification techniques; texture segmentation techniques are covered only briefly. Our classification of existing computational approaches to texture analysis consists of the following categories: 1) statistical approach, 2) structural approach, 3) model-based approach, and 4) multi-channel filtering approach. While an unambiguous and exhaustive classification of texture analysis techniques is impossible, we believe that the above categories represent a compact and descriptive classification. In Chapter 2, we will review multi-channel filtering techniques in more detail.

1.3.1 Statistical Approach

In the statistical approach to texture analysis, the image texture is represented by a point or a pattern in the feature space, where the features are various statistics of the gray level distribution in the image [62, Ch. 9]. Since texture is a property of image regions, as opposed to individual pixels, these statistics try to capture the interactions, or dependencies, among neighboring pixel values. Well-known examples are features computed from gray-level co-occurrence matrices [43], and autocorrelation and power spectrum features.

1.3.2 Structural Approach

Certain textures, in particular 'man-made' textures, possess a high degree of regularity. Textures, such as one perceived on a brick wall, can be described by their building blocks or primitives and their placement rules. This approach to texture analysis is referred to as the *structural* approach [62, Ch. 9]. Either the primitives or the placement rules, or both, may have a random component associated with them. Given the primitives and their placement rules, one can generate samples of the texture. Unfortunately, extracting the primitives from a given texture is not an easy task. This difficulty imposes a serious limitation on the applicability of structural approaches to practical problems.

1.3.3 Model-Based Approach

A third approach to texture analysis, *viz.* the *model-based* approach, has received a great deal of attention in the recent years. Model-based techniques attempt to capture the dependencies among neighboring pixel values by fitting an analytical function to the textured image. Most model-based techniques treat texture as a realization of a two-dimensional stochastic process, or a random field. Once an appropriate model of a given texture has been found, the parameters of the model would completely specify the texture. The ability to represent a texture with a small number of parameters makes the storage and processing of texture images extremely efficient. Some of the well-known model-based techniques for texture classification and segmentation are based on Markov random field (MRF) models [3, 13, 24], mosaic models [1], and fractals [59, 76].

1.3.4 Multi-Channel Filtering Approach

In this dissertation, we focus on a particular approach to texture analysis which is referred to as the *multi-channel filtering* approach. The multi-channel filtering paradigm in image analysis has received considerable attention in the past decade [62, Ch. 6]. This paradigm is inspired by a multi-channel filtering theory for processing visual information in the early stages of the human visual system. According to the theory [9], the human visual system decomposes the retinal image into a number of filtered images, each of which contains intensity variations over a narrow range of frequency (size) and orientation. The psychophysical experiments that suggested such a decomposition used various grating patterns as stimuli and were based on adaptation techniques [9]. Subsequent psychophysiological experiments provided additional evidence supporting the theory [28, 82].

In texture analysis, the multi-channel filtering approach is intuitively appealing, because it allows us to exploit differences in dominant frequencies and orientations of different textures. A decomposition of the original textured image based on frequency is also in agreement with the need for a multi-resolution approach to texture analysis. The need for processing images at different scales or resolutions is well recognized in image analysis and computer vision [71]. An important advantage of the multi-channel filtering approach is that one can use simple statistics of gray values in the filtered images as texture features. This simplicity is the direct result of decomposing the original textured image into several filtered images with limited spectral information.

In Chapter 2, we will discuss, in more detail, the biological motivations and analytical considerations for the multi-channel filtering approach, and survey existing multi-channel filtering techniques for texture analysis.

1.4 Texture Analysis Tasks

Textural cues are essential for basic image analysis tasks such as image classification and segmentation. In *texture classification*, the entire image is assigned to one of several known categories, on the basis of its textural properties. A statistical approach is often used to represent each image by a feature vector containing various texture measures computed over the entire image. *Texture segmentation*, on the other hand, involves identifying regions with "uniform" textures in a given image. Appropriate measures of texture are needed in order to decide whether a given region has uniform texture. Sklansky [85] has suggested the following definition of texture which is appropriate in the segmentation context. "A region in an image has a constant texture if a set of local statistics or other local properties of the picture are constant, slowly varying, or approximately periodic." Texture segmentation, therefore, has both local and global connotations — it involves detecting invariance of certain local measures or properties over an image region.

Compared to texture classification, texture segmentation is a much more difficult problem. In texture segmentation the number of texture categories that are present in an image and the information about the size, shape, and number of textured regions often are not known *a priori*. In fact, some texture segmentation problems have more than one possible solution, and determining the "correct" segmentation depends on the goal of the analysis and may require additional knowledge of the scene. Texture segmentation may be achieved in one of two ways. A *region-based* segmentation is obtained by identifying regions with homogeneous textures. This is usually done by computing texture measures for each pixel (or block of pixels) and assigning pixels with similar measures to the same texture category. An *edge-based* segmentation, on the other hand, is obtained by detecting the boundaries between the textures.

In addition to image classification and segmentation, gradients of texture primitives — density gradient, area gradient, and aspect-ratio gradient — can be used to estimate the orientation of a surface patch in the scene. That is, to extract three-dimensional information from a two-dimensional image. This so called problem of *shape-from-texture* is a difficult and challenging problem, because it requires that both texture and the change (gradient) in texture be characterized simultaneously. In a recent paper, Blostein and Ahuja [4] review the problem of shape-from-texture and propose a new technique that integrates extraction of texture elements with estimation of surface orientation. Coggins and Jain [21] have explored the effect of texture gradients on texture measures obtained using a multi-channel filtering technique.

1.5 Outline of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, we discuss biological motivations and analytical considerations for a multi-channel filtering approach to texture analysis, and survey the existing techniques. Chapter 3, describes the main components of our proposed texture segmentation algorithm. The choice of filter parameters, filter selection, computation of texture features, and the procedure used to integrate the *feature images* are described. We report both *unsupervised* and *supervised* texture segmentation experiments. In Chapter 4, we combine our region-based texture segmentation technique with an edge-based segmentation technique to eliminate the need for knowing the exact number of texture categories. Automated visual inspection of metallic finish texture is described in Chapter 5. Finally, in Chapter 6, we summarize the results and contributions of the thesis and discuss future research directions.

Chapter 2

Multi-Channel Filtering

2.1 Biological Motivations

In psychophysics, early attempts to model the human visual system focused on its overall input/output characteristics [62, Sec. 4.3]. To measure its "transfer function", for example, sinusoidal (sine-wave) gratings with different spatial-frequencies were used as visual stimuli[1] (Figure 2.1). Optical systems are often characterized by their *modulation transfer function* (MTF), which is determined by comparing some quantitative measurement of the input with that of the output. For sinusoidal gratings, a common measure is contrast, C, which is defined by

$$C = \frac{I_{max} - I_{min}}{I_{max} + I_{min}},$$

where I_{max} and I_{min}, respectively, are the maximum and minimum intensities of the grating. Thus, one way to define the MTF of the human visual system is

$$H(u) = \frac{\text{output contrast}}{\text{input contrast}}.$$

Unfortunately, it is not possible to measure the output contrast! The practical alternative is to use a psychological measurement known as *contrast sensitivity.* Experimentally, the contrast sensitivity is measured as follows [62]. The subject views the stationary sinusoidal gratings on a display which allows him/her to vary the contrast without changing the average intensity. For each frequency u, the threshold contrast $C_t(u)$ necessary to barely distinguish the grating from a uniform illumination is measured. The *contrast sensitivity function* (CSF) is then defined as:

$$\text{CSF}(u) = \frac{1}{C_t(u)}$$

Figure 2.2 shows a typical CSF. Clearly, the CSF is an oversimplified, "black box" representation of the human visual system. Nonetheless, it serves as a useful qualitative measure of the sensitivity of the human visual system to visual patterns of

[1]Spatial-frequencies are commonly given in cycles per degree of visual angle, although cycles per centimeter of test pattern or cycles per millimeter on the retina may be used.

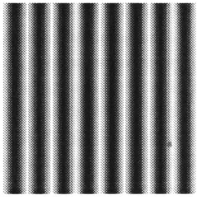


Figure 2.1: An example of sinusoidal gratings used as stimuli in the measurement of contrast sensitivity function.

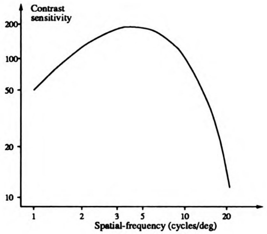


Figure 2.2: A typical contrast sensitivity function for the human visual system. (Redrawn from Campbell and Robson [9].)

different frequencies.

The evidence for multiple 'channels', as opposed to a single channel, in the human visual system comes from psychophysical as well as psychophysiological experiments. Campbell and Robson [9] proposed that the visual system decomposes the retinal image into a number of filtered images, each of which contains intensity variations over a narrow range of frequency (size) and orientation. The psychophysical experiments that suggested such a decomposition used various grating patterns as stimuli and were based on adaptation techniques [9].

Other experiments verified the frequency and orientation tuning properties of certain cells in the visual cortex of some mammals. De Valois *et al.* [28], for example, recorded the response of simple cells in the visual cortex of the Macaque monkey to sinusoidal gratings with different frequencies and orientations. It was observed that each cell responds to a narrow range of frequency and orientation only. Therefore, it appears that there are mechanisms in the visual cortex of mammals that are 'tuned' to combinations of frequency and orientation in a narrow range. These mechanisms are often referred to as 'channels' and are appropriately interpreted as band-pass filters.

More recently, Beck *et al.* [2] reported psychophysical experiments on texture segmentation using patterns containing squares with different gray levels or different colors. They conclude that the results of their experiments "support the argument that the higher order processes in texture segregation have access to information corresponding to the outputs of the spatial frequency channels".

Earlier experiments aimed at characterizing the channels focused on measuring the center frequencies and the frequency and orientation bandwidths of the channels. Several functional characterizations of channels in the frequency domain evolved from these experiments [38, 83]. More recent characterizations, however, have been largely based on psychophysiological data. In particular, some filter characteristics have been obtained by fitting band-limited functions to the receptive field profiles of simple cells in the visual cortex of some mammals [25, 70, 95]. In signal processing and systems theory terminology, a receptive field profile can be interpreted as the *impulse response* of a cell.

2.2 Analytical Considerations

The significance of frequency (size) and orientation cues for analyzing textures motivates the following view of the problem of texture analysis. *Texture analysis, in the early stages, relies on local frequency and orientation measurements.*

Once we accept this view, the question becomes "how do we measure it?" For simplicity, we present our analysis in one-dimensional space. The extension to two-dimensions is usually straightforward since it can be viewed as one-dimensional frequency estimation along different orientations. The 1-D analysis, therefore, can readily be extended to 2-D. Let $s(x)$ be the 1-D textured "image". To further facilitate our analysis, let us assume that $s(x)$ is continuous and may have infinite extent.

Fourier decomposition (transform) of $s(x)$ provides one way to represent its frequency content.

$$S(u) = \int_{-\infty}^{+\infty} s(x)\, e^{-j2\pi ux}\, dx \tag{2.1}$$

However, Fourier transform estimates the frequency *globally*. As seen in its definition, each frequency is influenced by $s(x)$ at all x values. That is, we can not tell the *location* from which the frequencies come. For texture analysis tasks, such as texture segmentation, we are interested in the frequency content in small regions around each pixel. One way to localize the estimation of frequencies is to use a window Fourier transform, which is defined by

$$S_w(u,\zeta) = \int_{-\infty}^{+\infty} s(x)\, w(x-\zeta)\, e^{-j2\pi ux}\, dx\,, \tag{2.2}$$

where $w(x)$ is a low-pass function. When $w(x)$ is a Gaussian function, the window Fourier transform is referred to as a Gabor transform [36]. The notion of localized frequency measurement is closely related to combined space-frequency image representations. Porat and Zeevi [79] have provided a thorough analysis of image representation using Gabor elementary functions (GEF). Bovik [5] addresses optimality criteria for channel filters, where each "narrow-band" filter is expressed as the product of an equivalent low-pass filter with a complex sinusoidal plane wave. As a result, the filtering operations are reminiscent of window Fourier transforms. Also, Reed and Wechsler [80] use a different "spatial/spatial-frequency" representation, based on Wigner distribution, to study the texture segmentation and clustering/grouping problems.

The ability to localize frequency estimation comes at the expense of more complexity. In particular, window Fourier transforms do *not* result in an orthogonal decomposition of $s(x)$. Computing such decompositions, therefore, is not straightforward [68]. Daugman [27] has proposed a neural network architecture for computing optimal coefficients in arbitrary two-dimensional transforms.

It is well known in signal analysis that there is a trade-off between the effective width of a localized signal (pulse) in the time domain and its bandwidth in the frequency domain [45]. Signals with short durations in the time domain will have large bandwidths in the frequency domain, and vice versa. The bandwidth and duration of a signal can be defined in several different ways. However, the inverse relationship between duration and bandwidth applies irrespective of these definitions [45, p. 40]. This inverse relationship implies a trade-off between spread or "uncertainty" of a localized signal in the time and the frequency domains.

A similar trade-off applies to two-dimensional signals. The uncertainty principle [26] relates the detection and localization performance of a filter. Texture analysis tasks such as segmentation require simultaneous measurements both in the spatial and the spatial-frequency domains. High resolution in the spatial-frequency domain is desirable because it allows us to make finer distinctions among different textures. On the other hand, accurate localization of texture boundaries requires high resolution in the spatial domain.

In a window Fourier transform, the effective width and bandwidth of the basis functions are determined by the window function $w(x)$. A Gaussian window function minimizes the joint uncertainty in the time and the frequency domains [36]. Texture analysis considerations, therefore, point to Gabor transform (decomposition) as the ideal means of frequency estimation. In the context of texture segmentation, however, smaller width for basis functions with higher spatial-frequencies will result in better localization of the texture boundaries. This suggests that the widths of the basis functions should be inversely proportional to their frequency. In other words, they should have a constant bandwidth on the logarithmic scale. When this is the case, the window Fourier transform becomes a wavelet transform [67, 68] defined as follows.

$$S_w(u,\zeta) = \int_{-\infty}^{+\infty} s(x)\, \sqrt{u}\, w(u\,(x-\zeta))\, e^{-j2\pi ux}\, dx\,. \tag{2.3}$$

In Chapter 3, we propose a texture segmentation algorithm that models the 'channels' by a set of two-dimensional Gabor filters. It will be shown that the filters constitute an approximate orthogonal basis for a wavelet transform, with the Gabor function as the wavelet.

2.3 Existing Techniques

The main issues involved in the multi-channel filtering approach to texture analysis in general, and texture segmentation in particular, are:

1. functional characterization of the channels and the number of channels,
2. extraction of appropriate texture features from the filtered images,
3. the relationship between channels (dependent vs. independent),
4. integration of texture features from different channels to produce a segmentation, and
5. segmentation method (region-based vs. edge-based).

Different multi-channel filtering techniques that are proposed in the literature vary in their approach to one or more of the above issues.

In this section, we survey existing multi-channel filtering techniques. Our emphasis is on various characterizations of the 'channels' and on texture segmentation algorithms. The background developed in this chapter will set the stage for presenting our proposed texture segmentation algorithm in Chapter 3, which uses a bank of Gabor filters. In Chapter 5, we use isotropic frequency-selective filters [19, 20] to analyze textural appearance of metallic finishes. Gabor filters and the isotropic frequency-selective filters are both described in this section.

Earlier multi-channel filtering techniques used the spatial-frequency domain characterization of the channels based on psychophysical and psychophysiological data. Faugeras [33] used the spatial-frequency domain characterization of the channels by Sakrison [83], which consisted of bandpass filters with both frequency- and orientation-selective properties. The modulation transfer function[2] (MTF) of these filters, in polar coordinates, is given by

$$H(f,\theta) = H_r(f)\,H_a(\theta)\,, \tag{2.4}$$

where

$$\begin{aligned} H_r(f) &= \left\{1 + \frac{(f-f_0)^2}{w}\right\}^{-1/2}, \\ H_a(\theta) &= \exp\left\{-\frac{1}{2}\frac{(\theta-\theta_0)^2}{b}\right\} + \exp\left\{-\frac{1}{2}\frac{(\theta-\theta_0-\pi)^2}{b}\right\}, \end{aligned}$$

f_0 and θ_0 determine the center radial frequency and orientation of the filter. w and b, on the other hand, determine the radial and angular bandwidths, respectively.

Faugeras computed the texture features by taking the sixth-order norm of the pixel values in a filtered image and then averaged them over the entire image. He chose the sixth-order norm because it contains "information about the phase" in the input image, and because it offers a good compromise between too much averaging of details (corresponding to the Euclidean norm) and the ability to detect isolated noise spikes (corresponding to the infinity norm). Faugeras used a total of 27 filters - three radial frequencies and nine orientations. He showed the potential of these features by constructing texture classification experiments, but he did not give any algorithm for texture segmentation.

Coggins [19] used a different set of filters that are also specified in the spatial-frequency domain. Each filter has either frequency-selective *or* orientation-selective

[2]The modulation transfer function of a filter specifies the amount by which it modulates the magnitude of each frequency component of the input image.

property only. The MTFs of the frequency-selective filters are given by

$$H(u,v) = \exp\left\{-\frac{1}{2}\frac{(\ln\sqrt{u^2+v^2} - \ln\mu)^2}{\sigma_1^2}\right\}, \qquad (u,v) \neq (0,0), \qquad (2.5)$$

where μ is the center radial frequency and σ_1 determines the bandwidth of the filter. Note that these filters are defined on a logarithmic scale. The MTFs of the orientation-selective filters, on the other hand, are given by

$$H(u,v) = \exp\left\{-\frac{1}{2}\frac{A(u,v)^2}{\sigma_2^2}\right\}, \qquad (u,v) \neq (0,0), \qquad (2.6)$$

where

$$A(u,v) = \text{Min}\left\{|\tan^{-1}(\frac{v}{u}) - \alpha|, |\tan^{-1}(\frac{v}{u}) - (\alpha+\pi)|\right\}.$$

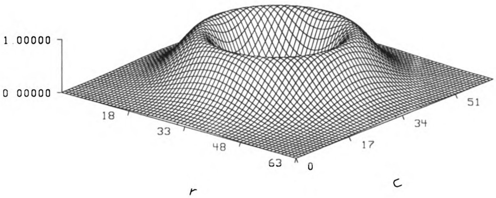
Here, $0 \leq \tan^{-1}(\cdot) < \pi$, α (in radians) is the center orientation, and σ_2 determines the orientation bandwidth of the filter. The value of MTFs at $(u,v) = (0,0)$, for both types of filters, was set to 1. So the mean gray value of each filtered image was the same as that of the input image.

The filter set used by Coggins [19] and Coggins and Jain [20] contained four orientation-selective filters tuned to 0°, 45°, 90°, and 135°. The number of frequency-selective filters in the filter set depended on the size of the image array. For a 128 × 128 image array, for example, they used six frequency-selective filters with center frequencies at 1, 2, 4, 8, 16, 32, and 64 cycles/image. Two examples of these filters are shown in Figure 2.3. Coggins and Jain demonstrated the utility of these filters for texture classification and segmentation. For texture classification, they use the average absolute deviation (AAD) from the mean gray value of each filtered image as texture features. The AAD feature for filtered image $o_k(x,y)$ is computed as follows:

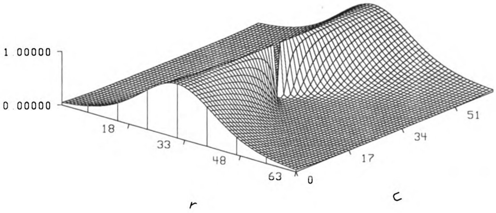
$$f_k = \frac{1}{N_r N_c}\sum_{a=1}^{N_r}\sum_{b=1}^{N_c} | o_k(a,b) - \bar{g}_k |, \qquad (2.7)$$

where N_r and N_c are the number of rows and columns of the image array, and $\bar{g}_k$ is the mean gray value of the filtered image[3]. Clearly, the number of texture features used depends on the number of filters, since there is one AAD feature corresponding to each filtered image.

[3]As pointed out earlier, the mean gray value ($\bar{g}_k$) of each filtered image is the same as that of the input image.



(a)



(b)

Figure 2.3: Examples of spatial filters used by Coggins and Jain [20]. The origin $(u, v) = (0, 0)$ is at $(r, c) = (32, 32)$. (**a**) A frequency-selective filter tuned to radial frequency of 16 cycles/image. (**b**) An orientation-selective filter tuned to $0°$.

Coggins and Jain use the AAD features also in their texture segmentation algorithm. However, instead of averaging over the entire image array, the AAD feature is computed over small overlapping windows around each pixel and is assigned to the center pixel. This local averaging process results in one 'feature image', $e_k(x,y)$, corresponding to each filtered image, $o_k(x,y)$. That is,

$$e_k(x,y) = \frac{1}{M^2} \sum_{(a,b)\in W_{xy}} \mid o_k(a,b) - \bar{g}_k \mid , \tag{2.8}$$

where W_{xy} is an $M \times M$ window centered at location (x,y). The collection of feature images, therefore, defines one feature vector (pattern) for each pixel in the original image. The following two-step procedure is used to obtain a segmentation. First, a pattern clustering algorithm is used to group a small subset of these patterns into a given number of clusters, and a generic label is assigned to patterns in each cluster. These labeled patterns are then used as 'training patterns' to classify all patterns (pixels). Coggins [19] and Coggins and Jain [20] successfully applied this algorithm to segment images containing natural as well as artificial textures. Jain [48] demonstrated the ability of the algorithm to segment images that contained artificially generated texture pairs with identical second- and third-order statistics.

More recently, a number of texture segmentation algorithms have been proposed that use two-dimensional Gabor filters. A Gabor function consists of a sinusoidal plane wave of some frequency and orientation, modulated by a two-dimensional Gaussian envelope. A "canonical" Gabor filter in the spatial domain is given by

$$h(x,y) = \exp\left\{-\frac{1}{2}\left[\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right]\right\} \cos(2\pi u_0 x + \phi), \tag{2.9}$$

where u_0 and ϕ are the frequency and phase of the sinusoidal plane wave along the x-axis (i.e. the 0° orientation), and σ_x and σ_y are the space constants of the Gaussian envelope along the x- and y-axis, respectively. A Gabor filter with arbitrary orientation, θ_0, can be obtained via a rigid rotation of the x-y coordinate system. These two-dimensional functions have been shown to be good fits to the receptive field profiles of simple cells in the striate cortex [70, 25].

As a spatial filter, we are interested in the frequency- and orientation-selective properties of a Gabor filter. These properties are more explicit in the frequency domain representation of a Gabor filter. With $\phi = 0$, the Fourier transform of the

Gabor function in (2.9) is real-valued and is given by

$$H(u,v) = A\left(\exp\left\{-\frac{1}{2}\left[\frac{(u-u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right]\right\}+\exp\left\{-\frac{1}{2}\left[\frac{(u+u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right]\right\}\right), \tag{2.10}$$

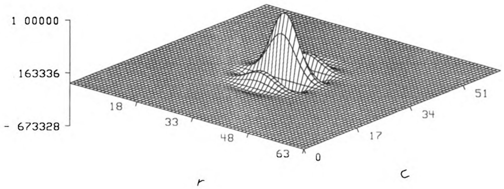
where $\sigma_u = 1/2\pi\sigma_x$, $\sigma_v = 1/2\pi\sigma_y$, and $A = 2\pi\sigma_x\sigma_y$. Figure 2.4 shows an even-symmetric Gabor filter and its MTF, in a 64×64 array.

An important property of Gabor filters is that they simultaneously achieve optimal joint localization, and hence resolution, in both the spatial domain and the spatial-frequency domain. Gabor [36] showed that one-dimensional Gabor functions uniquely achieve the lower bound of the uncertainty relationship $\Delta x\,\Delta u \geq 1/4\pi$, where Δx and Δu are the effective width and bandwidth of the signal in the one-dimensional spatial domain and the spatial-frequency domain, respectively (measured by the square root of the variance of the energy functions). Daugman [26] extended this result to two-dimensional Gabor functions, by showing that they uniquely achieve the lower bounds in the uncertainty relationships $\Delta x\,\Delta u \geq 1/4\pi$ and $\Delta y\,\Delta v \geq 1/4\pi$. Here, Δx and Δy are the effective widths in the spatial domain, and Δu and Δv are the bandwidths in the spatial-frequency domain. Texture analysis tasks such as segmentation require simultaneous measurements both in the spatial and the spatial-frequency domains. The above optimum property suggests that the Gabor filter is an ideal "tool" for analyzing textures. (See Section 2.2.)

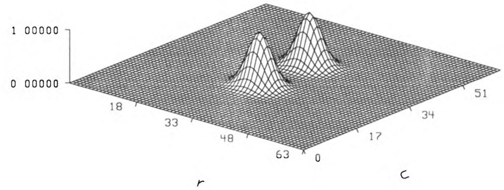
Turner [90] used a set of Gabor filters and demonstrated their potential for texture discrimination. The filters had four different frequencies, four orientations, and two quadrature phase pairs for each combination of frequency and orientation – a total of 32 filters. The filters were generated in the spatial domain. The spatial extent of all the filters was the same – they all had identical, circularly symmetric, Gaussian envelopes. The coefficients of each filter function were adjusted so that the mean gray value of each filtered image was zero. The input image was convolved by each filter function to obtain 32 filtered images. For computational efficiency, the convolution results were computed every 16th pixel in a row or column only, with the result being assigned to all the pixels in a 16×16 block.

For a given frequency and orientation, the filtered images $o_{k,1}(x,y)$ and $o_{k,2}(x,y)$, corresponding to a pair of filters with quadrature phase relationship, were combined to obtain a "phase insensitive" response, $o_k(x,y)$:

$$o_k(x,y) = \left[(o_{k,1}(x,y))^2 + (o_{k,2}(x,y))^2\right]^{1/2} \tag{2.11}$$



(a)



(b)

Figure 2.4: (**a**) An even-symmetric Gabor filter in the spatial domain. The radial frequency and orientation are 8 cycles/image-width and 0°, respectively. (**b**) Corresponding MTF. The origin is at $(r, c) = (32, 32)$.

This combination of pairs of filtered images transformed the initial 32 filtered images into 16 'response images'. In order to demonstrate the effectiveness of the Gabor filters in texture discrimination, Turner summed up these response images to obtain a single response image. A difference in the mean values of this 'total response' in different texture regions was taken as evidence of discrimination. In some cases, however, the difference in the mean values could only be revealed by adding a subset of response images rather than all of them.

By summing up the response images, Turner was actually performing a very crude feature extraction. However, adding the components of two feature vectors may result in similar values, even though individual components are very different. Turner's scheme falls short of producing a segmentation; it only demonstrates the potential of Gabor filters to obtain features that are capable of discriminating textures.

The texture segmentation algorithm proposed by Bovik and his co-investigators [6, 18] also uses Gabor filters. Like Turner [90], these authors also combine pairs of filtered images corresponding to filters with quadrature phase relationship. A more compact filter representation is used, however, where each filter pair is treated as a single *complex* Gabor filter. The real part of each complex filter is an even-symmetric Gabor filter (i.e., $\phi = 0$) and the imaginary part is an odd-symmetric Gabor filter (i.e., $\phi = \pi/2$). By linearity, the real and imaginary parts of each filtered image are responses to a pair of (real) Gabor filters with quadrature phase relationship.

Bovik *et al.* also combine the responses to each pair of filters (i.e., the real and imaginary parts of the response to a complex filter) to obtain a single response. Instead of using the Euclidean norm, however, they use the sum of absolute values. That is,

$$o_k(x,y) = | \, o_{k,1}(x,y) \, | + | \, o_{k,2}(x,y) \, | \tag{2.12}$$

These responses are then smoothed by a Gaussian weighted window "to counteract the effects of leakage and noise". The spread or space constant of this Gaussian filter is chosen to be slightly wider than the spread of the Gaussian envelope of the corresponding Gabor filter. This smoothing operation, however, can be interpreted as computing a measure of local energy using a weighted averaging window. We will, therefore, refer to these smoothed response images as feature images.

In their segmentation examples, Bovik *et al.* apply a peak-finding algorithm to the power spectrum of the image in order to determine the center frequencies of the appropriate Gabor filters. In addition, a "limited amount of human intervention" is used in determining the parameters of the Gabor filters. For example, for strongly oriented textures, the most significant spectral peak along the orientation direction is used. For periodic textures, on the other hand, the lower fundamental frequency is chosen.

The segmentation algorithm of Bovik *et al.* is based on the assumption that each texture has a distinct narrow range of frequencies, which is not present in other texture categories. The algorithm produces a region-based segmentation by labeling each pixel with the index of the complex Gabor filter which has the *maximum* response at that pixel. Using the indices of the filters as labels implies that the number of texture categories is constrained by the number of complex Gabor filters that are used. (If k filters are used then k labels are possible.) The algorithm, therefore, requires knowing the true number of texture categories, as well as their distinct narrow ranges of frequencies.

Tan and Constantinides [87] have used Gabor filters with quadrature phase relationship for texture classification and segmentation. Like Turner [90] they use (2.11) to obtain the response in each channel. For texture classification, a fixed set of Gabor filters tuned to one of four radial frequencies and one of four orientations is used. The mean and the standard deviation in each response image is used as texture features. For texture segmentation, on the other hand, the number of Gabor filters and their center frequencies are determined by identifying spectral peaks in the spatial-frequency domain. An edge-based segmentation is then obtained by "intensity gradient calculation" in each channel followed by "channel grouping", "thresholding" and "edge thinning". The paper [87] does not give the details of these stages. However, "channel grouping" appears to involve adding the responses of a gradient operator in different channels. The edge-based segmentation is obtained by thresholding this 'total' gradient response and thinning the resulting binary image.

Another texture segmentation algorithm that uses a set of Gabor filters is proposed by Perry and Lowe [77]. The filters have three frequencies (or *scales*, to use their terminology) corresponding to periods of 1, 2, and 4 pixels; eight orientations: 0, 30, 45, 60, 90, 120, 135, and 150 degrees; and two phases: 0, and 90 degrees. A procedure similar to that of Bovik *et al.* is used to obtain a set of feature images. Instead of using the feature vectors that are defined by these responses, however, they define two new feature vectors, based on the original feature vector, in an attempt to obtain a "more compact representation".

The first new feature vector is obtained as follows. First, the sum of all filter responses for all orientations are determined for each scale (frequency). The scale with the largest sum value is designated as "max scale". The responses of orientation filters with "max scale" and the "max scale" value itself form the first new feature vector. The other new feature vector is "a more compact version" of the first new feature vector and again emphasizes the orientation features. We must note that these new feature vectors are computed at "grid points" that are few pixels apart and, therefore, represent small blocks of pixels. The typical size of the blocks is 8×8.

A distance measure is defined for neighboring grid points by comparing their feature vectors[4]. An iterative procedure is then used to obtain a segmentation as follows. The procedure begins by detecting "seed regions" using an initial threshold on distance. Each seed region is then represented by the mean vector of its components. A small threshold value is used in the beginning, which, as expected, results in over-splitting of less uniform regions. In subsequent iterations, however, the threshold values are allowed to increase depending on a measure of uniformity of each region. This relaxation of threshold value allows the algorithm to recover from possible over-fragmentation. The procedure is stopped after a prespecified number of iterations (about 20). The authors give only three segmentation examples, but do not discuss the effect of different threshold values.

Malik and Perona [65, 66] have proposed a texture segmentation algorithm that also uses a bank of linear filters. As the functional form for the filters (channels), Malik and Perona choose the Gaussian derivative model proposed by Young [95]. These functional forms are shown by Young to be good fits to cortical receptive field profiles. Both radially symmetric difference of Gaussians (DOG) filters, and directionally tuned difference of offset Gaussians (DOOG) filters are used. DOG filters are assumed to model non-oriented simple cells, while DOOG filters model bar-sensitive simple cells. Following the filtering operation each filtered image is half-wave rectified to obtain a set of "neural" responses. These responses are smoothed using spatial averaging. They also use a nonlinear inhibition stage to model the "intracortical inhibition". Texture boundaries are then detected by combining the responses of the Canny edge detector [10] applied to the resulting images.

[4]We note that the authors use two different feature vectors at each grid point, which play different roles in computing the distance between two grid points. However, for simplicity, we will refer to *a* feature vector for each point.

An important advantage of the multi-channel filtering approach, as seen in the above examples, is that one can use simple statistics of gray values in the filtered images as texture features. This simplicity is the direct result of decomposing the original image into several filtered images with limited spectral information. In contrast, texture features that are based on the statistics of the gray-level distribution in the given image, such as gray-level co-occurrence features [43], are usually very complicated and also lack physical interpretation. As an example, consider an application where rotation invariant texture features are needed. In the multi-channel filtering approach, such features can be obtained using the isotropic frequency-selective filters of Coggins and Jain [20]. (See Figure 2.3). Most other techniques for extracting rotation invariant features, such as that proposed by Kashyap and Khotanzad [58] which uses a "circular symmetric autoregressive model", are less intuitive and require more complicated operations.

2.4 Summary

In this chapter, we discussed biological motivations as well as analytical considerations for the multi-channel filtering approach to texture analysis. In texture analysis, a decomposition of the textured image based on frequency (size) and orientation is intuitively appealing, because size and orientation are strong properties of most natural and artificial textures. Furthermore, these properties are general enough to allow discriminating a large number of textures.

We emphasized the interpretation of multi-channel filtering as localized frequency estimation, and its relationship to combined space-frequency representations. We also discussed the importance of joint localization in the space and spatial-frequency domains, in the context of texture segmentation. Accurate localization of the texture boundaries calls for using filters with smaller width at higher frequency channels. Also, for a given width in the spatial domain, a two-dimensional Gabor filter has the smallest possible bandwidth in the spatial-frequency domain. These arguments favored a wavelet transform (decomposition) interpretation of the multi-channel filtering operations, with the Gabor function as the wavelet.

We identified the main issues involved in the multi-channel filtering approach to texture segmentation and presented a survey of the existing techniques. One limitation of these techniques is the lack of a systematic method for determining appropriate filter parameters. Furthermore, only limited segmentation results have been provided in the literature. Bovik *et al.* [6], for example, apply their segmentation technique only to images containing at most two textures. In Chapter 3, we address these limitations and propose a new multi-channel filtering technique that uses a bank of even-symmetric Gabor filters to model the channels.

Chapter 3

Texture Segmentation Using Gabor Filters

In this chapter, we present a multi-channel filtering technique for texture segmentation that uses a bank of Gabor filters to characterize the channels. Figure 3.1 shows an overview of the texture segmentation algorithm. The organization of this chapter is as follows. The choice of the parameters of the Gabor filters in the initial filter set and a systematic filter selection scheme are described in Section 3.1. In Section 3.2, we describe how texture features are computed from filtered images. Section 3.3 describes the process of integrating the feature images to obtain an *unsupervised* segmentation. Supervised texture segmentation experiments using a feed-forward neural network and several other classifiers are reported in Section 3.4. Section 3.5 concludes with a summary and a general discussion.

3.1 Characterizing the Channels

In our texture segmentation algorithm, we represent the channels with a bank of two-dimensional Gabor filters. The spatial and spatial-frequency domain representations of a 'canonical' Gabor filter were given in (2.9) and (2.10). Psychophysical and psychophysiological studies of biological visual systems have provided us with some clues for appropriate bandwidth of the channels. However, the choice of the radial frequencies and .the amount of overlap between the channels remains unclear. Like Turner [90] and Perry and Lowe [77], we model the channels with a *fixed* set of Gabor filters. However, our choice of filter parameters results in a filter set that preserves almost all the information in the input image.

3.1.1 Choice of Filter Parameters

Our filter set consists of even-symmetric Gabor filters. In the spatial-frequency domain, these filters are completely specified by their MTF (see (2.10)). In addition to

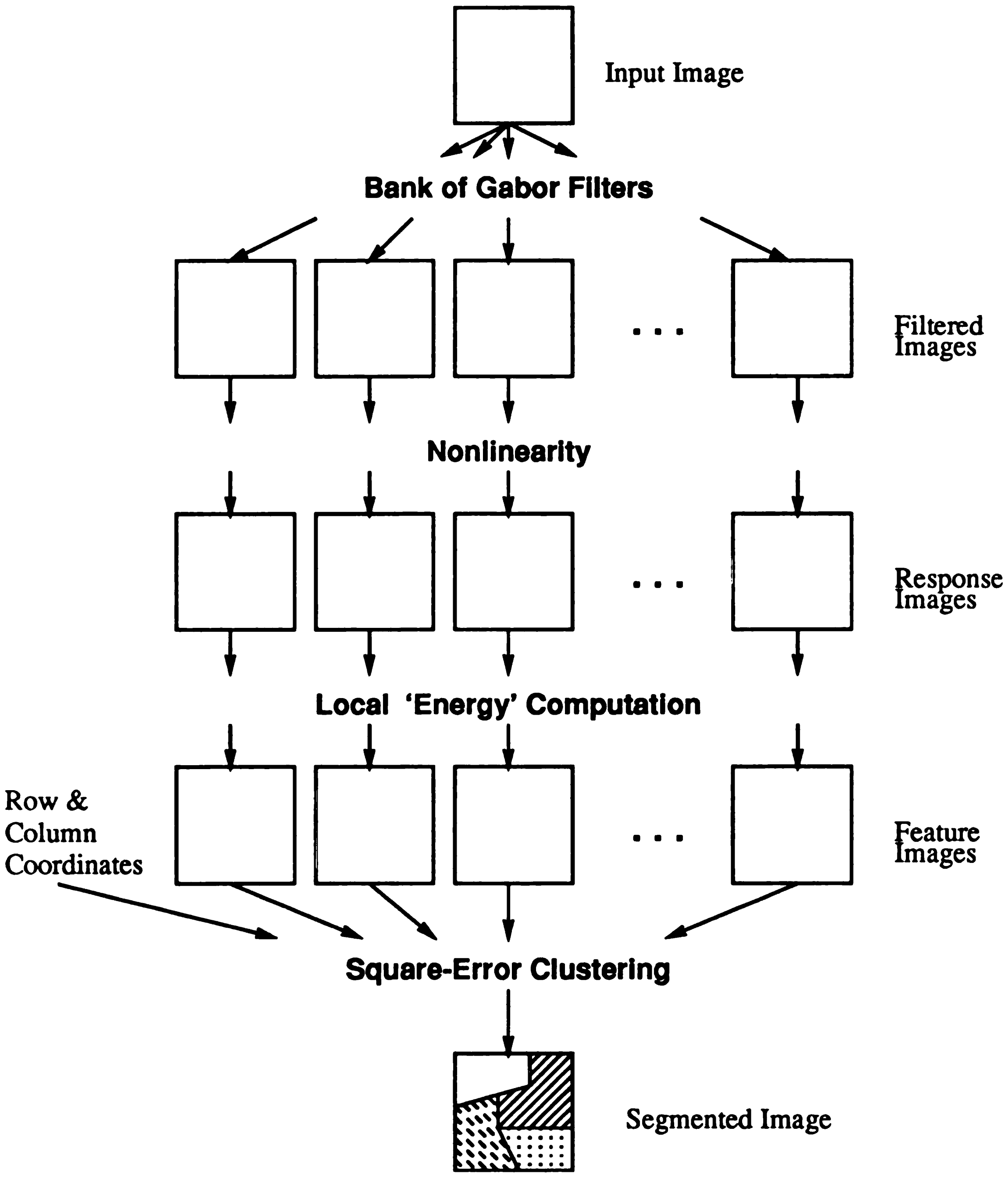


Figure 3.1: An overview of the texture segmentation algorithm.

radial frequency and orientation, the frequency bandwidth B_r and orientation bandwidth B_θ of a spatial filter are also of interest. For the Gabor filter defined by (2.10), the half-peak magnitude bandwidths are given by

$$B_r = \log_2\left(\frac{u_0 + (2\ln 2)^{1/2}\sigma_u}{u_0 - (2\ln 2)^{1/2}\sigma_u}\right), \text{and} \tag{3.1}$$

$$B_\theta = 2\tan^{-1}\left(\frac{(2\ln 2)^{1/2}\sigma_v}{u_0}\right), \tag{3.2}$$

where B_r is in octaves and B_θ is in degrees. (The frequency bandwidth, in octaves, from frequency f_1 to frequency f_2 is given by $\log_2(f_2/f_1)$.)

We implement each even-symmetric Gabor filter by direct sampling of the MTF in (2.10). Details of the implementation are provided in Appendix A. We use four values of orientation θ_0: 0°, 45°, 90°, and 135°. For an image array with a width of N_c pixels, where N_c is a power of 2, the following values of radial frequency u_0 are used:

$$1\sqrt{2},\ 2\sqrt{2},\ 4\sqrt{2},\ \cdots,\ \text{and}\ (N_c/4)\sqrt{2} \qquad \text{cycles/image-width}$$

Note that the radial frequencies are 1 octave apart. The above choice of radial frequencies guarantees that the passband of the filter with the highest radial frequency, *viz.* $(N_c/4)\sqrt{2}$ cycles/image-width, falls inside the image array[1]. We let the orientation and frequency bandwidths of each filter be 45° and 1 octave, respectively. Several experiments have shown that the frequency bandwidth of simple cells in the visual cortex is about 1 octave [78]. Figure 3.2 shows the filter set used for segmenting 256 × 256 images.

Psychophysical experiments show that the resolution of the orientation tuning ability of the human visual system is as high as 5°. Therefore, in general, finer quantization of orientation will be needed. The restriction to four orientations is made for computational efficiency in the current implementation of the algorithm, and is sufficient for discriminating many textures. The total number of Gabor filters in the filter set is given by $4\log_2(N_c/2)$. For an image with 256 columns, for example, a total of 28 filters can be used — 4 orientations and 7 radial frequencies. For some textures, however, filters with low radial frequencies (e.g., $1\sqrt{2}$ and $2\sqrt{2}$ cycles/image-width) are not very useful, because these filters capture spatial variations that are

[1]In psychophysics, frequencies are expressed in cycles per degree of visual angle subtended on the eye. The frequencies in cycles/image-width can be converted to cycles/degree if the width of the image in degrees of visual angle is known.

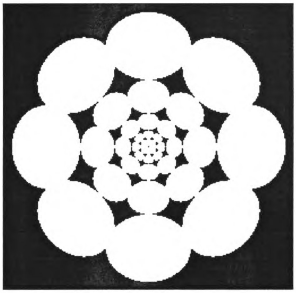


Figure 3.2: The filter set in the spatial-frequency domain (256×256). There are a total of 28 Gabor filters. Only the half-peak support of the filters is shown.

too large to explain textural variations in an image. Therefore, we do not use these filters, in the texture segmentation experiments.

In order to assure that the filters do not respond to regions with constant intensity, we have set the MTF of each filter at $(u, v) = (0, 0)$ to zero. As a result each filtered image has a mean of zero. Furthermore, the FFT algorithm that is used to perform the convolutions requires that the dimensions of the input image be powers of two. When this requirement is not met, the input image can be padded by zeros to obtain a rectangular image with appropriate dimensions.

The set of filters used in the algorithm results in nearly uniform coverage of the spatial-frequency domain (Figure 3.2). A decomposition obtained by the filter set is nearly orthogonal, as the amount of overlap between the filters (in the spatial-frequency domain) is small. One way to demonstrate this property is through reconstruction of an image from the filtered images. Figure 3.3 shows two 128×128 images and their reconstructed versions. The original images are shown in the top row. The reconstructed images, obtained by adding all 24 filtered images are in the bottom row. After adding all the filtered images, the gray values were linearly mapped to 0 – 255 interval.

From a signal analysis point of view, our filter set constitutes an approximate

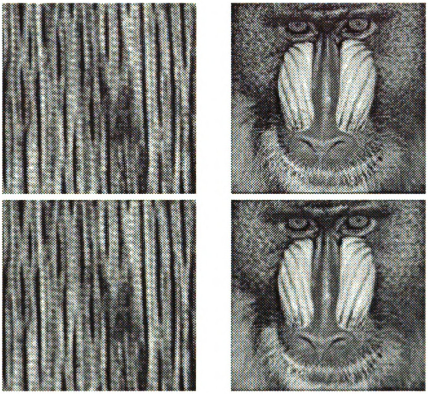

(a) (b)

Figure 3.3: Examples demonstrating the advantage of nearly uniform coverage of the spatial-frequency domain by the filter set. (a) 'Wood grain' (D68) from the Brodatz album [7]. (b) 'Mandrill'. Top row: original images. Bottom row: reconstructed images. Both images are 128×128.

orthogonal basis for a wavelet transform, with the Gabor function as the wavelet. (See Section 2.2.) Intuitively, a wavelet transform can be interpreted as a band-pass filtering operation on the input image. The Gabor function is an admissible wavelet; however, it does not result in an orthogonal decomposition. This means that a wavelet transform based on the Gabor wavelet is redundant [67]. The filtering operations using the filter set can be interpreted as computing the wavelet transform of the input image at selected spatial-frequencies (frequency and orientation pairs). The ability to reconstruct good approximations of the input image from the filtered images demonstrates that the filter set forms an almost complete basis for the wavelet transform.

Figure 3.4 shows examples of filtered images for an image containing 'straw matting' (D55) and 'wood grain' (D68) textures from the photographic album of textures by Brodatz [7]. To maximize visibility, each filtered image has been scaled to full contrast. (Note that this scaling does not affect the relative differences in the strength of the responses in different regions.) The ability of the filters to exploit differences in frequency (size) and orientation in the two textures is evident in these images. The differences in the strength of the responses in regions with different textures is the key to the multi-channel approach to texture analysis.

3.1.2 Filter Selection

We now describe a systematic filter selection scheme which is based on an intuitive least squares error criterion. Using only a subset of the filtered images can reduce the computational burden at later stages, because this directly translates into a reduction in the number of texture features.

Let $s(x,y)$ be the reconstruction of the input image obtained by adding all the filtered images. (We have demonstrated that $s(x,y)$ is a good approximation of the original input image.) Let $\hat{s}(x,y)$ be the *partial* reconstruction of $s(x,y)$, obtained by adding a subset $\mathcal{A}$ of filtered images. That is,

$$\hat{s}(x,y) = \sum_{j\in\mathcal{A}} r_j(x,y)\,, \tag{3.3}$$

where $r_j(x,y)$ is the j^{th} filtered image. The error involved in using $\hat{s}(x,y)$ instead of

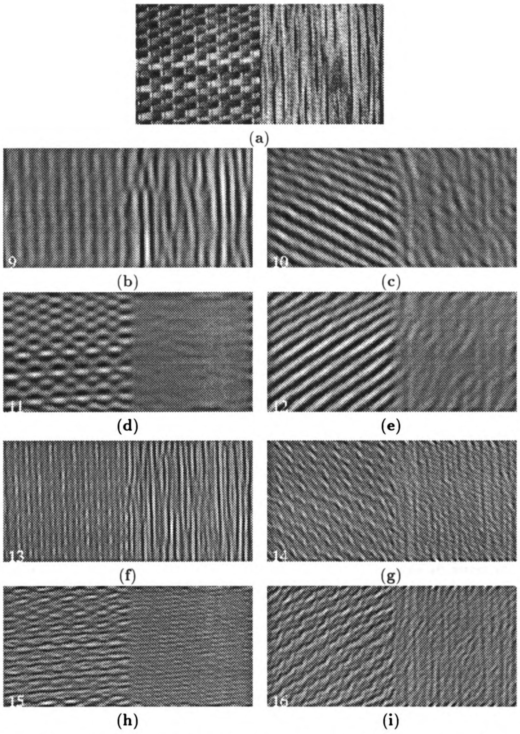


Figure 3.4: Examples of filtered images for the 'D55-D68' texture pair (128×256). (**a**) Input image. (**b–e**) Filtered images corresponding to Gabor filters tuned to $16\sqrt{2}$ cycles/image-width. (**f–i**) Filtered images corresponding to Gabor filters tuned to $32\sqrt{2}$ cycles/image-width. All four orientations — $0°, 45°, 90°$, and $135°$ — for each frequency are shown.

$s(x,y)$ can be measured by

$$SSE = \sum_{x,y} [\hat{s}(x,y) - s(x,y)]^2. \tag{3.4}$$

The fraction of intensity variations in $s(x,y)$ that is explained by $\hat{s}(x,y)$ can be measured by the ***coefficient of determination***[2] (COD)

$$R^2 = 1 - \frac{SSE}{SSTOT}, \tag{3.5}$$

where

$$SSTOT = \sum_{x,y} [s(x,y)]^2. \tag{3.6}$$

Note that $s(x,y)$ has a mean of zero, since the mean gray value of each filtered image is zero.

The motivation behind the filter selection scheme is to use only a subset of filtered images that together explain a "significant" portion of the intensity variations in $s(x,y)$. We determine the "best" subset of the filtered images (filters) by the following sequential forward selection procedure [30]:

1. Select the filtered image that best approximates $s(x,y)$, i.e. results in the highest R^2 value.

2. Select the next filtered image that together with previously selected filtered image(s) best approximate $s(x,y)$.

3. Repeat Step 2 until $R^2 \geq 0.95$.

Since adding all filtered images gives $s(x,y)$, the value of R^2 when all filters are used is 1.0. A minimum value of 0.95 for R^2 means that we will use only as many filtered images as necessary to account for at least 95% of the intensity variations in $s(x,y)$. Note that the above sequential forward selection scheme is not optimal. Determining the best subsets of filtered images requires examination of all possible subsets of all possible sizes. An exhaustive search, however, is computationally prohibitive.

An important point to bear in mind is that the least squares error criterion in (3.5) only reflects convergence in the mean of $s(x,y)$ and $\hat{s}(x,y)$. A large R^2 value, therefore, does not necessarily guarantee a good fit at every point. If there are texture categories in the input image that occupy very small portions of the

[2]The terminology used here is borrowed from linear regression analysis.

image, it is recommended that a larger minimum value for R^2 (e.g., 0.99) be used. Figure 3.5 illustrates the filter selection results for the 'D55-D68' texture pair shown in Figure 3.4(a). Based on the forward selection procedure, only 13 filters, out of a total of 20, explain more than 95% of the intensity variations.

Approximate Method

Again, let $r_j(x,y)$ be the j^{th} filtered image, and $R_j(u,v)$ be its discrete Fourier transform (DFT). The amount of overlap between the MTFs of the Gabor filters in the filter set is small. (See Figure 3.2.) Therefore, the total energy E in $s(x,y)$ can be approximated by

$$E \approx \sum_{j=1}^{n} E_j, \tag{3.7}$$

where

$$E_j = \sum_{x,y} [r_j(x,y)]^2 = \sum_{u,v} |R_j(u,v)|^2. \tag{3.8}$$

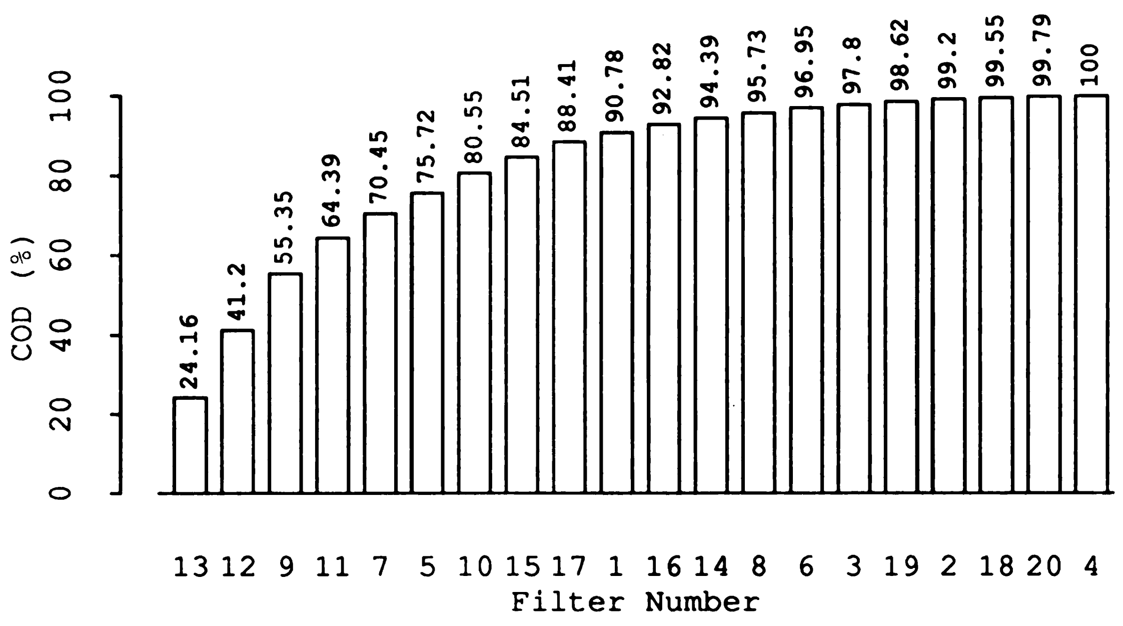
and n is the total number of filters (typically 20). Now, it is easily verified that for any subset $\mathcal{A}$ of filtered images,

$$R^2 \approx \frac{\sum_{j\in\mathcal{A}} E_j}{E}. \tag{3.9}$$

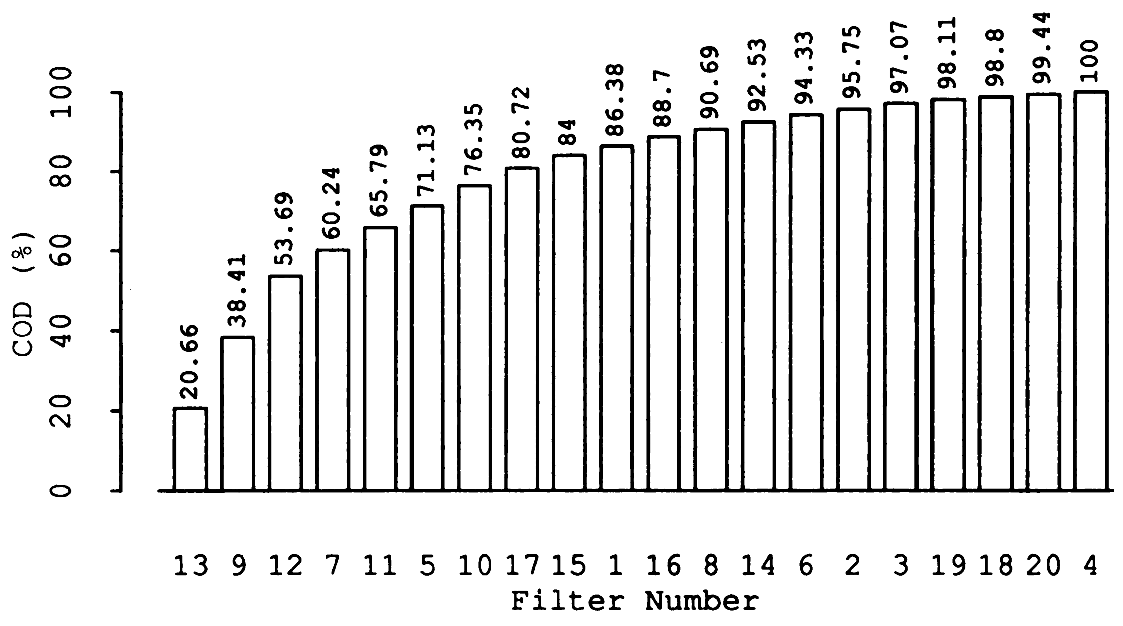
An *approximate* filter selection then consist of computing E_j for $j = 1, \cdots, n$. These energies can be computed in the Fourier domain, hence avoiding unnecessary inverse DFTs. We then sort the filters (channels) based on their energy and pick as many filters as needed to achieve $R^2 \geq 0.95$. Computationally, this procedure is much more efficient than the sequential forward selection procedure described before. The inclusion of filters (channels) with higher energy is intuitively appealing. On the other hand, if an input image does not contain frequency components that fall in the passband of a Gabor filter, then that filter will not be very useful for discriminating the textures in the image.

3.2 Computing Feature Images

An important goal of the research in texture analysis is to develop a set of texture measures (features) that can successfully discriminate arbitrary textures. Here, we



(a)



(b)

Figure 3.5: (**a**) Filter selection by reconstruction for the 'D55-D68' texture pair. Note that 13 filters alone, out of a total of 20, account for at least 95% of intensity variations in the original textured image. (**b**) Filter selection by approximate method. This method calls for using 15 filters, which include the 13 filters in (**a**).

present one such set which captures the attributes of "blobs" detected in the Gabor filtered images.

Psychophysical studies of texture perception suggest that preattentive texture discrimination may be explained by differences in the attributes of a few conspicuous local features, called textons [55]. Some features identified as textons include elongated blobs (e.g., rectangles, ellipses, line segments) with specific colors, angular orientations, widths, and lengths; line segment terminators (end-of-lines); and line segment crossings. Julesz and his co-investigators [54, 55] demonstrate the ability of textons to predict and to explain texture discrimination in numerous artificially generated texture pairs. The main criticism of the texton theory has been that it does not describe how textons are extracted from natural (grayscale) textured images.

Voorhees and Poggio [93] have proposed an algorithm for extracting blob textons from grayscale images by using a Laplacian of Gaussian (LOG) operator followed by thresholding at a small positive value and using morphological operations. Differences in the statistical distribution of attributes of the blobs, such as contrast, orientation, width, length, area, and area density, in small windows are then used to detect texture boundaries. Tuceryan and Jain [89] have used a similar approach to extract texture primitives that they call "tokens". Voorhees and Poggio contend that line segment crossings and terminators are not textons, and that texture discrimination can be explained using blob textons only. As recognized by the authors, their segmentation algorithm does not address the problem of determining the appropriate scale(s) for detecting the blobs. As we will see, the multi-resolution nature of our segmentation algorithm offers one possible solution to this problem. Furthermore, the computation of texture features in our approach does not require extraction of explicit texture primitives such as textons or tokens.

Several investigators have speculated on the possible relationship between Gabor filters and texton detection [18, 90]. However, no clear procedure has been set forth that describes how Gabor filters act as texton detectors or how texton attributes are captured by them. Our feature extraction scheme, which involves a nonlinear stage, provides a more clear explanation of the purported role of Gabor filters as blob detectors. Some of the experiments (see Section 3.3.3) support the position taken by Voorhees and Poggio [93] that differences in the attributes of blob textons alone can explain texture discrimination.

We use the following procedure to compute features from each filtered image.

First, each filtered image is subjected to a nonlinear transformation. Specifically, we use the following bounded nonlinearity

$$\psi(t) = \tanh(\alpha t) = \frac{1 - e^{-2\alpha t}}{1 + e^{-2\alpha t}}, \tag{3.10}$$

where α is a constant. This nonlinearity is similar to the sigmoidal activation function used in artificial neural networks [63]. In our experiments, we have used an empirical value of $\alpha = 0.25$ which results in a rapidly saturating, threshold-like transformation. As a result, the application of the nonlinearity transforms the sinusoidal modulations in the filtered images to square modulations and, therefore, can be interpreted as a blob detector. However, the detected blobs are not binary, and unlike the blobs detected by Voorhees and Poggio [93] they are not necessarily isolated from each other. Also, since each filtered image has a zero mean and the nonlinearity in (3.10) is odd-symmetric, both dark and light blobs are detected.

Instead of identifying individual blobs and *then* measuring their attributes, we capture their attributes by computing the average absolute deviation (AAD) from the mean value in a small window around each pixel in the 'response images' (at the output of nonlinear stages). This is similar to the 'texture energy' measure that was first proposed by Laws [61]. Formally, the feature image $e_j(x,y)$ corresponding to the filtered image $r_j(x,y)$ is given by

$$e_j(x,y) = \frac{1}{M^2} \sum_{(a,b)\in W_{xy}} |\psi(r_j(a,b))|, \tag{3.11}$$

where $\psi(\cdot)$ is the nonlinear function in (3.10) and W_{xy} is an $M \times M$ window centered at the pixel with coordinates (x,y).

The size, M, of the averaging window in (3.11) is an important parameter. More reliable measurement of texture features calls for larger window sizes. On the other hand, more accurate localization of region boundaries calls for smaller windows. This is because averaging blurs the boundaries between textured regions. Furthermore, using Gaussian weighted windows, rather than unweighted windows, will minimize distortions due to the Gibbs phenomenon. Gaussian-weighted windows are also likely to result in more accurate localization of texture boundaries. Therefore, for each filtered image we use a Gaussian window whose space constant σ is proportional to the *average* size of the intensity variations in the image. For a Gabor filter with center radial frequency u_0 this average size is given by

$$T = N_c/u_0 \qquad \text{pixels}, \tag{3.12}$$

where N_c is the width (number of columns) of the image.

We found $\sigma = 0.5\sqrt{2}T$ to be appropriate in most of the segmentation experiments. Note that although we use different window sizes for different filtered images, they are all specified by a single parameter — the proportionality constant. When computing the texture features for pixels near the image boundary we assume that the image is extended by its mirror image — often referred to as even reflection boundary condition [22]. Figure 3.6 shows feature images corresponding to filtered images shown in Figure 3.4.

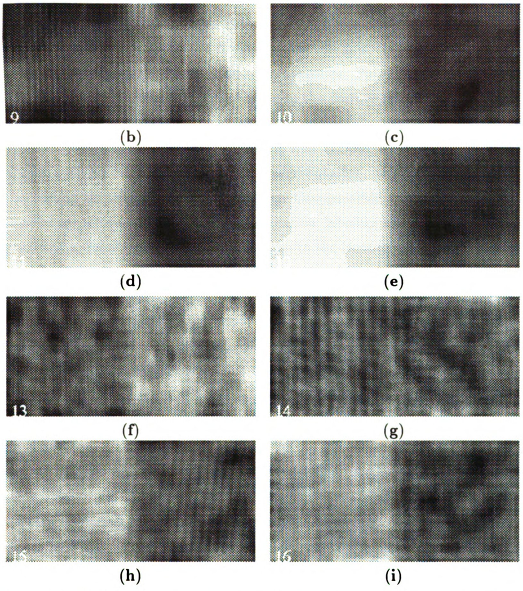


Figure 3.6: Feature images corresponding to filtered images in Figure 3.4. A $\sigma = 0.5\sqrt{2}T$ was used for the Gaussian averaging windows.

3.3 Unsupervised Texture Segmentation

Having obtained the feature images, the main question is how to integrate features corresponding to different filters to produce a segmentation. Let's assume that there are K texture categories, $\mathcal{C}_1, \ldots, \mathcal{C}_K$, present in the image. If our texture features are capable of discriminating these categories then the patterns belonging to each category will form a cluster in the feature space which is "compact" and "isolated" from clusters corresponding to other texture categories. Pattern clustering algorithms are ideal vehicles for *recovering* such clusters in the feature space.

A segmentation algorithm based on clustering pixels using their associated feature vectors alone suffers from an important shortcoming — it does not utilize the spatial (contextual) information. In texture segmentation, neighboring pixels are very likely to belong to the same texture category. One possible approach to incorporate this contextual information is to use a relaxation labeling technique. That is, first obtain an initial labeling by clustering patterns in the feature space, and then enforce the spatial constraints using relaxation [47]. Instead, we propose a simple method that incorporates the spatial adjacency information directly in the clustering process. This is achieved by including the spatial coordinates of the pixels as two additional features (see Figure 3.1). The spatial coordinates of pixels have been used by Hoffman and Jain [46] for segmentation of range images. The inclusion of spatial coordinates in the computation of the distance between feature vectors encourages neighboring pixels to cluster together. As a result, over-fragmentation of otherwise uniform texture regions is avoided.

In our texture segmentation experiments we have used a square-error clustering algorithm known as CLUSTER [49]. The algorithm iterates through two phases. Phase 1 (the K-means pass) creates a sequence of clusterings containing $2, 3, \cdots, k_{max}$ clusters, where k_{max} is specified by the user. Phase 2 (the forcing pass) then creates another set of clusterings by merging existing clusters two at a time to see if a better clustering can be obtained. After each pass through phase 1 and phase 2, the square errors of the clusterings are compared with the square errors of the clusterings that existed before that pass. (Each new clustering is compared with the old clustering having the same number of clusters.) If any of the square errors are smaller than before, another pass through phases 1 and 2 is initiated. This continues until the square error cannot be decreased.

3.3.1 How Many Categories?

Determining the number of texture categories that are present in an image is a difficult problem. Relative indices provide a means of comparing clusterings with different number of clusters and deciding which clustering is "best". In our segmentation algorithm we rely on the modified Hubert (MH) index, proposed by Dubes [32]. For a given clustering, the MH index is computed as follows. Let $L(i)$ be the label function

$$L(i) = l \qquad \text{if pattern } i \text{ is in cluster } l,$$

and $d_{p,q}$ the Euclidean distance between cluster centers p and q. Define

$$Y(i,j) = d_{L(i),L(j)}.$$

The (normalized) MH index is then given by:

$$\mathrm{MH} = \left\{ \frac{1}{M} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} [X(i,j) - m_x]\,[Y(i,j) - m_y] \right\} / s_x\, s_y\,, \tag{3.13}$$

where $X(i,j)$ is the Euclidean distance between patterns i and j, n is the total number of patterns, $M = n(n-1)/2$, and

$$\begin{aligned} m_x &= \frac{1}{M} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X(i,j) & m_y &= \frac{1}{M} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Y(i,j) \\ s_x^2 &= \frac{1}{M} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X^2(i,j) - m_x^2 & s_y^2 &= \frac{1}{M} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Y^2(i,j) - m_y^2. \end{aligned}$$

The MH index is, therefore, the point serial correlation between the entries of X and Y matrices. Unusually large values of MH suggest that corresponding entries in the two matrices are close to each other. Intuitively, the cluster centers are assumed to be the true representation of the texture categories, and any deviations from the centers are assumed to be due to errors in measurements and distortions. Note that the MH index will be 1 for the trivial clustering in which each pattern is an individual cluster and is not defined for the clustering in which all patterns are in the same cluster.

The "true" number of texture categories is estimated as follows. First a sequence of clusterings is obtained using the CLUSTER algorithm. We assume k_{max} is known, or can be reliably estimated, and plot MH(k) for $k = 2, \cdots, k_{max}$. When the data contain a strong clustering, MH(k) first increases and then levels off, and a "significant" knee is formed at the true number of clusters. The following intuitive justification for this behavior is suggested by Dubes [32]. Suppose the true number of clusters is k^*. The clusterings with $k > k^*$ will then be formed by breaking the true clusters into smaller ones. As a result, the correlation between the entries of X and Y matrices will be high. The clusterings with $k < k^*$ clusters, however, will be formed by merging the true clusters, hence reducing the correlation. Therefore, assuming that our texture features provide strong discrimination between different texture categories, we should see a significant knee in the plot of MH(k) at the true value of k.

A major difficulty with clustering indices is that it is hard to determine the significance of an observed index. In our segmentation experiments, the significant knee in the plot of the MH(k) is determined visually. When such a knee is hard to identify, we will simply assume that the "true" number of texture categories is known *a priori*. In Chapter 4, we will propose an alternative, integrated approach to eliminate the need for knowing the true number of texture categories.

Some implementation details must be explained. Prior to clustering we normalize each feature to have a mean of zero and a constant variance ($= 10.0$). When used as additional features, the row and column coordinates are normalized in the same way. (Feature images with very small variances ($< 10^{-4}$) are simply discarded.) This normalization is intended to avoid domination of features with small numerical ranges by those with larger ranges[3]. Clustering a large number of patterns becomes computationally demanding. The following two-step grouping of pixels is, therefore, adopted for computational efficiency. First, we cluster a small randomly selected subset of patterns into a specified number of clusters. Patterns in each cluster are given a generic category label that distinguishes them from those in other clusters. These labeled patterns are then used as training patterns to classify patterns (pixels) in the entire image using a minimum distance classifier.

3.3.2 Performance Evaluation

The lack of appropriate quantitative measures of the goodness of a segmentation makes it very difficult to evaluate and compare different texture segmentation algorithms. A simple criterion that is often used is the percentage of misclassified pixels.

[3]For a study of standardization strategies in cluster analysis, see the recent article by Milligan and Cooper [73].

This criterion, however, has certain disadvantages. For example, it often does not reflect the ability of the algorithm to accurately locate the boundaries. By changing the locations of misclassified pixels in the vicinity of a boundary we can make the boundary "look" less (or more) accurate, and still have the same percentage of misclassified pixels. Despite such drawbacks, we use this simple criterion, because it is the only general and practical criterion that is currently available.

3.3.3 Experimental Results

We now apply our texture segmentation algorithm to several images in order to demonstrate its performance. These images are created by collaging subimages of natural as well as artificial textures. We start with a total of 20 Gabor filters in each case. Each filter is tuned to one of the four orientations and one of the five highest radial frequencies. For an image with a width of 256 pixels, for example, $4\sqrt{2}, 8\sqrt{2}, 16\sqrt{2}, 32\sqrt{2}$, and $64\sqrt{2}$ cycles/image-width radial frequencies are used. We then use our filter selection scheme to determine a subset of filtered images that achieves an R^2 value of at least 0.95 (see Section 3.1.2).

The number of randomly selected feature vectors that are used as input to the clustering program is proportional to the size of the input image. For a 256×256 image, for example, 4000 patterns are selected at random, which is about 6% of the total number of patterns. This percentage is used in all the following experiments. The segmentation results are displayed as gray-level images, where regions belonging to different categories are shown with different gray levels.

Figure 3.7 shows the segmentation results for the 'D55-D68' texture pair. Only 13 Gabor filters (texture features) are used. As seen in the two-category segmentation, the two textures are successfully discriminated and the boundary between them is detected quite accurately. The segmentation with pixel coordinates included as additional features was essentially the same and is not shown here. The plot of the modified Hubert index versus number of texture categories for this image is shown in Figure 3.8. The curve levels off at $k = 2$, with MH(2) ≈ 0.9 — strong evidence for the two-category segmentation.

Figure 3.9(a) shows a 256×256 image ('GMRF-4') containing four Gaussian Markov random field (GMRF) textures. These textures have been generated using non-causal finite lattice GMRFs [14] and can not be discriminated on the basis of their

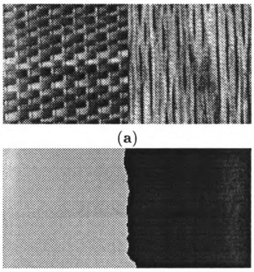


Figure 3.7: (a) The 'D55-D68' texture pair. (b) Two-category segmentation obtained using a total of 13 Gabor filters.

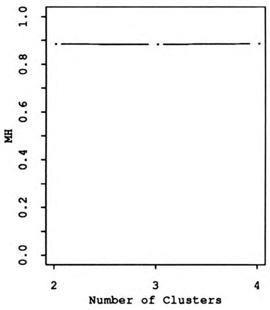


Figure 3.8: The plot of the MH index versus number of clusters (texture categories) for the 'D55-D68' texture pair.

mean gray value. In Figure 3.9 we show the segmentation results for this image. The difference between segmentations in Figures 3.9(b) and (c) shows the improvement due to inclusion of pixel coordinates as additional features in the clustering algorithm. The plot of MH(k) is shown in Figure 3.10. The curve levels off at $k = 4$, following a "significant" knee. The high value of MH index at $k = 4$ (≈ 0.80) also strongly supports the four-category segmentation.

Figure 3.11(a) shows another 256×256 image ('Nat-5') containing natural textures D77, D55, D84, D17, and D24 from the Brodatz album. Only 13 Gabor filters, out of a total of 20, are used. The five-category segmentation of this image is shown in Figure 3.11(b). As seen in Figure 3.12, the plot of MH(k) is not helpful for deciding the true number categories.

Figures 3.13 and 3.14 summarize the segmentation results for a 512×512 image ('Nat-16') containing sixteen natural textures, also from the Brodatz album. Again, it is difficult, if not impossible, to decide the true number of texture categories using the plot of MH(k). Nonetheless, assuming that we know the true number of texture categories, we have shown the 16-category segmentation.

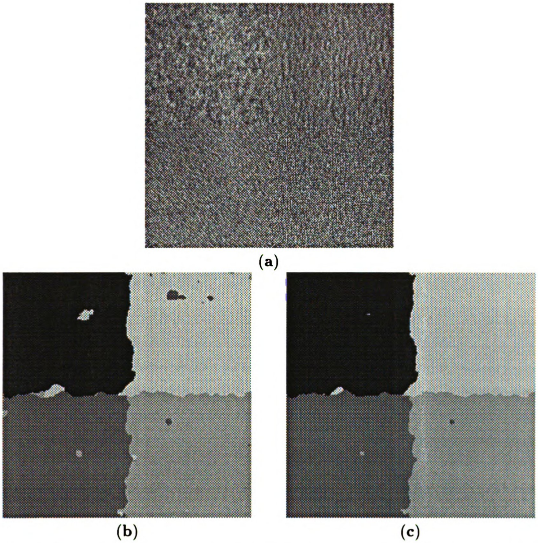


Figure 3.9: (**a**) A 256 × 256 image ('GMRF-4') containing four Gaussian Markov random field textures. (**b**) Four-category segmentation obtained using a total of 11 Gabor filters. (**c**) Same as (**b**), but with pixel coordinates used as additional features.

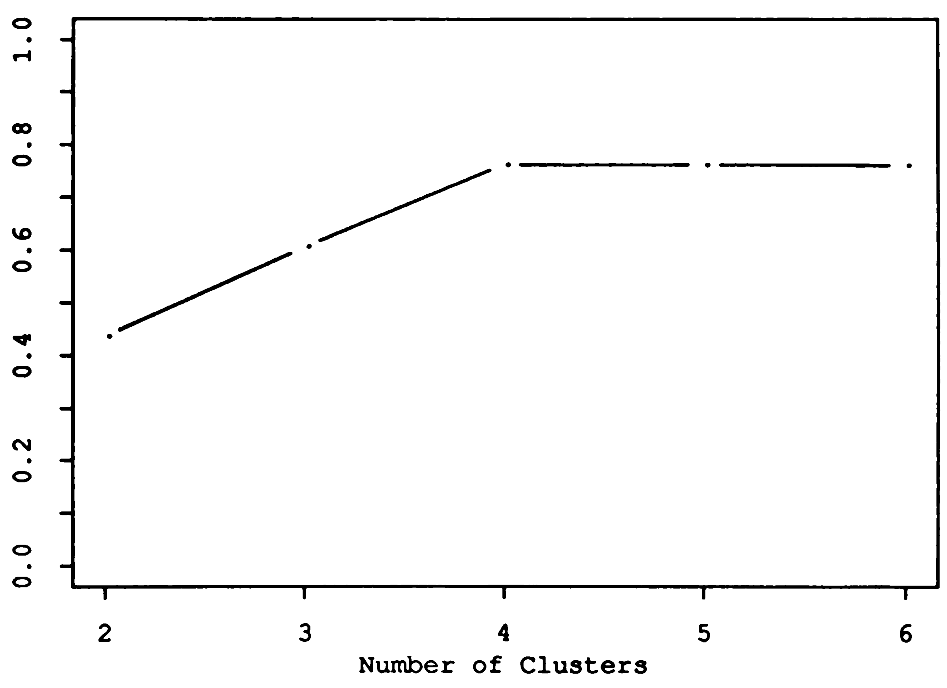


Figure 3.10: The plot of the MH index versus number of texture categories for the 'GMRF-4' image shown in Figure 3.9(a).

The filter selection (with a threshold of 0.95 for R^2) indicated that only 14 filtered images are sufficient. However, the resulting segmentations were not very good. The 16-category segmentation in Figure 3.13(b) is obtained using all 20 filtered images (and the pixel coordinates). Compared to previous examples where each texture category constituted about 1/2 or 1/4 of the image, in this example each category occupies only 1/16 of the image. Recall that the fitting criterion in our filter selection scheme is computed globally over the entire image. A larger threshold for R^2 should, therefore, be used if any texture category is expected to occupy only a small fraction of the image.

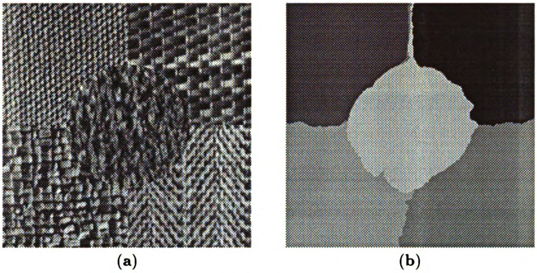


Figure 3.11: (**a**) A 256 × 256 image ('Nat-5') containing five natural textures (D77, D55, D84, D17, and D24) from the Brodatz album. (**b**) Five-category segmentation obtained using a total of 13 Gabor filters and the pixel coordinates.

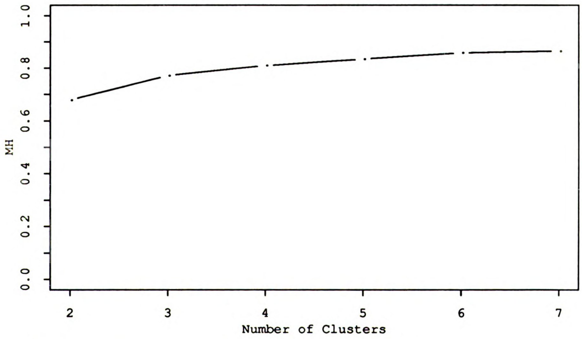
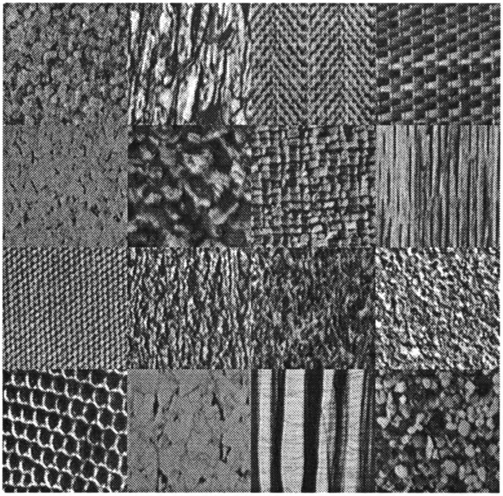


Figure 3.12: The plot of the MH index versus number of texture categories for the 'Nat-5' image shown in Figure 3.11(**a**).



(a)

Figure 3.13: (**a**) A 512 × 512 image ('Nat-16') containing sixteen natural textures (row 1: D29, D12, D17, D55; row 2: D32, D5, D84, D68; row 3: D77, D24, D9, D4; row 4: D3, D33, D51, D54) from the Brodatz album (**b**) 16-category segmentation obtained using a total of 20 Gabor filters and the pixel coordinates.

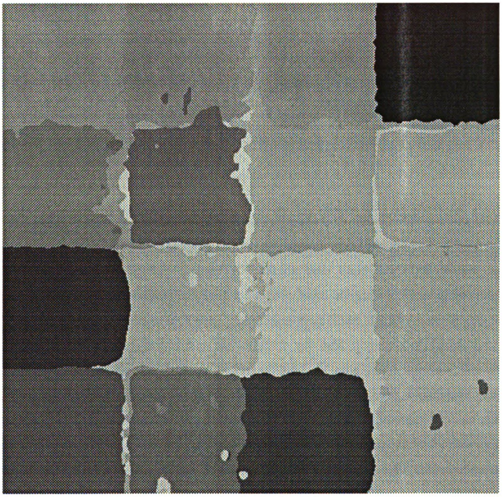

(b)

Figure 3.13: (*cont'd.*)

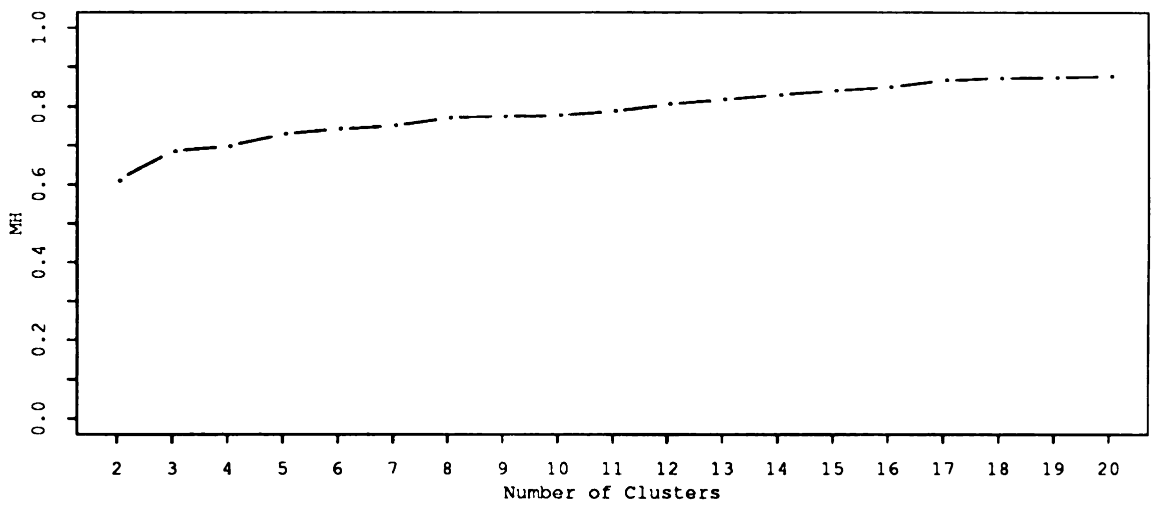


Figure 3.14: The plot of the MH index versus number of texture categories for the 'Nat-16' image shown in Figure 3.13(a).

Figures 3.15 and 3.16 summarize the segmentation results for a number of texture pair images that have been used in the psychophysical studies of texture perception. The two textures in the 'L and +' texture pair have identical power spectra. The textures in the 'Even-Odd' texture pair [57] have identical third-order statistics. (The 'Even-Odd' nomenclature comes from the fact that, in the two textures, any 2×2 neighborhood contains either an even or an odd number of black (or white) pixels.) The textures in the 'Triangle-Arrow' and 'S and IO' texture pairs [57], on the other hand, have identical second-order statistics. The 'Even-Odd' and 'Triangle-Arrow' textures are two counter-examples to the original Julesz conjecture that texture pairs with identical second-order statistics cannot be preattentively discriminated [56]. While the first three texture pairs in Figure 3.15 are easily discriminated, the 'S and IO' texture pair is not *preattentively* discriminable.

Compared to our previous examples, the observed values of the MH indices for these artificially generated texture pairs are low (≈ 0.6). Moreover, while the plot of the MH index for the 'Triangle-Arrow' texture pair levels off at $k = 2$, it is not easy to judge the behavior of the curve for other texture pairs. Nonetheless, assuming that the true number of categories is two, we obtained the two-category segmentation of each image (Figure 3.15). Our algorithm appears to perform as predicted by preattentive texture discrimination by humans — the algorithm successfully segments the first

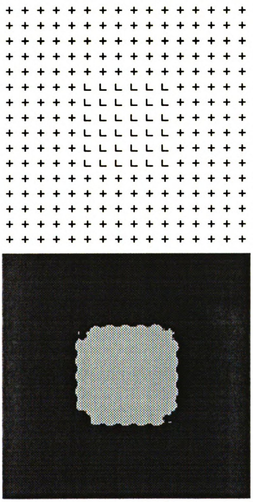
Table 3.1: Percentage of pixels misclassified in the segmentation results.

Input Image			Misclassified (%)	
name	size	# categories	without (r,c)	with (r,c)
D03-D17	128 × 256	2	0.97	0.94
D03-D68	128 × 256	2	1.07	1.04
D17-D77	128 × 256	2	0.89	0.81
D55-D68	128 × 256	2	0.61	0.58
GMRF-4	256 × 256	4	2.68	1.78
Nat-5	256 × 256	5	4.87	2.96
Nat-16	512 × 512	16	12.85	7.47
L and +	256 × 256	2	2.21	2.21
Even-Odd	256 × 256	2	3.05	2.60
Tri-Arr	256 × 256	2	6.12	5.20

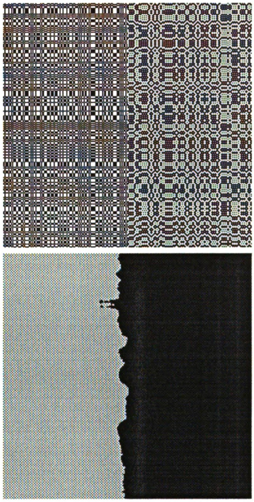
three texture pairs, but fails to do so for the 'S and IO' texture pair.

The texton theory of Julesz associates the preattentive discrimination of the 'Triangle-Arrow' texture pair to the difference in the density of termination points [55]. The successful discrimination of this texture pair by our algorithm supports the position taken by Voorhees and Poggio [93], that differences in the attributes of blobs detected in the filtered versions of textures alone can explain the observed discrimination.

Table 3.1 gives the percentage of misclassified pixels for the segmentation experiments reported here. As seen in this table, there is a clear advantage in using the pixel coordinates (spatial information) as additional features.

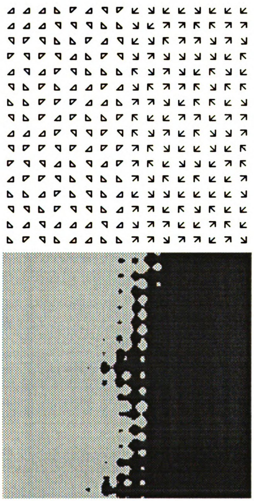


(a)



(b)

Figure 3.15: Segmentation of texture pairs that have been used in the psychophysical studies of texture perception. All images are 256 × 256. The number of Gabor filters used varied between 8 – 11. (**a**) 'L and +'. (**b**) 'Even-Odd'. (**c**) 'Triangle-Arrow'. (**d**) 'S and IO'.


(c)

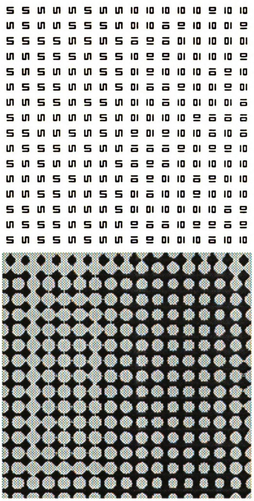

(d)

Figure 3.15: *(cont'd.)*.

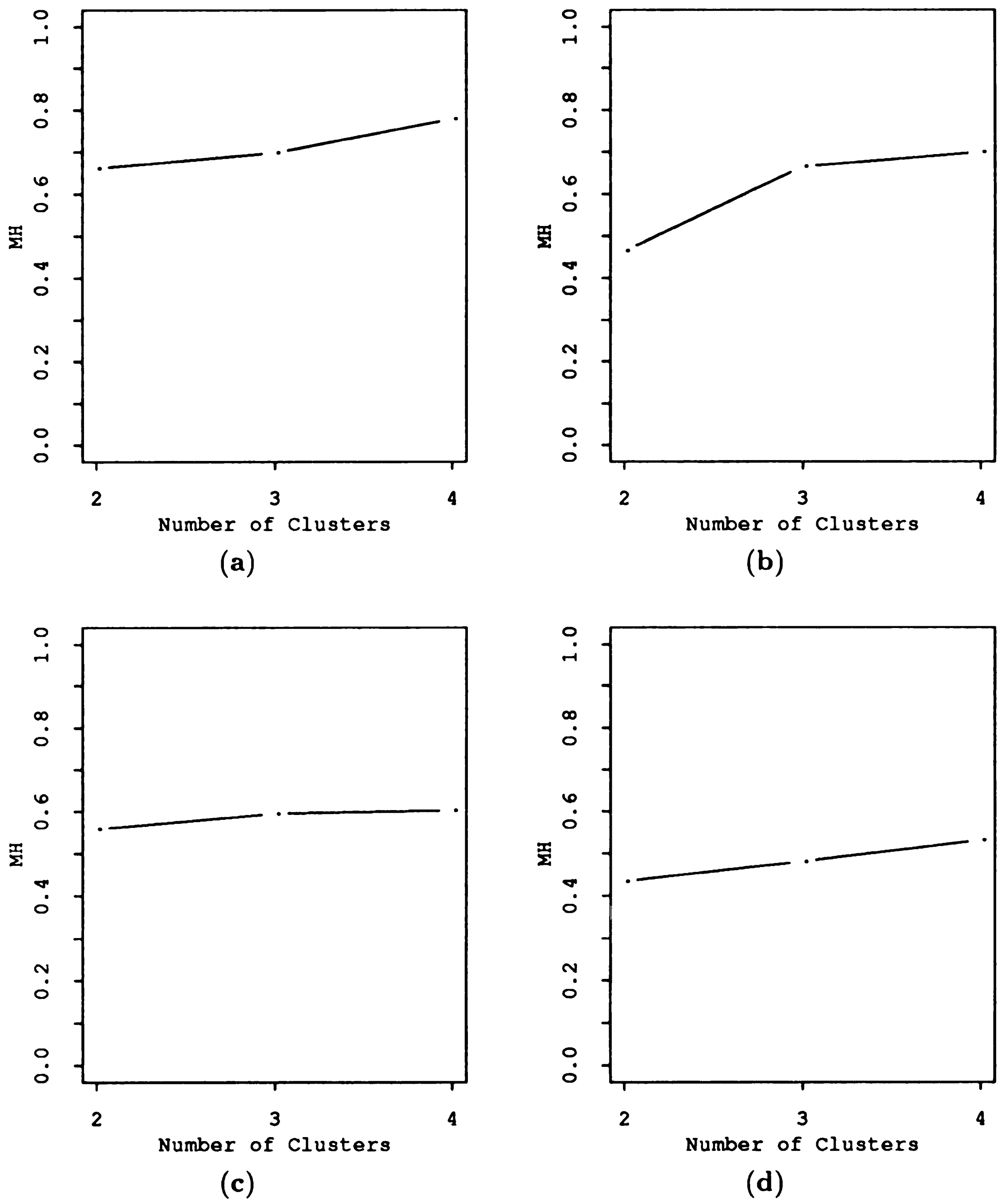


Figure 3.16: The plot of the MH index versus number of texture categories for the texture pair images shown in Figure 3.15. (**a**) The plot for 'L and +'. (**b**) The plot for 'Even-Odd'. (**c**) The plot for 'Triangle-Arrow'. (**d**) The plot for 'S and IO'.

3.4 Supervised Texture Segmentation

In Section 3.3, we assumed that the texture categories were unknown and relied on a clustering algorithm to identify them. In many instances, such as in remote sensing or medical applications, one has access to previously collected data with known categories. In others, data with known category labels can be obtained from the image itself — for example with the help of a human expert. In our texture segmentation technique, when such training data are already available, we can replace the clustering stage with a classifier.

Motivated by the biological plausibility of neural network classifiers, we will use a feed-forward network as our main classifier. To assess the performance of the feed-forward network we compare its performance with classifiers used in pattern recognition literature. Specifically, we use the minimum Euclidean distance, minimum Mahalanobis distance, and k-nearest neighbor (k-NN) classifiers.

In the following supervised texture segmentation experiments, the training data are obtained by randomly sampling the feature images. The randomly sampled patterns make up about 6% of the total number of patterns. (This fraction is the same as that used in Section 3.3 when a small randomly sampled subset of patterns was clustered to identify the texture categories.) For a 256×256 image, for example, 4000 patterns are used. The performance of the classifiers is reported as percentage of misclassified pixels. The method of error estimation is essentially a "hold out" method — we use about 6% of the patterns to train the classifier, then use the entire patterns, including the training patterns, to test it.

3.4.1 Segmentation Using a Neural Network Classifier

Several papers have appeared in the literature that address texture segmentation by neural networks [69, 94]. One might question the biological plausibility of the clustering algorithm used to obtain the segmentations in Section 3.3. However, there are reasons to believe that biological systems are capable of carrying out such clustering or grouping operations [11]. Computational models of the brain are largely characterized by highly interconnected information processing units. This computational paradigm is known as massively parallel computers, connectionist architecture, or neural networks [40].

An important characteristic of neural networks is their learning capability. The supervised pattern classification capability of neural networks have been demonstrated by many researchers. In this section, we will use feed-forward networks along with the back-propagation training algorithm to carry out supervised texture segmentation experiments.

A feed-forward network may be regarded as a mapping from the input space to the output space. In this application, the input consists of texture features. Therefore, there are as many input units as features. Furthermore, there are as many output units as texture categories. In addition, one or more *hidden* layers usually are used between the *input* and *output* layers. Figure 3.17 shows a feed-forward neural network with one hidden layer.

Networks with no hidden layers, known as *perceptron*, have been studied and used extensively. However, the set of mappings from input to output that can be carried out by these networks is restricted [74]. Adding one or more hidden layers allows for an *internal representation(s)* to be formed, which in turn enables the network to carry out arbitrary mappings from input patterns to output patterns. In fact, a feed-forward neural network with only two hidden layers, linear output nodes, and sigmoidal nonlinearities, can perform complex nonlinear mappings [64].

Training a feed-forward neural network, using a set of training patterns, is equivalent to finding a set of weights for all the links (connections), such that the proper output unit is activated for the corresponding input pattern. Several algorithms for 'training' neural networks have been proposed and their utility for pattern classification has been demonstrated. For example, Rumelhart *et al.* [81] have proposed a network training algorithm based on error propagation known as the back-propagation or generalized delta rule.

Although there are some guidelines for the minimum number of hidden layers, similar guidelines for number of units in each layer are not available. In our experiments, we use a single hidden layer consisting of 10 units. For our simulations, we have used the back-propagation routines in the Rochester Connectionist Simulator (RCS) [39]. The training is stopped after a prespecified number of training cycles. More specifically, the performance under a small number (10) and a larger number (100) of training cycles is studied.

Figures 3.18–3.22 show several examples of supervised texture segmentation using the feed-forward neural network in Figure 3.17. The filters used in each case

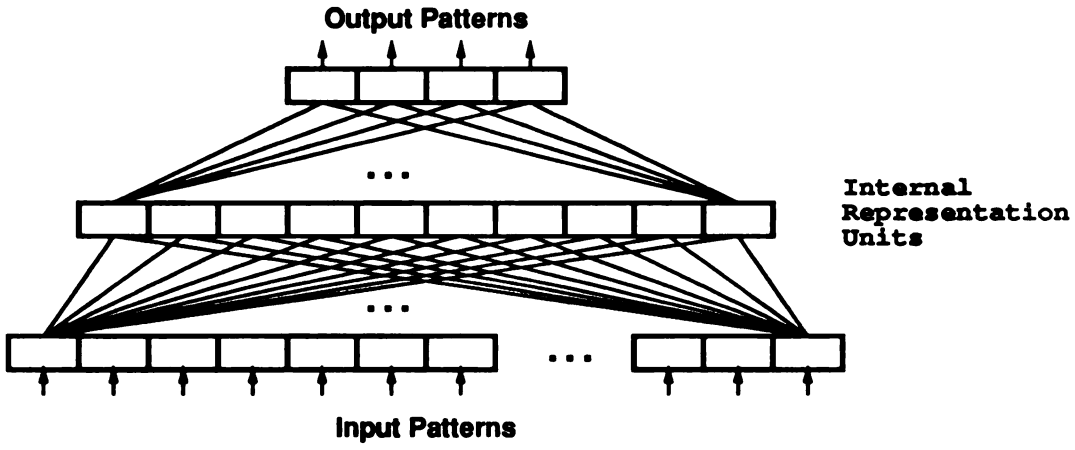


Figure 3.17: The feed-forward neural network used in our supervised texture segmentation experiments. The network has a single hidden layer with 10 units.

account for 95% of the intensity variation captured by an initial filter set with 20 filters. (See Section 3.1.2.) Table 3.2 lists the percentage of misclassified pixels when only 10 training cycles are used. Table 3.3 shows the results for 100 cycles. Even with a small number of training cycles, the feed-forward network's performance is impressive.

Table 3.2: Percentage of misclassified pixels using a feed-forward neural network classifier. Only 10 training cycles were used in each case.

Input Image			Misclassified (%)	
name	size	# categories	without (r,c)	with (r,c)
D03-D17	128 × 256	2	1.32	0.80
D03-D68	128 × 256	2	0.89	1.49
D17-D77	128 × 256	2	0.63	0.50
D24-D09	128 × 256	2	5.41	4.41
D55-D68	128 × 256	2	1.80	0.78
GMRF-4	256 × 256	4	2.92	1.75
Nat-5	256 × 256	5	5.64	4.54
Nat-16	512 × 512	16	12.77	8.43
L and +	256 × 256	2	3.39	3.61
Even-Odd	256 × 256	2	3.07	1.29
Tri-Arr	256 × 256	2	2.88	2.93

Table 3.3: Percentage of misclassified pixels using a feed-forward neural network classifier. 100 training cycles were used in each case.

Input Image			Misclassified (%)	
name	size	# categories	without (r,c)	with (r,c)
D03-D17	128 × 256	2	1.11	0.99
D03-D68	128 × 256	2	1.02	1.20
D17-D77	128 × 256	2	0.60	0.78
D24-D09	128 × 256	2	4.29	3.19
D55-D68	128 × 256	2	0.93	0.79
GMRF-4	256 × 256	4	2.77	1.65
Nat-5	256 × 256	5	3.73	2.75
Nat-16	512 × 512	16	7.32	5.67
L and +	256 × 256	2	3.32	3.02
Even-Odd	256 × 256	2	0.83	0.84
Tri-Arr	256 × 256	2	2.78	2.95

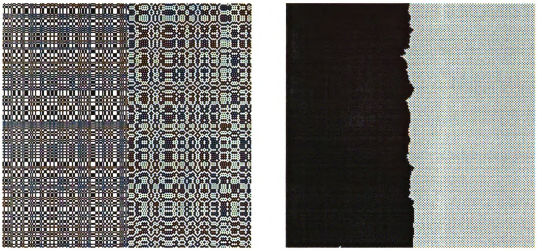

(a) (b)

Figure 3.18: (**a**) The 'Even-Odd' texture pair (256×256). (**b**) Supervised segmentation obtained using a feed-forward neural network. (Number of training cycles = 100.)

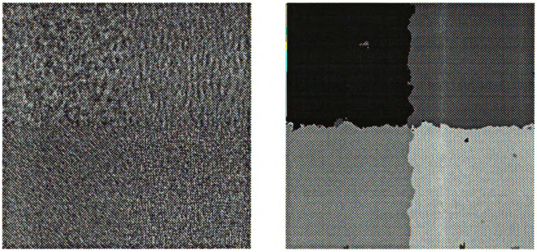

(a) (b)

Figure 3.19: (**a**) The 'GMRF-4' image (256×256) containing four Gaussian Markov random field textures. (**b**) Supervised segmentation obtained using a feed-forward neural network. (Number of training cycles = 100.)

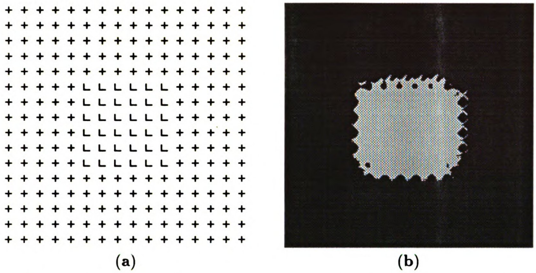


Figure 3.20: (a) The 'L and +' texture pair (256 × 256). (b) Supervised segmentation obtained using a feed-forward neural network. (Number of training cycles = 100.)

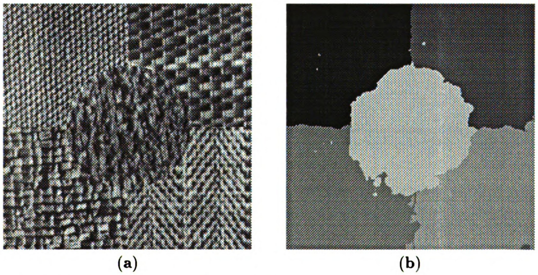
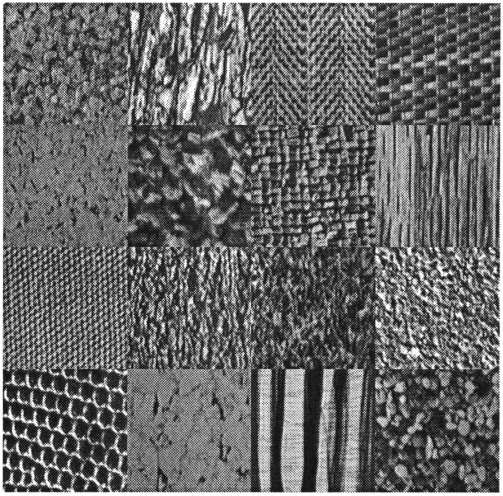


Figure 3.21: (a) The 'Nat-5' image (256×256) containing five natural textures (D77, D55, D84, D17, and D24) from the Brodatz album. (b) Supervised segmentation obtained using a feed-forward neural network. (Number of training cycles = 100.)



(a)

Figure 3.22: (**a**) The 'Nat-16' image (512×512) containing sixteen natural textures (row 1: D29, D12, D17, D55; row 2: D32, D5, D84, D68; row 3: D77, D24, D9, D4; row 4: D3, D33, D51, D54) from the Brodatz album. (**b**) Supervised segmentation obtained using a feed-forward neural network. (Number of training cycles = 100.)

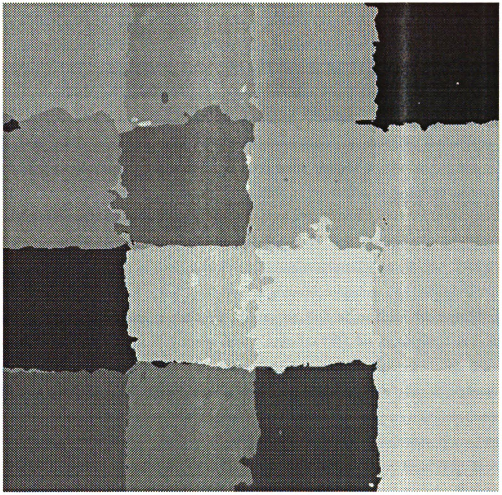

(b)

Figure 3.22: (*cont'd.*)

3.4.2 Comparison with Other Classifiers

How does the feed-forward neural network classifier in Section 3.4.1 compare with commonly used classifiers in the pattern recognition literature? To answer this question on a quantitative basis we carried out supervised texture segmentation experiments using a number of commonly used classifiers. We will briefly describe each classifier. The reader may refer to [30] for more details.

To classify a new pattern, the k-nearest neighbor classifier first determines the k nearest training patterns. It then assigns the pattern to the class that is most heavily represented in the k nearest neighbors. In addition to k-NN classifiers we will consider two other conventional classifiers also: the minimum Euclidean distance classifier [30] and the minimum Mahalanobis distance classifier. The latter is also known as Fisher's classifier [50]. We used the minimum Euclidean distance classifier in Section 3.3 in our two-step clustering. (There, we referred to it simply as the minimum distance classifier.) The following experiments will allow us to evaluate possible advantages of using a different classifier in step 2.

For the k-NN classifier, no training is required, but the storage requirement is large. Also, classification of test patterns is computationally expensive. Determining the k nearest neighbors of a test pattern, in general, requires computing its distance from all of the stored training patterns. However, fast algorithms for searching nearest neighbors are available. Fukunaga and Narendra [35], for example, have proposed a branch and bound algorithm for computing k-nearest neighbors. An alternative approach is to reduce the number of training samples by selecting a representative subset. The *condensing technique* proposed by Hart [44] is only one of the many reduction techniques. When combined with such preprocessing techniques, the complexity of the k-NN classifier compares quite favorably with other classifiers [30]. In our implementation of the k-NN classifier we did not use any preprocessing. However, for computational efficiency, the Manhattan (city block) distance measure, rather than the usual Euclidean distance, is used. Three different values of k: 1, 3, and 5 are tried. Since the performance of all three classifiers was essentially the same, only the results for the 3-NN classifier are reported here.

Tables 3.4, 3.5, and 3.6 show the percentage of misclassified pixels for different segmentation experiments. For easier comparison, Figures 3.23 and 3.24 show the same data as scatter plots. The image numbers on the horizontal axis correspond

Table 3.4: Percentage of misclassified pixels using the minimum Euclidean distance classifier.

Input Image			Misclassified (%)	
name	size	# categories	without (r,c)	with (r,c)
D03-D17	128 × 256	2	0.94	0.90
D03-D68	128 × 256	2	0.98	0.96
D17-D77	128 × 256	2	0.92	0.82
D24-D09	128 × 256	2	9.48	7.15
D55-D68	128 × 256	2	0.63	0.60
GMRF-4	256 × 256	4	2.59	1.75
Nat-5	256 × 256	5	4.44	2.85
Nat-16	512 × 512	16	9.20	6.19
L and +	256 × 256	2	1.97	1.97
Even-Odd	256 × 256	2	2.20	1.95
Tri-Arr	256 × 256	2	5.47	4.68

to the row numbers in the tables. So, for example, image number 8 refers to the 'Nat-16' image which contains sixteen natural textures. Clearly, the 3-NN classifier outperforms the minimum distance classifiers. It also outperforms the feed-forward neural network used in Section 3.4.1. However, it took more than four days of CPU time on a Sun-4/390 to obtain the segmentation for the 'Nat-16' image. As pointed out before, preprocessing techniques can be used to reduce computational complexity of the k-NN classifier. Note that, in most cases, the performance of the feed-forward neural network classifier has improved when the number of training cycles is increased to 100. It is very likely that using a larger number of training cycles, e.g. 500 cycles, will further improve its performance.

In the two-step clustering scheme in Section 3.3 we used the minimum distance classifier in the second step. The supervised texture segmentation experiments in this section suggest using other classifiers. For example, the minimum Mahalanobis distance classifier, or the k-NN classifier in conjunction with the condensing technique, may result in better segmentations.

Table 3.5: Percentage of misclassified pixels using the minimum Mahalanobis distance classifier.

Input Image			Misclassified (%)	
name	size	# categories	without (r,c)	with (r,c)
D03-D17	128 × 256	2	0.70	0.63
D03-D68	128 × 256	2	0.82	0.81
D17-D77	128 × 256	2	0.62	0.55
D24-D09	128 × 256	2	2.89	2.91
D55-D68	128 × 256	2	0.72	0.45
GMRF-4	256 × 256	4	1.97	1.51
Nat-5	256 × 256	5	2.82	2.39
Nat-16	512 × 512	16	5.41	3.37
L and +	256 × 256	2	1.26	1.26
Even-Odd	256 × 256	2	0.88	0.75
Tri-Arr	256 × 256	2	2.35	2.49

Table 3.6: Percentage of misclassified pixels using the 3-NN classifier. Classification errors for 1-NN and 5-NN were essentially the same.

Input Image			Misclassified (%)	
name	size	# categories	without (r,c)	with (r,c)
D03-D17	128 × 256	2	0.37	0.35
D03-D68	128 × 256	2	0.57	0.53
D17-D77	128 × 256	2	0.42	0.38
D24-D09	128 × 256	2	0.65	0.63
D55-D68	128 × 256	2	0.39	0.45
GMRF-4	256 × 256	4	1.48	1.07
Nat-5	256 × 256	5	1.17	1.12
Nat-16	512 × 512	16	1.23	1.23
L and +	256 × 256	2	0.82	0.81
Even-Odd	256 × 256	2	0.45	0.48
Tri-Arr	256 × 256	2	1.87	1.50

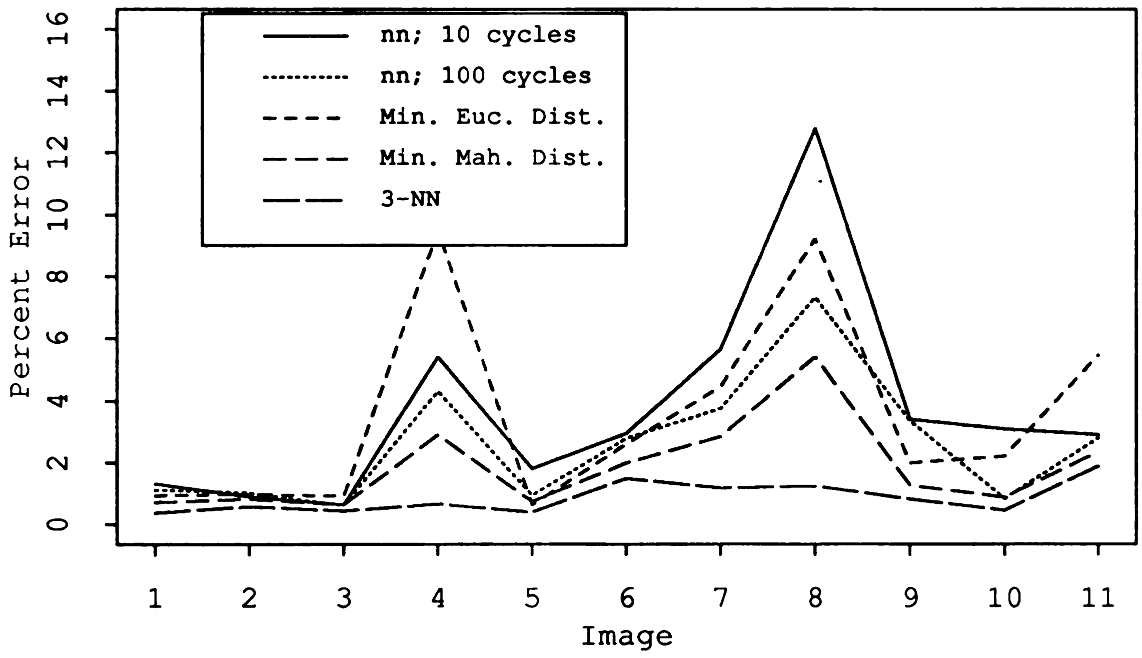


Figure 3.23: Percent misclassified pixels for various classifiers. Here, the (row, col) coordinates of pixels are *not* used.

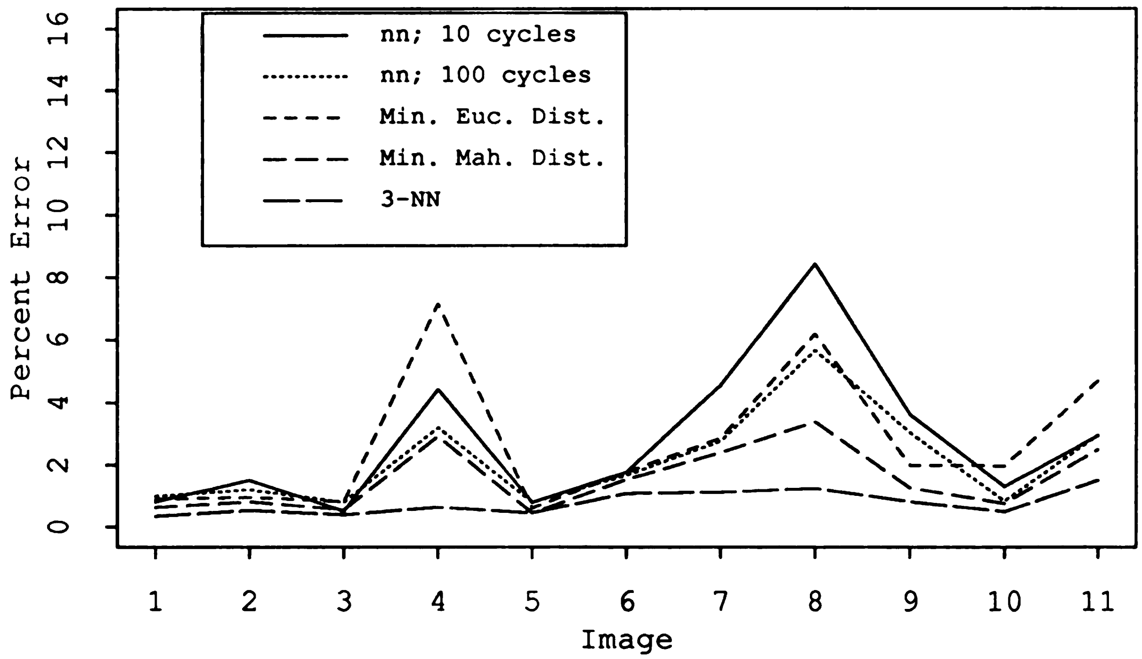


Figure 3.24: Percent misclassified pixels for various classifiers. Here, the (row, col) coordinates of pixels are used as additional features.

3.5 Summary

In this chapter, we presented a multi-channel filtering technique for texture segmentation. The channels were represented with a fixed set of Gabor filters and a systematic filter selection scheme was proposed, which is based on reconstruction of the original image from the filtered images. As a result, unlike some of the existing techniques, our segmentation algorithm does not require any knowledge of the frequency content of textures in the input image. Both unsupervised and supervised texture segmentation experiments were conducted and the ability of the "texture energy" features to discriminate among various textures was demonstrated. In particular, we demonstrated the ability of the segmentation technique to discriminate artificially generated texture pairs with identical second- and third-order statistics.

The filtering and feature extraction operations in the algorithm account for most of the required computations. However, these operations can be performed in parallel, regardless of the number of filters. The use of a nonlinear transformation following the linear filtering operations has been suggested as one way to account for the inherently nonlinear nature of biological visual systems [29]. We argued that the localized filtering by the Gabor filter set followed by a "squashing" nonlinear transformation can be interpreted as a multi-scale blob detection operation.

One of the limitations of the texture segmentation algorithm is the lack of a criterion for choosing the value of α in the nonlinear transformation. In the experiments, we used a fixed empirical value. Also, the algorithm assumes that different channels are independent from each other. However, there is psychophysical and physiological evidence indicating inhibitory interactions between different spatial frequency channels [29]. Some researchers have incorporated such interactions in their texture segmentation algorithms [8, 66]. Feature selection or extraction from the initial pool of features is computationally desirable, and may result in more accurate segmentations. Allowing inhibitory interactions among the channels is shown to have the potential to reduce the effective dimensionality of the feature space [8].

For unsupervised texture segmentation, we tried to use the plot of the modified Hubert index versus number of texture categories. In most cases, however, it was assumed that the true number of categories is known *a priori*. In the following chapter we will propose an integrated approach that combines the current region-based segmentation technique with an edge-based technique. The integrated approach requires only a reliable upper bound for the number of texture categories.

In the *supervised* texture segmentation experiments a feed-forward neural network was used. Some of the architectures for neural networks are capable of *unsupervised* classification. One example, is the "self-organizing" and "self-stabilizing" architecture proposed by Carpenter and Grossberg [11]. An attempt was made to use the ART2 architecture [12] as a substitute for the square-error clustering algorithm that was described in Section 3.3. However, the experiments were discontinued because the initial experimental results were not promising.

Chapter 4

Integrating Region- and Edge-Based Texture Segmentations

The texture segmentation technique proposed in Chapter 3 results in a ***region-based*** segmentation, as it assigns pixels with similar texture properties to the same region. One of the disadvantages of our region-based segmentation is the need for knowing the "true" number of texture categories ahead of time. In fact, this drawback applies to *all* region-based techniques. Another method of discriminating regions with different textures is to detect the boundaries between them. The output of a ***boundary detection*** operation is commonly referred to as an ***edge-based segmentation***. The knowledge of the true number of texture categories is not necessary for obtaining an edge-based segmentation, where the presence of an edge point is determined locally based on a measure of disparity across the (unknown) boundaries.

In this chapter, we describe a new technique that produces an edge-based segmentation by combining the magnitude responses of feature images to a common edge detector. As we will see, an important limitation of an edge-based segmentation is that, in practice, the region boundaries are not closed. Some postprocessing is, therefore, required to obtain closed regions. In contrast, the region boundaries in a region-based segmentation are always closed. An integrated approach that combines the advantages of the two methods could result in a better segmentation. We propose one such integrated approach and demonstrate its effectiveness.

In the multi-channel filtering approach, there are two fundamentally different ways in which the integration can take place.

1. Integration takes place separately in different channels. The resulting segmentations are then combined to obtain the final segmentation.

2. A single edge-based segmentation is first obtained, then integrated with a region-based segmentation.

The multidimensional nature of texture representation tells us that the information from all channels should be used simultaneously to obtain a segmentation. That is, the information from a single channel is often insufficient to discriminate different textures or to properly measure the disparity across the boundaries. As a result, the second approach appears to be more plausible.

4.1 Edge-Based Segmentation

Detecting texture boundaries requires simultaneous consideration of spatial variations in all feature images. The idea is to combine the "evidence" for texture edges in different feature images to obtain a single measure of edge strength at each point. One example of this approach is the method proposed by Khotanzad and Chen [60]. Our edge-based segmentation technique is similar to that of Malik and Perona [66].

The multi-dimensional edge detection is performed as follows. First, we apply the Canny step edge detector [10] to each feature image. The implementation of the Canny edge detector used in our experiments uses the efficient approximation of the optimal detector by the first derivative of a Gaussian. Only one detector with appropriate operator width is used for each feature image. (The Canny edge detector, in its general form, uses several operator widths whose outputs are then combined using the 'feature synthesis' method.) The width of the operator σ is adapted to each feature image. In our experiments we have used $\sigma = \sqrt{2}T$, where $T = N_c/u_0$ is the average size of the intensity variations detected by the corresponding Gabor filter. The reason for using different operator widths for different feature images is that the step edges in different feature images have different widths — those in lower frequency channels tend to be wider.

Each feature image is normalized to have a mean of zero and a constant standard deviation (= 30). Again, this normalization is intended to avoid domination of features with smaller numerical ranges by those with larger numerical ranges. Only the magnitude response of the Canny edge detector is computed for each feature image, the nonmaximum suppression and the hysteresis thresholding [10] are not performed. Figure 4.1 shows examples of Canny magnitude images for the 'D55-D68' texture pair. These magnitude images correspond to feature images shown in Figure 3.6.

To obtain a single 'total' magnitude image, the Canny magnitude responses corresponding to individual feature images are summed, point-by-point. Similarly, the gradient images, one along the x-axis and one along the y-axis, are summed to obtain 'total' gradient images. At this point, the nonmaximum suppression and hysteresis operations are applied to the total magnitude image to obtain an edge-based segmentation.

4.1.1 Experimental Results

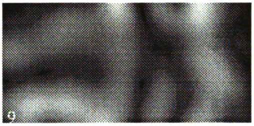
Figure 4.2 illustrates the edge-based segmentation for the 'D55-D68' texture pair. Figure 4.2(b) shows the total magnitude response of the Canny edge detector, obtained by adding the magnitude responses of 13 feature images. Lighter gray values in this image indicate higher edge strengths. Figure 4.2(c) shows the edge image obtained by nonmaximum suppression and hysteresis thresholding. The low and high threshold values were 0.5 and 0.8, respectively.

Other examples of edge-based segmentation are given in part (c) of Figures 4.5 through 4.8. In Section 4.2 we describe how these edge-based segmentations can be used in an integrated segmentation technique. Here we will give a second example to demonstrate the shortcomings of our current edge-based segmentation technique. This example is shown in Figure 4.3 and involves a 256×256 image with five natural textures.

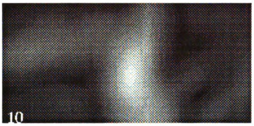
As seen in (Figure 4.3(b)), the total magnitude response for some true texture boundaries is much stronger than others. The primary reason is that some texture boundaries have a strong response in several feature images, while some others enjoy a strong response in only a few feature images. Adding the magnitude responses in different channels has some desirable consequences. In particular, it helps suppress noise and allows evidence for a boundary in different channels to accumulate. Unfortunately, as seen in this example, some true texture boundaries are enhanced more than others. Using adaptive hysteresis thresholding should alleviate some of these problems. However, it is our belief that a different method of combining the magnitude responses is needed.

4.2 Integrated Approach

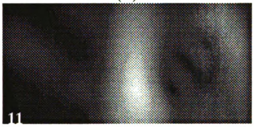
The general principle of integration or fusion of information from different sources is well recognized in the computer vision community. Several techniques for simultaneously utilizing the region and edge information in image segmentation have been



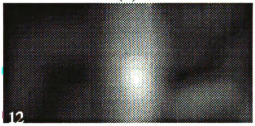
(b)



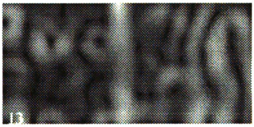
(c)



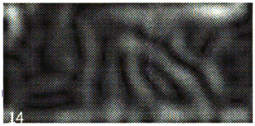
(d)



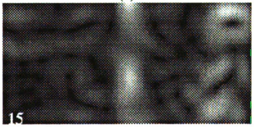
(e)



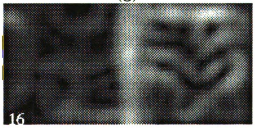
(f)



(g)



(h)



(i)

Figure 4.1: Canny magnitude images corresponding to feature images shown in Figure 3.6. A $\sigma = \sqrt{2}T$ was used for the Canny edge detectors.

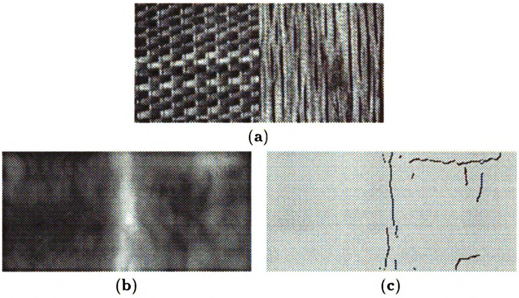


Figure 4.2: An example illustrating the edge-based segmentation technique. (**a**) Input image ('D55-D68'). (**b**) Total Canny magnitude response to 13 feature images. (**c**) Edge-based segmentation.

proposed [16, 41, 72, 75]. A similar integrated approach to texture segmentation, however, has not been emphasized. The multidimensional nature of texture segmentation makes either of the region- and edge-based segmentations complicated, making an integrated approach even more formidable. As we will see, an integrated approach need not necessarily be overwhelming. A proper integration method should suppress the weaknesses and emphasize the strengths of the region- and edge-based segmentation techniques.

The integration of the region-based and the edge-based segmentations is carried out as follows. Jain and Nadabar [51] have applied a similar integration technique to segmentation of range images. First, using the estimated upper bound k_{max} for the number of texture categories, a segmentation is obtained using the algorithm described in Section 3.3. Suppose the true number of categories (clusters) is k^*. The segmentation (clustering) with k_{max} ($> k^*$) categories is always formed by breaking some of the true clusters. Therefore, assuming that our texture features provide strong discrimination between different texture categories, the false splitting of regions will occur within true segments only, not across them. The algorithm described in Section 4.1 is used to obtain an edge-based segmentation (an edge image). For the

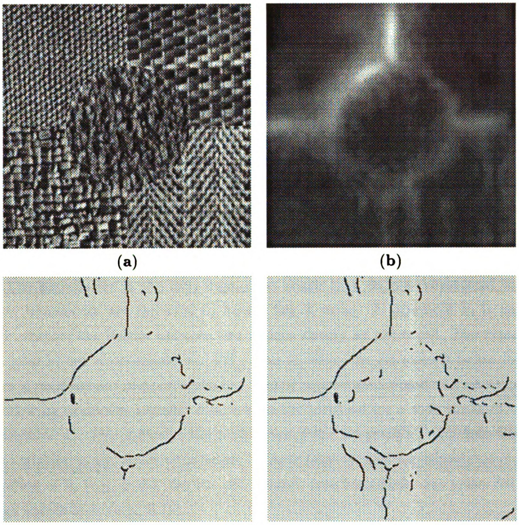

(a) (b) (c) (d)

Figure 4.3: An example demonstrating some of the shortcomings of the current edge-based segmentation technique. (**a**) Original input image. (**b**) Total Canny magnitude response to 13 feature images. (**c**) Edge-based segmentation. The low and high hysteresis thresholds were 0.5 and 0.8, respectively. (**d**) Edge-based segmentation. The low and high hysteresis thresholds were 0.5 and 0.7, respectively.

Canny edge detector, we use 0.5 and 0.8 for the low and high thresholds of the hysteresis, respectively. These relatively low threshold values may generate a large number of spurious edges. However, a perfect edge image is not crucial for the purpose of integration. What is important is that we minimize the likelihood of missing true edges.

Now, in the over-segmented image, for each border between pairs of regions, we compute the fraction of border sites that "coincide" with an edge point in the edge image. We refer to this fraction as hit-ratio h. The coincidence is determined by examining a small rectangular neighborhood centered at the border site. The typical neighborhood size used in our experiments was 5×10. The longer dimension of this rectangular neighborhood is along the local orientation of the border site. We have only considered the $0°$ and $90°$ orientations. The border site between two adjacent pixels, one with label l_1 and the other with label l_2, is said to have a $0°$ orientation if the pixels are to the east and west of each other. The $90°$ orientation is formed by pixels to the north or south of each other.

The boundaries in the over-segmented image that do not correspond to true texture boundaries are expected to have a low h value. A threshold h_t is used to decide whether the border between two regions should be preserved. The typical h_t value used in our experiments was 0.5. That is, when the hit-ratio is below 0.5 the border is removed and the corresponding pair of regions are merged. As we will see in the following examples, segmentation results are identical for a wide range of values for threshold h_t. This indicates the robustness of the integration technique. That is, the integration is not sensitive to noise, because false borders are eliminated even at low values of h_t (e.g., 0.25). On the other hand, true boundaries are preserved even at high values of h_t (e.g., 0.75).

4.2.1 Experimental Results

As our first example we will apply our integration method to the 'D55-D68' texture pair in Figure 4.4(a). A region-based segmentation assuming a maximum of four texture categories is shown in Figure 4.4(b). The edge-based segmentation is shown in Figure 4.4(c). The segmentation after integration is shown in Figure 4.4(d). The same segmentation is obtained for all $h_t \in (0.12, 0.69)$. The integration result for another image containing two GMRF textures ('GMRF-2') is illustrated in Figure 4.5.

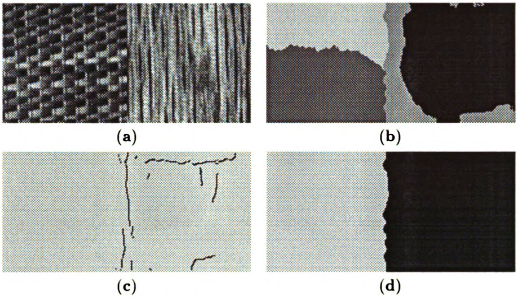


Figure 4.4: Region- and edge-based integration results for the 'D55-D68' texture pair (128×256). (**a**) Original input image. (**b**) Four-category region-based segmentation (over-segmented). (**c**) Edge-based segmentation. (**d**) New segmentation after integration.

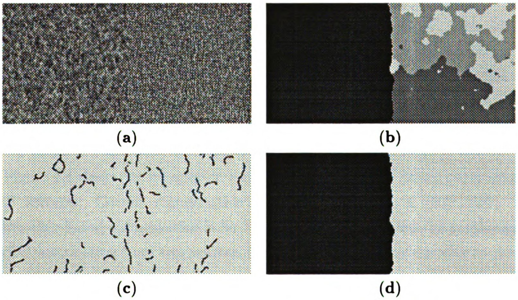


Figure 4.5: Region- and edge-based integration results for the 'GMRF-2' image (128×256). (**a**) Original input image. (**b**) Four-category region-based segmentation (over-segmented). (**c**) Edge-based segmentation. (**d**) New segmentation after integration.

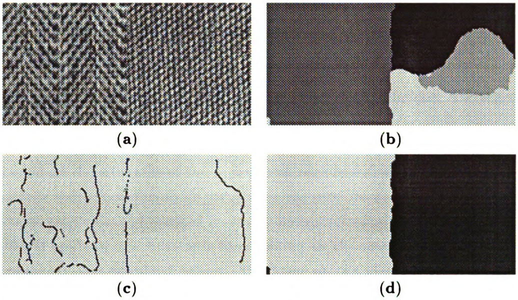


Figure 4.6: Region- and edge-based integration results for 'D17-D77' texture pair (128×256). (**a**) Original input image. (**b**) Four-category region-based segmentation (over-segmented). (**c**) Edge-based segmentation. (**d**) New segmentation after integration.

In this case, the correct segmentation is obtained for all $h_t \in (0.33, 0.89)$.

Figure 4.6 shows similar results for the 'D17-D77' texture pair. Here, identical segmentation is obtained for all $h_t \in (0.23, 0.89)$, indicating that the integration method is highly robust. Figure 4.7 illustrates the integration method on a 256×256 image with one natural texture embedded in another ('D84 in D68'). Again, correct segmentation is obtained for all $h_t \in (0.32, 0.95)$. Identical segmentations for a wide range of values for threshold h_t indicates that the integration is highly robust.

Figure 4.8 shows the integration results for the 'GMRF-4' image containing four GMRF textures. Correct segmentation is obtained for all $h_t \in (0.31, 0.69)$. Note that the edge-based segmentation is far from perfect. Nonetheless, by combining the imperfect information from two sources, the integration technique is able to produce the correct segmentation.

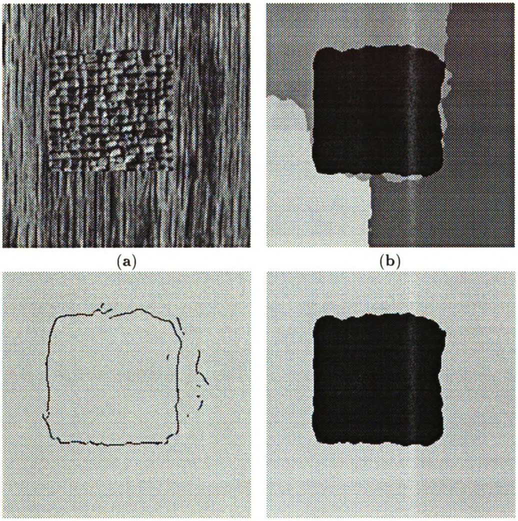
Finally, in Figure 4.9, we show the integration result for the 'Nat-5' texture. In order to extract all true edges, hysteresis thresholds of 0.5 and 0.7 are used, rather than 0.5 and 0.8 which were used in all the previous examples. (See discussion in Section 4.1.1.) Also, instead of a 5×10 window, a larger 9×18 window is used. With these parameters, the correct segmentation is obtained for all $h_t \in (0.32, 0.65)$.

4.3 Summary

In this chapter, we presented an edge-based segmentation technique. The segmentation technique detects texture boundaries by combining the magnitude responses of feature images to the Canny edge detector. We then proposed an integrated approach that combines the strengths of the region- and edge-based segmentations. The integrated approach allowed us to do away with the need for knowing the true number of texture categories, and resulted in a truly unsupervised segmentation technique.

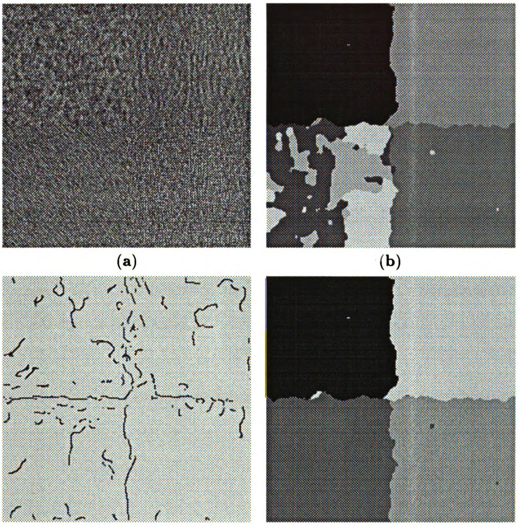
When applying the hysteresis thresholding to the total magnitude image, we used relatively low threshold values. By doing so, we allowed for more spurious edges, but minimized the likelihood of missing true texture edges. As a result, the edge-based segmentations in most of the examples were far from perfect. Nonetheless, combining imperfect information from two sources, the integration technique was able to produce the correct segmentations. The robustness of the integration method is reflected in identical segmentation results for a wide range of the threshold h_t on hit-ratio.

The edge-based segmentation technique, in its current form, has certain limitations. We showed that, in some cases, adding magnitude responses in different channels may enhance some true texture boundaries more than others. Consequently, when magnitude responses from different channels are added together, in addition to noise, some true texture boundaries are also suppressed. It is our belief that a different method of combining the magnitude responses is needed.



(a) (b) (c) (d)

Figure 4.7: Region- and edge-based integration results for the ‘D84-in-D68’ (256×256). (**a**) Original input image. (**b**) Four-category region-based segmentation (over-segmented). (**c**) Edge-based segmentation. (**d**) New segmentation after integration.



(a) (b) (c) (d)

Figure 4.8: Region- and edge-based integration results for the 'GMRF-4' image (256 × 256). (**a**) Original input image. (**b**) Six-category region-based segmentation (over-segmented). (**c**) Edge-based segmentation. (**d**) New segmentation after integration.

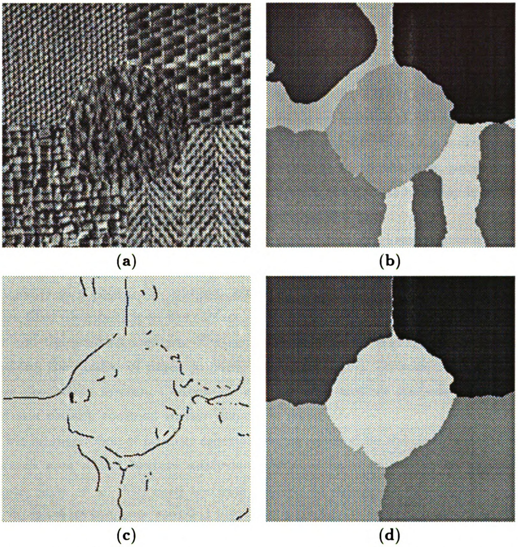


Figure 4.9: Region- and edge-based integration results for a 256×256 image containing five natural textures ('Nat-5'). (**a**) Original input image. (**b**) Seven-category region-based segmentation (over-segmented). (**c**) Edge-based segmentation. (**d**) New segmentation after integration.

Chapter 5

Texture Analysis of Automotive Finishes

In recent years, there has been a growing emphasis on machine vision and its applications in manufacturing processes. To achieve higher speed and increased reliability, machine vision systems are being used with increasing frequency to perform various inspection tasks. For example, visual inspection of mass-produced printed circuit boards, integrated circuit chips, and photomasks in electronics industry is an important area where machine vision techniques are used [15, 42].

Since in many cases the quality of a surface is best characterized by its texture, texture analysis plays an important role in automated visual inspection of surfaces. The texture of a paper, for example, controls its printability. This is because, the random fiber distribution on the surface of the paper affects the contact area between paper and the printing medium. Texture analysis techniques are, therefore, useful in controlling the quality of paper in paper-rolling mills [17]. As part of an automated lumber processing system, Conners *et al.* [23] used texture analysis techniques to detect and classify common surface defects in wood.

Visual inspection of product appearance, as assessed by the customer, is another important area where texture analysis techniques have proved to be useful. For example, Siew *et al.* [84] used textural features to determine the degree of carpet wear. In the food industry, textural appearance is an important factor in determining product quality [37].

In this chapter, we describe a problem involving automated visual inspection of automotive metallic finishes. The appearance of metallic finishes, which are primarily used in the automotive industry, is affected by their color as well as their *visual* texture. One of the factors that determines the acceptability of the finish is the degree of "uniformity" of its visual texture. Our goal is to find quantitative measures that capture the characteristics of the metallic finish texture, hereafter called simply finish texture. We use a multi-channel filtering technique to compute texture features that are used to grade the uniformity of metallic finish samples.

The organization of this chapter is as follows. In Section 5.1, we describe metallic finish and various factors that affect its appearance. We also describe the psychometric experiments which were designed to grade the degree of uniformity of finish textures. In section 5.2, we address image acquisition and preprocessing requirements. Section 5.3 describes the multi-channel filtering technique that is used to characterize the finish texture. The functional form of the 'channels' (filter functions) and the choice of filter parameters, as well as the definition of texture features are discussed. In Section 5.4, we propose two alternative ways to grade the degree of uniformity of finish texture. Finally, we conclude with a summary and a general discussion of the results in Section 5.5.

5.1 Metallic Finish

The sparkle and color directionality appearances of metallic automotive finishes are due to metal particles such as aluminum flakes that are added to the paint. The non-uniform distribution of position and tilt angle of these flakes within the paint film give rise to a *visual* texture which consists of patterns of light and dark color regions. The distribution of the flakes, and hence the perceived finish texture, is influenced by various parameters of the paint itself, and by various paint application parameters such as pot pressure, air pressure, gun distance, and rheology treatment. Ideally, we would like the finish texture to 'look' uniform. Judging the degree of uniformity of a finish texture, however, is a rather subjective process. Even finish inspection experts, among themselves, tend to have different opinions of uniformity.

Over the years, finish inspection experts have adopted various terms to describe the appearance of metallic finishes. Two frequently used terms are 'mottle' and 'blotchy' which appear to make up two potential components of uniformity. Mottleness refers to a pseudo-random positioning of metallic flakes that creates an accidental patterning effect. The size of these patterns is usually on the order of a millimeter. Blotchiness, on the other hand, refers to the non-uniformity characterized by irregularly spaced areas of color change. The size of these irregularities is usually on the order of an inch.

Metallic finish samples used in our experiments consist of metal panels that are painted under various settings of paint application parameters. Different settings of these parameters give rise to finish textures with different degrees of texture "uniformity". Specifically, two sets of finish samples are analyzed in the following experiments: the light blue set (LBLUE) and the medium blue set (MBLUE). There are 13 4" × 6" (about 10 cm × 15 cm) panels in each set. Panels in each set vary in paint application parameters (flash time, gun distance, and air pressure) as well as in the grade (size) of the aluminum flakes.

A group of paint technicians were asked to judge the uniformity of finish samples in each set. First, 10 observers were asked to rank the panels from most to least uniform. The rank order average was then used as *initial* ranking in a paired comparison experiment. Each panel was compared to eight other panels nearest to its rank order. For example, the panel with rank order 7 was compared to panels with rank orders 3, 4, 5, 6, 8, 9, 10, and 11. The pairs were presented, in random order, to four observers. Each observer was asked to select the more uniform panel from the pair shown. Each observer performed the comparisons 10 times. Using the results of the paired comparisons, a preference frequency matrix was constructed for each set. (The (i, j) entry in a preference frequency matrix shows the number of times the panel in row i was preferred over the panel in column j.) Ordinal scale values for the panels were then obtained using a scaling technique [88, Ch. 4]. The resulting 'visual scale values' are given in Tables 5.1 and 5.2.

In another visual scaling experiment, a panel of 10 paint technicians were asked to grade the finish samples in each set along other visual components that might be related to the perceived uniformity of the finish. These components are 'mottle', 'flake-size', and 'blotchy'. Each technician was asked to place the panels on a scale of 1 to 10, for each of the above components, with 10 indicating severe mottle effect, extremely coarse flake-size, or severe blotchy effect. Since the observers had no reference samples to define their base lines, significant individual biases in the resulting values are possible. The rank order of the scale values, on the other hand, are less likely to suffer from these individual biases. The mean rank values for each of the visual components are given in Tables 5.1 and 5.2. Our goal is to develop quantitative measures of finish texture that 'explain' these subjective data.

Table 5.1: Visual scale values for texture uniformity, mottle, flake-size, and blotchy appearance of panels in the LBLUE set.

Panel	Uniformity	Mottle	Flake-Size	Blotchy
1	3.02493	5.71604	4.80	5.8452
2	0.21339	8.61112	4.65	9.4642
3	3.62269	3.79093	11.10	3.0198
4	2.14104	5.90562	6.10	7.3358
5	2.42717	5.63374	7.75	7.2702
6	1.32080	8.00890	8.10	8.7932
7	1.04008	8.38379	5.25	9.7165
8	3.36581	3.17102	2.25	3.9988
9	0.24747	9.20337	6.60	9.6850
10	4.80362	3.34307	9.85	3.3227
11	0.00000	9.89626	11.30	11.8894
12	1.39585	6.65147	3.45	7.3092
13	4.28204	2.51435	9.80	3.3501

Table 5.2: Visual scale values for texture uniformity, mottle, flake-size, and blotchy appearance of panels in the MBLUE set.

Panel	Uniformity	Mottle	Flake-Size	Blotchy
1	1.10309	6.35	5.40	4.70
2	0.60587	10.10	4.25	8.85
3	2.32165	5.70	12.40	6.05
4	2.03682	3.85	5.00	3.80
5	0.00000	11.20	8.05	11.25
6	1.35577	9.30	8.05	7.95
7	1.11753	5.40	4.55	8.05
8	1.77176	3.75	4.10	5.90
9	0.12917	9.05	5.35	9.10
10	3.42172	5.20	11.75	4.40
11	2.28994	5.35	11.85	4.80
12	0.43111	7.65	3.95	6.95
13	0.85181	8.10	6.30	9.20

5.2 Image Acquisition and Preprocessing

Although there are some general guidelines for lighting and imaging setups, every machine vision application has its own unique peculiarities that have to be dealt with individually. A crucial issue in any imaging problem is the selection of an appropriate light source(s) that highlights features of interest. The lighting geometry, i.e. relative positions of light source, camera, and the object, is equally important.

We have experimented with various types of light sources and concluded that directed lighting - as opposed to diffused lighting - is more appropriate for highlighting the texture of the finish. The specular nature of the metallic finish poses a great challenge in achieving uniform illumination. In general, imaging metallic and specular objects are much harder than imaging lambertian objects.

A common technique for dealing with the problem of specular reflections is to use a pair of polarizing filters. The first filter (the polarizer) is used to polarize the light source. The object is then viewed through the second, cross polarized, filter (the analyzer). Since specular regions do not alter the polarization of the incident light, the reflected light from these regions is blocked by the analyzer. The reflected light from truly diffuse regions, on the other hand, is depolarized and partially passed by the analyzer. This technique, however, proved to be unsuitable to our application. The reason was that almost all the reflected light from the surface of the panel was blocked by the analyzer. This resulted in a significant loss of detail and contrast in the acquired images. We encountered a similar problem when we tried to use dulling spray to cut down the specularity of the finish[1].

In order to alleviate the problems arising from specular nature of the metallic finish, we were forced to reduce the size of the area on the panel surface that is being imaged. The relatively high image resolution required for analyzing the finish texture also dictated using high magnifications, and hence imaging a small area on the panel surface. Our current imaging setup is shown in Figure 5.1. We use a single light projector to illuminate the finish sample (panel). To minimize the illumination variations, we keep the angle between the axis of the camera and the axis of the light projector as small as possible. Note that too small an angle will result in a very large specular reflection into the camera. A value between $15°$ to $20°$ was found to be a

[1]Dulling sprays have been used by professional photographers for eliminating the glare on shiny surfaces.

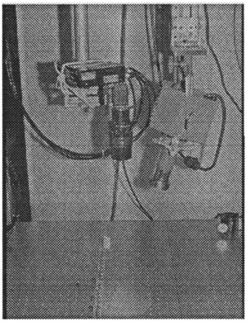


Figure 5.1: The imaging setup used for acquiring images from finish samples. The main components of the setup are a *Mole-Richardson* light projector and a *Panasonic* CCD camera with a 90 mm macro lens.

good compromise.

The maximum resolution of the human eye is estimated to be about 60 cycles per degree of visual angle [29]. Assuming a standoff of 0.5 meter, this value translates into 0.073 mm per individual receptor. The resolution of images obtained using our imaging setup is 0.08 mm/pixel which is close to the above value. We acquired several images from each panel by shifting the panel with respect to a fixed position of the camera and the light projector. The image data base used in the following experiments contains eight 256 × 256 images from each panel in each of the two sets. Each image, therefore, corresponds approximately to a 2.06 cm wide by 1.54 cm high (about 0.81" × 0.61") physical area[2]. Figure 5.2 illustrates the physical location of the images taken from a given panel.

[2]Note that the physical area is not a square, because the CCD camera used for acquiring the images has a 4:3 aspect ratio. The resolution indicated here is along the horizontal direction. The resolution in the vertical direction is slightly higher (by a factor of 4/3).

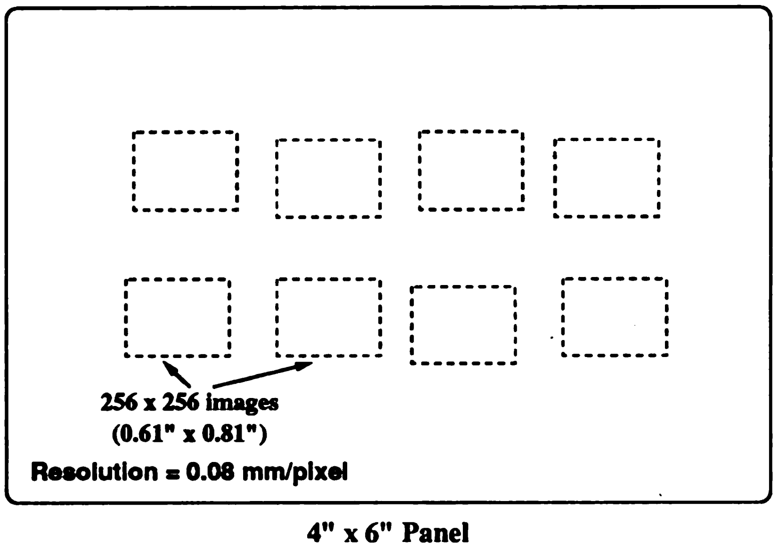
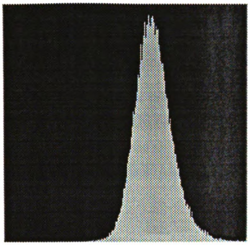


Figure 5.2: Multiple imaging from a given panel. The resolution of acquired images is close to the maximum resolution of the human eye.

5.2.1 Preprocessing

We use a number of preprocessing operations to compensate for non-uniform illumination of the panels. These operations include an 'image subtraction' stage where the (smoothed) image of the "background" is subtracted from the original image. If the intensity variations inherent to the light source itself, or due to the position of the light source, were known, one could compensate for the resulting non-uniformities in the illumination by subtracting these variations from the acquired images. In practice, these variations can be approximated by an intensity image of the "background". We obtain the background image by imaging an unpainted metal panel. The background image is smoothed to suppress the fine texture of the unpainted metal panel.

Figure 5.3 shows the gray level histograms of two finish samples, one with fine and the other with coarse aluminum flakes. Both histograms are highly symmetric and have a similar shape (they both look like a Gaussian distribution). However, the histogram of the finish sample with coarse aluminum flakes is wider than the other. Such differences in the gray level distribution is caused by variations in lighting, as well as by differences in paint factors such as color and the grade (size) of the aluminum flakes. Further preprocessing is needed in order to compensate for such variations.


(a)

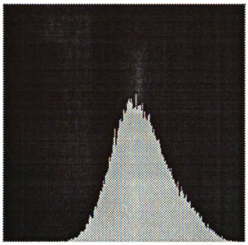

(b)

Figure 5.3: Two examples demonstrating differences in the histograms of the acquired images. (**a**) Histogram of a finish sample with fine aluminum flakes. (**b**) Histogram of a finish sample with coarse aluminum flakes.

Histogram equalization operations (also known as histogram flattening or probability equalization) are often used to remove differences in the first-order statistics of images. Histogram equalization algorithms achieve this goal by reassigning pixel gray levels so that the population of pixels with a given gray level (or a small range of gray levels) is the same for each gray level (or range of gray levels). However, the similarity in the shapes of the histograms for different finish samples suggests that a linear scaling of the gray levels should be sufficient for suppressing differences in the first-order statistics of the acquired images. The effective width, or spread, of a histogram can be measured in several different ways. We use the average absolute deviation (AAD) from the mean value for this purpose. Let $s(x, y)$ be the acquired image. The AAD measure is then given by

$$f_0 = \frac{1}{N_r N_c} \sum_{a=1}^{N_r} \sum_{b=1}^{N_c} \mid s(a,b) - \bar{g} \mid , \tag{5.1}$$

where N_r and N_c are the number of rows and columns, and $\bar{g}$ is the mean gray level in the image. Image normalization is achieved by dividing the gray levels in each acquired image by its AAD measure.

In Section 5.3, we will use the AAD measure in (5.1) to compute texture features in the filtered images. There we will show that the above image normalization can be applied equivalently to texture features obtained by processing the original image.

5.3 Characterization of Finish Texture

We characterize the textural appearance of the metallic finish by using a multi-channel filtering technique. In this section, we describe the functional form of the channels, the choice of the filter parameters, and the definition of texture features. We will use the resulting texture features as input to the texture grading methods described in Section 5.4.

5.3.1 Filter Functions and Parameters

Metallic finish textures do not possess significant orientation tendencies, i.e. they are practically isotropic. Even-symmetric Gabor filters used in our texture segmentation algorithms have both frequency- and orientation-selective properties. Instead of using Gabor filters, therefore, in this application, we characterize the channels by isotropic frequency-selective filters that originated with Coggins [19]. The modulation transfer function (MTF) of these filters is defined in (2.5), which is repeated here for convenience.

$$H(u,v) = \exp\left\{-\frac{1}{2}\frac{\left(\ln\sqrt{u^2+v^2}-\ln\mu\right)^2}{\sigma_1^2}\right\}, \qquad (u,v) \neq (0,0),$$

Again, μ is the center radial frequency and σ_1 determines the bandwidth of the filter. Note that these filters are defined on a logarithmic scale.

We use $\sigma_1 = 0.275$ for all filters. This results in a bandwidth of about one octave which is close to the estimated bandwidth of simple cells in the mammalian visual cortex. (See Section 3.1.) Also, we set the value of MTFs at $(u,v) = (0,0)$ to zero so that the mean gray values of the filtered images are zero. (That is, we block the DC component.)

We address the problem of determining the appropriate values for the center frequencies of the filters by considering a large, but finite, number of center frequencies. Specifically, we consider a set of filters whose center frequencies are one half octave apart. The number of the filters considered depends on the size of the input image. For a 256×256 image, for example, we shall consider a total of fourteen frequency-selective filters tuned to 1, $1\sqrt{2}$, 2, $2\sqrt{2}$, 4, $4\sqrt{2}$, 8, $8\sqrt{2}$, 16, $16\sqrt{2}$, 32,

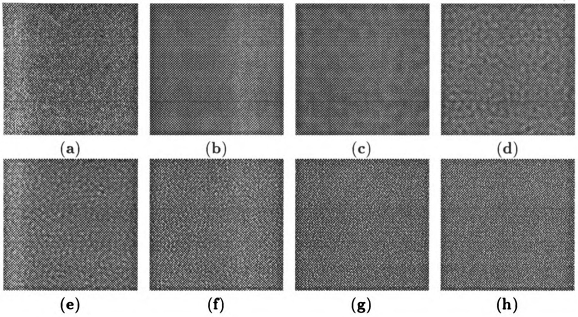


Figure 5.4: (**a**) A 256 × 256 image of a metallic finish sample. (**b** – **h**) Filtered images corresponding to frequency-selective filters with center frequencies at 4, 8, 16, $16\sqrt{2}$, 32, $32\sqrt{2}$, and 64 cycles/image-width.

$32\sqrt{2}$, 64, and $64\sqrt{2}$ cycles/image-width. This choice of center frequencies for the filters provides a nearly uniform coverage of the spatial-frequency domain. Any significant range of spatial-frequencies in the input image should, practically, fall in the passband of one of these filters. In Section 5.4, we will describe procedures that allow us to determine which subset of filters is best suited for a given set of finish samples.

The filtering operations are again carried our using a fast Fourier transform (FFT). Figure 5.4 shows an image of a finish sample along with some of the filtered images. The ability of the filters to exploit differences in spatial-frequency (size) is evident in these filtered images.

5.3.2 Texture Features

The texture features are defined as the average absolute deviation (AAD) in the filtered images. The texture feature f_j for the j^{th} (zero mean) filtered image $r_j(x, y)$ is computed as follows.

$$f_j = \frac{1}{N_r N_c} \sum_{a=1}^{N_r} \sum_{b=1}^{N_c} \mid r_j(a, b) \mid, \tag{5.2}$$

where N_r and N_c are the number of rows and columns in the image. Each filtered image is, therefore, 'summarized' by one feature, and there are as many features as filtered images. With a total of ten filters, for example, we will have ten texture features, resulting in a ten-dimensional feature vector for each image. These feature vectors will be used in grading the texture uniformity of panels using texture grading schemes described in Section 5.4.

In Section 5.2.1, we described an image normalization operation. This normalization can be achieved equivalently by dividing each texture feature by the AAD measure in the input image defined by (5.1). Formally, the normalized feature f'_j corresponding to texture feature f_j is then given by

$$f'_j = f_j / f_0, \qquad i = 1, \cdots, 14,$$

where f_0 is the AAD measure of the input image. In the following sections, for convenience, we will refer to the *normalized* texture features as f_1, f_2, etc. Also, we will not use features f_1 and f_2 in the grading experiments. These features correspond to filters that respond to very slow intensity variations. Such variations, however, are very likely to result from variations in the lighting rather than from finish texture.

5.4 Grading Finish Texture Uniformity

How can the above texture features be used to 'grade' the degree of uniformity of the finish texture? In this section, we propose two alternative grading schemes to achieve this goal.

5.4.1 Reference-Based Grading

Our first grading scheme can be summarized as follows. Given a set of panels, we use a few panels with extreme appearances as 'reference panels'. These reference panels are, in a sense, our training samples. Since finish samples with highly uniform or highly non-uniform finishes are easier to identify, these reference panels can be selected with very high confidence. In our experiments, we typically use the two panels with lowest visual scale values and the two panels with the highest visual scale values in each set. Table 5.3 lists the least- and most-uniform panels for the LBLUE and the MBLUE sets. Using the feature vectors corresponding to images from these

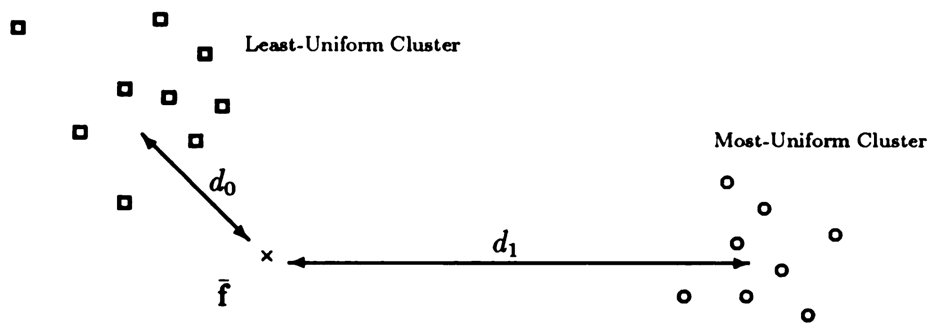


Figure 5.5: Illustration of reference-based grading in a two-dimensional feature space. The mean feature vector $\bar{\mathbf{f}}$ of the panel to be graded is shown as 'x'.

four panels we construct a *least-uniform* and a *most-uniform* 'reference cluster'. We then assign a texture uniformity grade to each panel based on the "distances" of its mean feature vector to these reference clusters.

Formally, let $\mathbf{f}_i$ denote the feature vector for the i^{th} image from a panel. We represent each panel by its mean feature vector

$$\bar{\mathbf{f}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{f}_i, \tag{5.3}$$

where n is the number of images taken from a panel. In our experiments $n = 8$.

Let d_0 and d_1 be the distances between the mean feature vector $\bar{\mathbf{f}}$ and the least-uniform and most-uniform clusters, respectively. (See Figure 5.5.) The distance of a point from a cluster (of points) can be defined in several different ways. Here, we simply use the Euclidean distance between the point and the centroid (mean) of the cluster[3]. We define the texture uniformity grade γ for the panel by the following ratio.

$$\gamma = \frac{d_0}{d_0 + d_1} \tag{5.4}$$

Note that γ lies between 0 and 1. A value of γ close to 1 indicates that the corresponding panel can be classified as uniform.

To get an idea of the discrimination provided by individual features, we show the box plot of the patterns in the reference clusters. These plots for the LBLUE and MBLUE sets are given in Figures 5.6 and 5.7, respectively. In the box plot, the horizontal line inside a box marks the location of the median. The box contains the

[3]We also experimented with the Mahalanobis distance measure. Since the grading results were essentially the same, we only report the results based on the Euclidean distance.

Table 5.3: Panels that were used as references when grading the finish texture uniformity of the LBLUE and the MBLUE sets. Each set contains 13 panels.

Set		Reference Panels	
No.	Abb. Name	Least-Unif.	Most-Unif.
1	LBLUE	2, 9	10, 13
2	MBLUE	5, 9	10, 3

middle half of the data. The extent of the whiskers reflects a confidence interval for the median, and outlying points (outliers) are plotted individually. One interesting observation to be made in these plots is the general "behavior" or "trend" in the feature values. Patterns from the most-uniform reference cluster have smaller feature values at lower frequencies (features $f3$ through $f9$) than the patterns from the least-uniform reference cluster. The opposite situation is true at higher frequencies (features $f10$ through $f14$). This observation is consistent with the physical interpretation of the frequency-selective filters — less uniform finish textures are richer in low frequency components (have larger spatial variations in intensity) than more uniform finish textures, and vice versa.

Feature Selection

Which subset of the texture features should we use when grading a given set of panels? Recall that there is a one-to-one correspondence between the texture features and the filters. Selecting a subset of features, therefore, is equivalent to selecting a subset of filters. Here we describe feature selection experiments that are based on maximizing the rank correlation between the visual scale and the texture uniformity grade given by (5.4). The rank correlation is given by

$$r = \frac{\sum_{i=1}^{n}(R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum_{i=1}^{n}(R_i - \bar{R})^2 \sum_{i=1}^{n}(S_i - \bar{S})^2}} = \frac{\sum_{i=1}^{n} R_i S_i - n\left(\frac{n+1}{2}\right)^2}{\frac{n(n^2-1)}{12}}, \tag{5.5}$$

where R_i and S_i are, respectively, the rank of texture uniformity grade γ_i (among γ's) and the rank of visual scale value v_i (among v's), and n is the total number of panels. Note that unlike correlation, which measures the linear association, the rank correlation measures the *monotone* association between two sets of data.

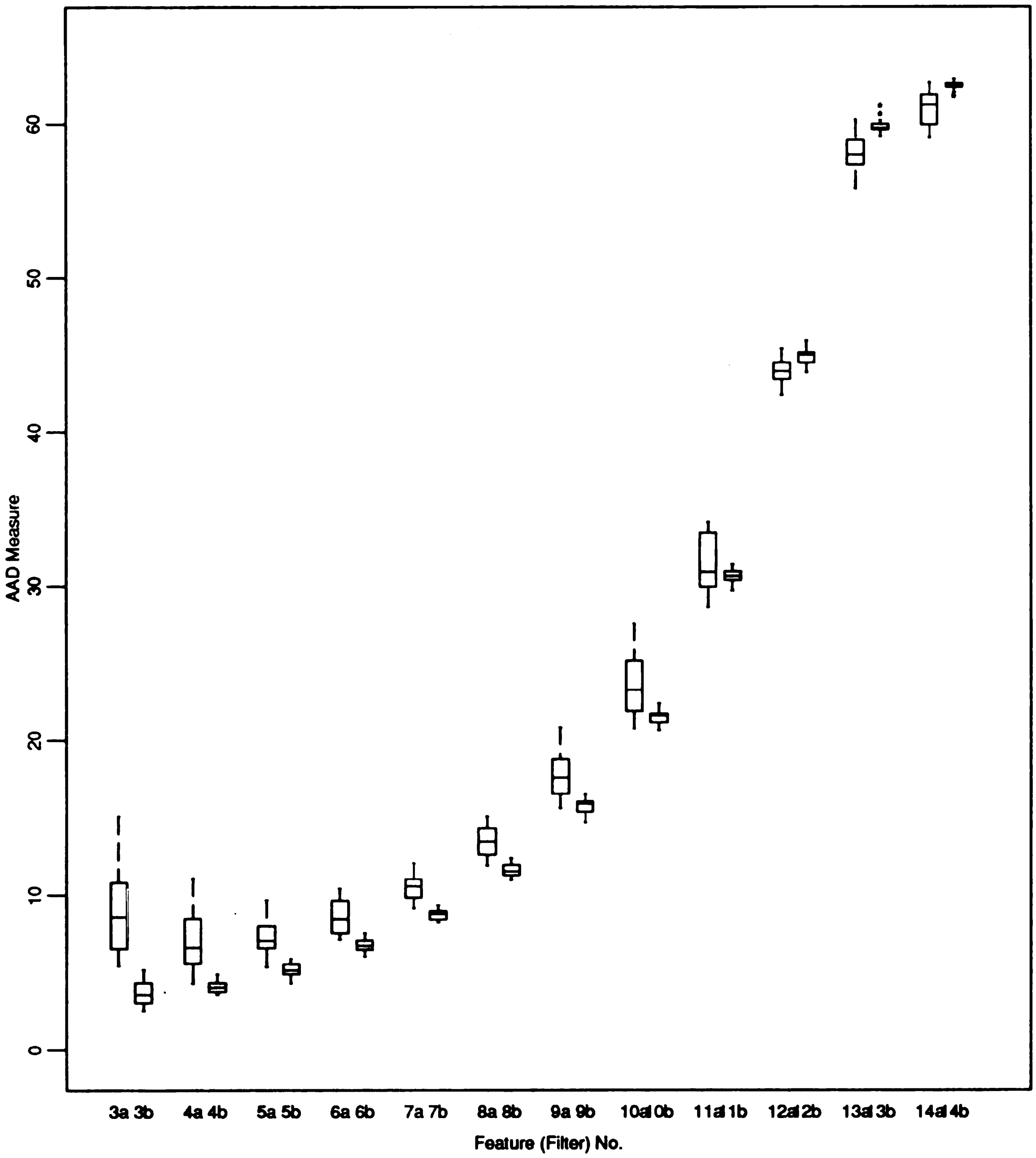


Figure 5.6: Box plot of reference patterns used in grading the texture uniformity of panels in the LBLUE set. There are 16 patterns in each cluster. The “a” and “b” suffixes indicate least-uniform and most-uniform clusters, respectively.

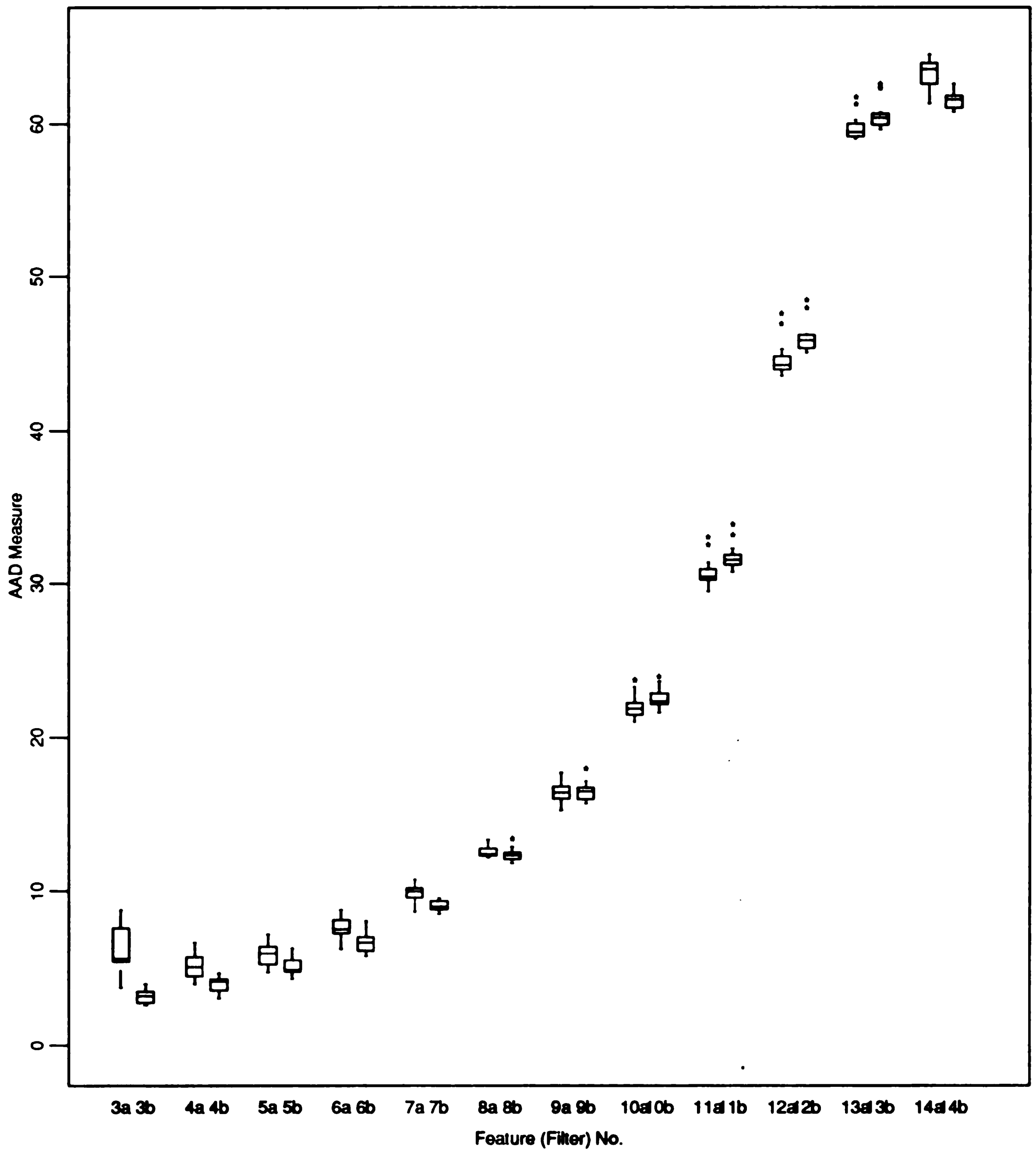


Figure 5.7: Box plot of reference patterns used in grading the texture uniformity of panels in the MBLUE set. There are 16 patterns in each cluster. The “a” and “b” suffixes indicate least-uniform and most-uniform clusters, respectively.

Our feature selection procedure can be summarized as follows. Using a texture grading scheme we assign a texture grade to each panel in a given set. We then compute the rank correlation between the texture grade and the visual scale value. We repeat this procedure for every subset of texture features. We then choose the feature subset that results in the highest rank correlation as the "best" feature subset.

Starting with all 12 texture features (corresponding to filters with 2 through $64\sqrt{2}$ cycles/image-width center frequencies) we performed exhaustive feature selection. The best feature subsets for the LBLUE and the MBLUE sets of panels, along with the corresponding rank correlations, are given in Tables 5.4 and 5.5, respectively. As expected, occasionally, there were ties between different feature subsets. These ties were resolved based on the *direct* correlation between the texture grade and the visual scale value. That is, we chose the feature subset with a higher correlation.

The highest rank correlation for the LBLUE set is is 0.98, and is achieved by feature subsets $\{f5, f14\}$ and $\{f5, f12, f14\}$. The highest rank correlation for the MBLUE set, on the other hand, is 0.91 and is achieved by feature subset $\{f3, f7, f10, f11\}$. Even though, the LBLUE and MBLUE sets of panels have different colors, one would expect that the best feature subsets for both sets of panels to be the same. By comparing the feature subsets of size 4 for both sets of panels, we looked for a common subset that could be used for grading both sets of panels. The feature subset $\{f3, f7, f9, f11\}$ results in a rank correlation of 0.95 for the LBLUE set and 0.89 for the MBLUE set. Therefore, although the "best" feature subsets for the two sets of panels are not the same, there does exist a feature subset which results in acceptable performance for both sets.

Table 5.4: Results of reference-based grading of finish texture uniformity for the LBLUE set. This table shows the "best" feature subsets of size 1–7 and corresponding rank correlations between texture grade and visual scale.

Size	Best Subset	Rank Correlation
1	{f5}	0.91
2	{f5, f14}	0.98
3	{f5, f12, f14}	0.98
4	{f5, f6, f12, f14}	0.97
5	{f5, f6, f7, f12, f14}	0.97
6	{f4, f6, f7, f12, f13, f14}	0.96
7	{f4, f5, f6, f7, f12, f13, f14}	0.96

Table 5.5: Results of reference-based grading of finish texture uniformity for the MBLUE set. This table shows the "best" feature subsets of size 1–7 and corresponding rank correlations between texture grade and visual scale.

Size	Best Subset	Rank Correlation
1	{f10}	0.81
2	{f3, f10}	0.89
3	{f5, f10, f13}	0.89
4	{f3, f7, f10, f11}	0.91
5	{f3, f7, f10, f11, f13}	0.91
6	{f3, f7, f9, f10, f11, f13}	0.91
7	{f3, f6, f7, f10, f11, f13, f14}	0.88

5.4.2 Regression-Based Grading

Now we propose a different approach to relate the visual scale values for finish texture uniformity to the texture features. This alternative grading scheme is based on the classical linear regression model. Unlike our previous texture grading scheme, which uses only the panels with extreme appearance qualities as 'training' samples, in the following regression-based grading scheme we will use all the panels in a given set for estimating the parameters of the regression models.

In the regression model, we treat the visual scale for texture uniformity as the *dependent* variable, and the texture features computed from the filtered images as *independent* (or *predictor*) variables. Specifically, let v be the visual scale for a panel and f_i be the texture features associated with the panel. (Recall that texture features for a panel are obtained by averaging the texture features for all eight images corresponding to the panel.) Then the linear regression model is given by

$$v = \beta_0 + \beta_1 f_1 + \beta_2 f_2 + \cdots + \beta_r f_r + \epsilon, \tag{5.6}$$

where ϵ accounts for random measurement error or the effects of other variables not explicitly considered in the model. We estimate the *regression coefficients*, β_j, using the method of least squares. Let the least square estimates of these coefficients be $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_r$. Then the predicted visual scale values are given by

$$\hat{v} = \hat{\beta}_0 + \hat{\beta}_1 f_1 + \hat{\beta}_2 f_2 + \cdots + \hat{\beta}_r f_r. \tag{5.7}$$

The quality of the fit can be measured by the *coefficient of determination* (COD) which is given by

$$R^2 = 1 - \frac{sse}{sstot}, \tag{5.8}$$

where $sse = \sum_{j=1}^{n} (v_j - \hat{v}_j)^2$, $sstot = \sum_{j=1}^{n} (v_j - \bar{v})^2$, and n is the total number of observations. This quantity is an indicator of the proportion of the total variation in the v_j explained by predictor variables. In the process of deciding which subset of features explains the visual scale values better, we will compare regression models with different number of predictors. In order to be able to compare these models with one another, we use the *adjusted* coefficient of determination which is given by

$$R^2_{adj} = 1 - \frac{sse/(n-p)}{sstot/(n-1)}, \tag{5.9}$$

where p is the total number of parameters (including β_0) in the fitted model [31].

Model Selection

In the following experiments, we continue to use the texture features computed from images normalized by their AAD measure. Although examining the regression models corresponding to *all* possible subsets of features is computationally demanding, the required computations in this case were not prohibitive. The "best" regression model, and hence the "best" feature subset, is determined as follows. First, we determine the best regression model for a given size of the feature subset. The criterion for best model is to maximize the COD (R^2). Among these feature subsets, the subset with the highest adjusted COD (R^2_{adj}) is singled out as the best model. This model can then be used for grading (predicting) the degree of uniformity of finish texture for future samples.

Tables 5.6 and 5.7 show the feature subsets of size 1 through 7 from the set $\{f3, \ldots, f14\}$ that give the best regression models for the LBLUE and MBLUE sets, respectively. As seen in Table 5.6, for example, the adjusted COD first tends to increase as more variables are included, but it begins to flatten when more and more variables are used. Note that the number of samples used for estimating the regression coefficients is small (12 or 13). The estimated coefficients for regression models with a large number of independent variables is, therefore, not very reliable. Also, our reference-based grading scheme indicated that a feature subset of size 4 results in acceptable performance. We will, therefore, consider models with no more than 4 independent variables.

Based on the above constraints, the best regression model for the LBLUE set is found to be

$$\hat{v} = 13.859 - 1.5315\, f_3 + 3.1389\, f_4 - 2.1269\, f_5 - 0.4014\, f_8 \tag{5.10}$$

The best regression model for the MBLUE set, on the other hand, is found to be

$$\hat{v} = 82.9490 - 3.3293\, f_7 + 3.3342\, f_8 - 1.4469\, f_{10} - 0.9514\, f_{14} \tag{5.11}$$

Note that the best regression models for the LBLUE and the MBLUE sets are not the same. Looking for a common regression model that gives acceptable performance for both sets of finish samples, we found the feature subset $\{f3, f4, f6, f13\}$. The corresponding regression model has a COD of 96.93% for the LBLUE set and 88.60% for the MBLUE set. We may, therefore, use the same subset of features for grading both sets of panels.

Table 5.6: Results of regression-based grading of finish texture uniformity for the LBLUE set. This table shows selected variables (texture features) for regression models with 1–7 variables and corresponding coefficients of determination.

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f5}	84.79	83.27
2	{f3, f6}	91.75	89.91
3	{f3, f4, f6}	96.92	95.77
4	{f3, f4, f5, f8}	98.37	97.44
5	{f3, f4, f5, f10, f11}	98.99	98.16
6	{f3, f4, f5, f10, f11, f12}	99.30	98.47
7	{f3, f4, f5, f7, f8, f9, f13}	99.35	98.22

Table 5.7: Results of regression-based grading of finish texture uniformity for the MBLUE set. This table shows selected variables (texture features) for regression models with 1–7 variables and corresponding coefficients of determination.

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f3}	56.17	52.18
2	{f3, f4}	70.38	64.46
3	{f3, f11, f14}	83.10	77.47
4	{f7, f8, f10, f14}	93.32	89.99
5	{f7, f10, f12, f13, f14}	96.91	94.71
6	{f3, f7, f10, f12, f13, f14}	97.87	95.74
7	{f4, f5, f6, f11, f12, f13, f14}	98.70	96.88

Grading Examples

Since there are only a small number of panels in each set we have to use all of them for parameter estimation. If we had more panels in each set, we could have used a subset of them to estimate the parameters of the regression model and use the rest to validate the model. An alternative strategy is to obtain a regression model by using the samples in one set and then use the model to predict the visual scale values for another set. We have to remember, however, that since each set of panels was evaluated separately, the visual scale values for the two sets are not on the same scale. In fact, the range of visual scale values for the LBLUE set is wider than that

Table 5.8: Grading MBLUE set using regression model for LBLUE set. Correlation = 0.87, Rank Correlation = 0.88.

Panel No.	Actual Value	Predicted Value
1	1.10	3.7
2	0.61	1.8
3	2.32	5.0
4	2.04	4.5
5	0.00	1.6
6	1.36	3.3
7	1.12	3.5
8	1.77	4.2
9	0.13	2.1
10	3.42	5.2
11	2.29	4.1
12	0.43	3.2
13	0.85	4.1

of the MBLUE set (see second columns in Tables 5.1 and 5.2). Therefore, the rank correlation between predicted and actual visual scale values is perhaps a more suitable figure of merit than the direct correlation between them.

In the following two grading examples, we used the feature subset $\{f3, f4, f6, f13\}$. Using the visual scale values for texture uniformity for panels in the LBLUE set, we first estimated the parameters of the regression model. We then obtained the predicted visual scales for panels in the MBLUE set using this model. The actual and predicted visual scale values are tabulated in Table 5.8. The correlation and rank correlation between predicted and actual scales are 0.87 and 0.88, respectively.

Similarly, we used the visual scale values for panels in the MBLUE set to estimate the parameters of the regression model. We then obtained the predicted visual scale values for panels in the LBLUE set. The results are given in Table 5.9. The correlation and the rank correlation between predicted and actual scales are 0.76 and 0.74, respectively. These results indicate that our regression-based grading scheme is fairly robust.

Table 5.9: Grading LBLUE set using regression model for MBLUE set. Correlation = 0.76, Rank Correlation = 0.74.

Panel No.	Actual Value	Predicted Value
1	3.02	1.40
2	0.21	-0.75
3	3.62	1.78
4	2.14	-0.44
5	2.43	1.28
6	1.32	0.80
7	1.04	0.25
8	3.37	0.38
9	0.25	0.88
10	4.80	2.56
12	1.40	0.78
13	4.28	3.11

5.4.3 Mottle, Flake-Size, and Blotchy Components

We now consider additional visual scale values that rank the appearance of the finish samples along other components. In this section we will use the linear regression setting described to obtain best regression models explaining *mottle, flake-size, and blotchy* components of the appearance of the metallic finish samples. Our criterion for the best regression model is also the same — i.e., to maximize R^2_{adj}.

Tables 5.10 and 5.11 give the best feature subsets of size 1 through 7 that result in the best regression models, for the LBLUE and MBLUE sets, respectively. The best feature subsets for the LBLUE and the MBLUE sets are $\{f3, f4, f6, f9\}$ and $\{f3, f4, f5, f13\}$ and they explain 97.50% and 85.49% of the variations in the visual scales for mottle appearance, respectively. Based on these experiments, the best regression model explaining the mottle component of panels in the LBLUE set is

$$\hat{v} = -12.61 + 2.0233\, f_3 - 2.9597\, f_4 + 1.6562\, f_6 + 0.5511\, f_9 \qquad (5.12)$$

The best regression model for the MBLUE set, on the other hand, is

$$\hat{v} = -149.079 + 4.713\, f_3 - 13.042\, f_4 + 9.861\, f_5 + 2.309\, f_{13} \qquad (5.13)$$

The feature subsets resulting in the best regression models explaining the 'flake-size' component of finish appearance of panels in the LBLUE and MBLUE sets are given in Tables 5.12 and 5.13, respectively. Similar results for 'blotchy' components of finish appearance are given in Tables 5.14 and 5.15.

Table 5.10: "Best" feature subsets of size 1–7 from the set $\{f3, \ldots, f14\}$ explaining the visual scales for the 'mottle' component of finish texture appearance for panels in the LBLUE set.

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f6}	76.11	73.72
2	{f3, f9}	93.32	91.84
3	{f3, f10, f12}	96.40	95.06
4	{f3, f4, f6, f9}	97.50	96.07
5	{f3, f4, f5, f9, f12}	98.99	98.15
6	{f3, f4, f7, f8, f9, f11}	99.61	99.13
7	{f3, f4, f6, f7, f8, f9, f10}	99.86	99.62

Table 5.11: "Best" feature subsets of size 1–7 from the set $\{f3, \ldots, f14\}$ explaining the visual scales for the 'mottle' component of finish texture appearance for panels in the MBLUE set.

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f7}	47.92	43.19
2	{f7, f14}	69.26	63.12
3	{f3, f4, f5}	78.85	71.80
4	{f3, f4, f5, f13}	85.49	78.23
5	{f8, f9, f12, f13, f14}	89.18	81.45
6	{f3, f8, f9, f12, f13, f14}	91.76	83.52
7	{f3, f8, f9, f11, f12, f13, f14}	94.70	87.29

Table 5.12: **"Best" feature subsets of size 1–7 from the set $\{f3, \ldots, f14\}$ explaining the visual scales for the 'flake-size' component of finish texture appearance for panels in the LBLUE set.**

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f3}	46.02	40.63
2	{f3, f8}	73.76	67.93
3	{f3, f6, f7}	77.74	69.40
4	{f3, f8, f11, f14}	87.40	80.20
5	{f3, f8, f10, f13, f14}	95.47	91.69
6	{f3, f6, f7, f9, f12, f14}	99.51	98.93
7	{f3, f4, f6, f7, f9, f12, f14}	99.64	99.00

Table 5.13: **"Best" feature subsets of size 1–7 from the set $\{f3, \ldots, f14\}$ explaining the visual scales for the 'flake-size' component of finish texture appearance for panels in the MBLUE set.**

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f3}	62.38	58.96
2	{f3, f8}	89.58	87.50
3	{f3, f4, f8}	93.34	91.13
4	{f7, f8, f10, f12}	94.09	91.14
5	{f6, f7, f8, f10, f12}	96.07	93.27
6	{f6, f7, f8, f10, f11, f12}	96.36	92.72
7	{f4, f6, f7, f8, f10, f11, f12}	96.67	92.02

Table 5.14: "Best" feature subsets of size 1–7 from the set $\{f3, \ldots, f14\}$ explaining the visual scales for the 'blotchy' component of finish texture appearance for panels in the LBLUE set.

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f5}	78.56	76.42
2	{f3, f7}	90.14	87.95
3	{f5, f12, f13}	96.80	95.60
4	{f5, f9, f11, f12}	97.85	96.62
5	{f3, f4, f5, f13, f14}	99.34	98.79
6	{f3, f4, f5, f12, f13, f14}	99.64	99.21
7	{f3, f4, f7, f8, f10, f11, f14}	99.82	99.51

Table 5.15: "Best" feature subsets of size 1–7 from the set $\{f3, \ldots, f14\}$ explaining the visual scales for the 'blotchy' component of finish texture appearance for panels in the MBLUE set.

Size	Best Subset	R^2 %	R^2_{adj} %
1	{f7}	34.52	28.57
2	{f7, f14}	70.87	65.04
3	{f7, f12, f14}	72.57	63.43
4	{f7, f12, f13, f14}	75.28	62.92
5	{f8, f10, f12, f13, f14}	83.93	72.45
6	{f7, f8, f10, f12, f13, f14}	90.60	81.20
7	{f3, f8, f9, f11, f12, f13, f14}	91.81	80.33

5.5 Summary

In this chapter, we addressed a practical problem where texture analysis is required. The problem involved automated visual inspection of the textural appearance of automotive metallic finishes. We addressed imaging and preprocessing requirements and demonstrated that a multi-channel filtering technique can be used to successfully characterize the finish texture. We also developed two alternative methods for grading the degree of uniformity of the finish texture. Our 'texture grading' experiments showed that there is a high correlation between our texture uniformity grade and the visual scaling of the finish samples by finish inspection experts.

Non-uniform illumination of the panels due to their specular nature, and resolution requirements forced us to acquire multiple images from small areas on the panel surface. Clearly, it is more desirable to have a single image acquired from the entire, or a significant portion, of the panel surface. When judging the appearance of a finish sample, human observers are more likely to base their judgment on simultaneous examination of the entire panel. Currently, the resolution of most 2-D sensors are limited to 1024 $\times$ 1024 pixels. For imaging larger areas, therefore, one will need to scan the panels with a 1-D sensor array (along with a linear light source).

The filtering and feature computation operations can be performed in parallel, regardless of the number of filters. Therefore, a fast, real-time implementation of our grading techniques is possible. Moreover, our grading experiments showed that only a small number of filters is sufficient. The results of our feature selection experiments indicate that using a small number of features is not only possible, but also leads to improved performance. Moreover, these results indicate that a common feature subset can give acceptable performance across different sets of metallic finish samples.

In our texture grading experiments, we represented each finish sample by the mean feature vector for all eight images from the panel. This representation assumes that the variation of texture features across images are negligible. However, when grading texture uniformity, the variation of texture features across the panel could be a good indicator of degree of uniformity of finish texture. That is, larger variations would indicate that the texture is less uniform. This approach to grading finish texture uniformity should be examined.

Chapter 6

Conclusions and Future Research

Texture analysis has been an active research area in computer vision for more than two decades, and has proved to be a very difficult problem. This difficulty largely stems from the diversity of natural and artificial textures, which makes a universal definition of texture impossible. Compared to other approaches, the multi-channel filtering approach to texture analysis is more general and applies to a larger class of textures. This generality is a direct consequence of reliance on basic attributes of frequency (size) and orientation, and the inherent multi-resolution nature of the approach.

In this dissertation, we presented several multi-channel filtering techniques. Major contributions have been:

1. A detailed methodology for modeling the 'channels' by even-symmetric Gabor filters, and a systematic filter selection scheme based on an intuitive least-squares criterion.

2. A simple but general methodology for extracting texture features using a non-linear transformation and local "energy" computation.

3. Incorporating spatial adjacency information in the region-based texture segmentation algorithm.

4. An edge-based texture segmentation technique based on combining the "evidence" for texture boundaries in different feature images.

5. Integrating the region- and edge-based texture segmentation algorithms and eliminating the need for knowing the "true" number of texture categories.

6. Application of a multi-channel filtering technique to automated visual inspection of automotive metallic finishes.

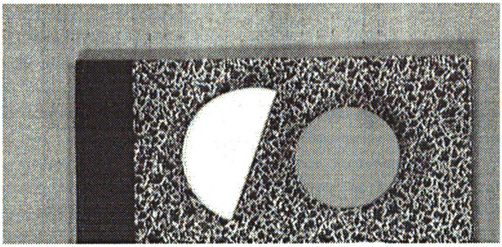
7. Reference-based and regression-based methodologies for grading the degree of uniformity of metallic finish texture.

We reported both unsupervised and supervised texture segmentation experiments. In the supervised segmentation experiments we used a feed-forward neural network, in addition to several other classifiers. The texture segmentation experiments showed that our texture features can discriminate among a large number of textures, including some artificially generated texture pairs with identical second- and third-order statistics.

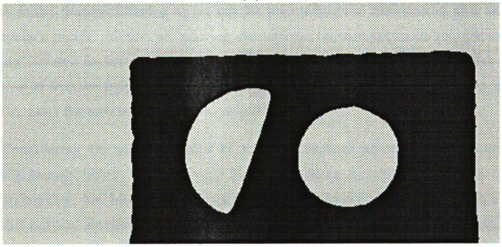
One limitation of our definition of texture features is the lack of a criterion for deciding the optimal value of α, which controls the severity of the threshold-like nonlinear transformation in (3.10). In the texture segmentation experiments, we used a *fixed* empirical value. However, the optimal value of α is likely to be different for different channels, and for different images. One should consider using the statistical properties of the input image and the filtered images to estimate the optimal values [10, 92]. Also, we assume that different channels are independent from each other. However, there is psychophysical and physiological evidence for inhibitory interactions between different spatial-frequency channels [29]. Allowing inhibitory interactions among the channels is shown to have the potential to reduce the effective dimensionality of the feature space [8].

Our texture segmentation techniques are only applicable to textured images. They are unable to discriminate between (nearly) uniform gray-level regions. However, in real world images, both textured regions as well as regions with nearly uniform gray levels are often present simultaneously. An extension of the current techniques that allows handling such images would be highly desirable. A simple approach would be to add low-pass Gaussian filters with different cutoff frequencies to the original Gabor filter set. Note that these low-pass filters can be viewed as Gabor filters with zero center frequencies. However, we would like to point out that, in some applications, it might be desirable to separate nearly-uniform (untextured) regions from textured regions. Figure 6.1(a) shows the top view of a scene with a notebook (textured cover) and two flat "tub blocks" with different colors — pink and blue. As seen in Figure 6.1(b), the algorithm successfully separates the textured region from untextured regions. Such a property may also be useful for separating text from "non-text" in automated document analysis applications [34]. When viewed at an appropriate resolution, the text forms a distinct texture of its own, allowing it to be discriminated from photographs and other non-text items in the document image.

In its current form, the edge-based texture segmentation technique in Chapter 4



(a)



(b)

Figure 6.1: (**a**) A 256 × 512 image of a scene containing both textured and untextured objects. (**b**) Two-category segmentation obtained using a total of 16 Gabor filters and the pixel coordinates.

has certain limitations. The process of adding the magnitude responses of different channels enhances some texture boundaries more than others. This is particularly true when there are a large number of textures in the image. An adaptive hysteresis thresholding, rather than the current global thresholding method, may alleviate this problem. However, we believe that a different method of combining the magnitude responses should be investigated.

In Chapter 5, the regression-based method for grading the degree of uniformity of metallic finish texture is more rigorous than the reference-based method. This method, however, requires more training data. In the experiments, we used the finish samples from one set to estimate the parameters of the regression model and used samples from the second set to validate it. Further evaluation of the regression-based grading method using larger sets of finish samples is recommended.

In addition to improvements/extensions suggested in the previous paragraphs, we have identified the following future research problems in texture analysis in general, and in multi-channel filtering approach in particular.

- A multi-channel filtering approach for estimating the orientation of a textured surface patch. Almost all existing shape-from-texture methods require extracting *isolated* texture primitives, e.g. blobs. Using the existing, or a similar definition of texture features we should be able to measure texture gradients *without* the need for extracting isolated primitives.

- Considering the survival value of some perceptual processes for animals and the human being, it seems very likely that some processes, including texture perception, are biologically adapted to solve specific visual tasks that arise in the natural environment. The feasibility of encoding higher-level knowledge of the scene, for example shape of the regions or boundaries, in the low-level texture processing must be explored.

- Color vision has become an active research area in computer vision. Integrating texture and color information, therefore, is another potential research area in texture analysis.

Appendix

Appendix A

Generating Filter Functions

In this appendix we describe the implementation of the Gabor filters used by our texture segmentation algorithm. We perform the filtering operations through multiplication in the Fourier domain, rather than through convolution. Therefore, we implement each Gabor filter by sampling its Fourier transform. This direct implementation is faster than sampling the spatial domain representation of the filter and then computing its DFT. Furthermore, direct sampling in the Fourier domain avoids possible distortions due to aliasing.

In order to generate a filter function in a rectangular array, prior to sampling, one must scale the Fourier transform of the 2-D function along one of the axes. This can be demonstrated as follows. Let $G(u,v)$ be the DFT of a discrete realization of a 2-D function, $g(x,y)$, in an $N_r \times N_c$ array, where N_r and N_c are the number of rows and columns, respectively. That is,

$$G(u,v) = \sum_x \sum_y g(x,y) \exp\left\{-\jmath 2\pi (\frac{ux}{N_c} + \frac{vy}{N_r})\right\} , \tag{A.1}$$

where u is in cycles/image-width and v is in cycles/image-height. We can rewrite (A.1) as follows.

$$G(u,v') = \sum_x \sum_y g(x,y) \exp\left\{-\jmath 2\pi \frac{ux + v'y}{N_c})\right\} , \tag{A.2}$$

where $v' = (N_c/N_r)v$. Here, both u and v' are in cycles/image-width. Thus, a discrete realization of an even-symmetric Gabor filter in an $N_r \times N_c$ array is obtained by uniformly sampling the following continuous function.

$$\begin{aligned} H(u,v) = \exp&\left\{-\frac{1}{2}\left[\frac{(u-u_0)^2}{\sigma_u^2} + \frac{((N_c/N_r)v)^2}{\sigma_v^2}\right]\right\} + \\ \exp&\left\{-\frac{1}{2}\left[\frac{(u+u_0)^2}{\sigma_u^2} + \frac{((N_c/N_r)v)^2}{\sigma_v^2}\right]\right\} \end{aligned} \tag{A.3}$$

Note that the center radial frequency of this Gabor filter is u_0 cycles/image-width and its center orientation is 0. A Gabor filter with center orientation of θ_0 is obtained by a rigid rotation of (A.3) prior to sampling. Also, note that the choice of cycles/image-width as the unit for measuring the frequency is arbitrary and that one can use cycles/image-height as well. Similar scaling applies to the direct implementation of other filter functions in the spatial-frequency domain, including the isotropic frequency-selective filters used in Chapter 5.

Bibliography

Bibliography

[1] N. Ahuja and A. Rosenfeld. Mosaic models for textures. *IEEE Trans. Pattern Anal. Machine Intell.*, 3(1):1–11, 1981.

[2] J. Beck, A. Sutter, and R. Ivry. Spatial frequency channels and perceptual grouping in texture segregation. *Computer Vision, Graphics, Image Process.*, 37:299–325, 1987.

[3] J. Besag. On the statistical analysis of dirty pictures. *J. Royal Stat. Soc., Series B*, 48(3):259–302, 1986.

[4] D. Blostein and N. Ahuja. Shape from texture: Integrating texture-element extraction and surface estimation. *IEEE Trans. Pattern Anal. Machine Intell.*, 11(12):1233–1251, 1989.

[5] A. C. Bovik. Properties of multichannel texture analysis filters. In *Proc. IEEE Int. Conf. on Acoust., Speech, Signal Process.*, pages 2133–2136, Albuquerque, New Mexico, April 1990.

[6] A. C. Bovik, M. Clark, and W. S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(1):55–73, 1990.

[7] P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover, New York, 1966.

[8] T. M. Caelli. An adaptive computational model for texture segmentation. *IEEE Trans. Syst., Man, Cybern.*, 18(1):9–17, 1988.

[9] F. W. Campbell and J. G. Robson. Application of Fourier analysis to the visibility of gratings. *J. Physiology*, 197:551–566, 1968.

[10] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Machine Intell.*, 8(6):679–698, 1986.

[11] G. A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, Image Process.*, 37:54–115, 1987.

[12] G. A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. In S. Grossberg, editor, *Neural Networks and Natural Intelligence*, pages 251–315. MIT Press, Cambridge, MA, 1987.

[13] R. Chellappa. Two-dimensional discrete Gaussian Markov random field models for image processing. In L. N. Kanal and A. Rosenfeld, editors, *Progress in Pattern Recognition 2*, pages 79–112. Elsevier Science, 1985.

[14] R. Chellappa, S. Chatterjee, and R. Bagdazian. Texture synthesis and compression using Gaussian-Markov random field models. *IEEE Trans. Syst., Man, Cybern.*, 15(2):298–303, 1985.

[15] R. T. Chin. Algorithms and techniques for automated visual inspection. In T. Y. Young and K. S. Fu, editors, *Handbook of Pattern Recognition and Image Processing*, pages 587–612. Academic Press, 1986.

[16] C. C. Chu and J. K. Aggarwal. The integration of region and edge-based segmentation. In *Proc. International Conference on Computer Vision*, pages 117–120, Osaka, Japan, December 1990.

[17] P. Cielo. *Optical Techniques for Industrial Inspection.* Academic Press, 1988.

[18] M. Clark and A. C. Bovik. Experiments in segmenting texton patterns using localized spatial filters. *Pattern Recognition*, 22(6):707–717, 1989.

[19] J. M. Coggins. *A Framework for Texture Analysis Based on Spatial Filtering.* PhD thesis, Dept. of Computer Science, Michigan State University, East Lansing, MI 48824-1027, 1982.

[20] J. M. Coggins and A. K. Jain. A spatial filtering approach to texture analysis. *Pattern Recognition Letters*, 3(3):195–203, 1985.

[21] J. M. Coggins and A. K. Jain. Surface orientation from texture. In *Proc. IEEE Int. Conf. on Syst., Man, Cybern.*, pages 1617–1620, Atlanta, GA, 1986.

[22] Y. Cohen and M. S. Landy. The hips picture processing software. Technical report, Psychology Dept., New York University, 1982.

[23] R. W. Conners, C. W. McMilin, K. Lin, and R. E. Vasquez-Espinosa. Identifying and locating surface defects in wood: Part of an automated lumber processing system. *IEEE Trans. Pattern Anal. Machine Intell.*, 5(6):573–583, 1983.

[24] G. R. Cross and A. K. Jain. Markov random field texture models. *IEEE Trans. Pattern Anal. Machine Intell.*, 5(1):25–39, 1983.

[25] J. G. Daugman. Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Research*, 20:847–856, 1980.

[26] J. G. Daugman. Uncertainty relation for resolution in space, spatial-frequency, and orientation optimized by two-dimensional visual cortical filters. *J. Opt. Soc. Amer.*, 2(7):1160–1169, 1985.

[27] J. G. Daugman. Complete discrete 2-d Gabor transforms by neural networks for image analysis and compression. *IEEE Trans. Acoust., Speech, Signal Process.*, 36(7):1169–1179, 1988.

[28] R. L. De Valois, D. G. Albrecht, and L. G. Thorell. Spatial-frequency selectivity of cells in macaque visual cortex. *Vision Research*, 22:545–559, 1982.

[29] R. L. De Valois and K. K. De Valois. *Spatial Vision.* Oxford University Press, 1988.

[30] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, London, 1982.

[31] N. R. Draper and H. Smith. *Applied Regression Analysis.* John Wiley, New York, 2nd edition, 1981.

[32] R. C. Dubes. How many clusters are best? — an experiment. *Pattern Recognition*, 20(6):645–663, 1987.

[33] O. D. Faugeras. Texture analysis and classification using a human visual model. In *Proc. Int. Conf. on Pattern Recognition*, pages 549–552, Tokyo, Japan, 1978.

[34] L. A. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Trans. Pattern Anal. Machine Intell.*, 10(6):910–918, 1988.

[35] K. Fukunaga and P. M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE Trans. Comput.*, 24:750–753, 1975.

[36] D. Gabor. Theory of communication. *J. Inst. Elect. Engr.*, 93:429–457, 1946.

[37] G. Gagliardi, G. F. Hatch, and N. Sarkar. Machine vision applications in the food industry. In *Proc. SME Vision Conf.*, pages 6–40 through 6–54, Detroit, Michigan, 1985.

[38] A. P. Ginsburg. Visual information processing based on spatial filters constrained by biological data. Technical Report AMRL-TR-78-129, Air Force Aerospace Medical Research Laboratory, December 1978.

[39] N. H. Goddard, K. J. Lynne, T. Mintz, and L. Bukys. Rochester connectionist simulator. Technical Report TR 233 (revised), Dept. of Computer Science, University of Rochester, October 1989.

[40] S. Grossberg, editor. *Neural Networks and Natural Intelligence.* MIT Press, Cambridge, MA, 1988.

[41] J. F. Haddon and J. F. Boyce. Image segmentation by unifying region and boundary information. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(10):929–948, 1990.

[42] Y. Hara, H. Doi, K. Karasaki, and T. Iida. A system for PCB automated inspection using fluorescent light. *IEEE Trans. Pattern Anal. Machine Intell.*, 10(1):69–78, 1988.

[43] R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.

[44] P. E. Hart. The condensed nearest neighbor rule. *IEEE Trans. Inform. Theory*, 14:515–516, May 1968.

[45] S. Haykin. *Communication Systems.* John Wiley & Sons, 1978.

[46] R. Hoffman and A. K. Jain. Segmentation and classification of range images. *IEEE Trans. Pattern Anal. Machine Intell.*, 9(5):608–620, 1987.

[47] J. Y. Hsiao and A. A. Sawchuk. Unsupervised textured image segmentation using feature smoothing and probabilistic relaxation techniques. *Computer Vision, Graphics, Image Process.*, 48:1–21, 1989.

[48] A. K. Jain. Experiments in texture analysis using spatial filtering. In *Proc. IEEE Workshop on Languages for Automation*, pages 66–70, Spain, June 1985.

[49] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice-Hall, New Jersey, 1988.

[50] A. K. Jain, R. C. Dubes, and C.-C. Chen. Bootstrap techniques for error estimation. *IEEE Trans. Pattern Anal. Machine Intell.*, 9(5):628–633, 1987.

[51] A. K. Jain and S. G. Nadabar. Mrf model-based segmentation of range images. In *Proc. International Conference on Computer Vision*, pages 667–671, Osaka, Japan, December 1990.

[52] B. Julesz. Visual pattern discrimination. *IRE Trans. Inform. Theory*, 8(2):84–92, 1962.

[53] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, 1981.

[54] B. Julesz. Texton gradients: The texton theory revisited. *Biol. Cybern.*, 54:245–251, 1986.

[55] B. Julesz and J. R. Bergen. Textons, the fundamental elements in preattentive vision and perception of textures. *Bell Syst. Tech. J.*, 62(6):1619–1645, 1983.

[56] B. Julesz, E. N. Gilbert, L. A. Shepp, and H. L. Frisch. Inability of humans to discriminate between visual textures that agree in second-order statistics — revisited. *Perception*, 2:391–405, 1973.

[57] B. Julesz, E. N. Gilbert, and J. D. Victor. Visual discrimination of textures with identical third-order statistics. *Biol. Cybern.*, 31:137–140, 1978.

[58] R. L. Kashyap and A. Khotanzad. A model-based method for rotation invariant texture classification. *IEEE Trans. Pattern Anal. Machine Intell.*, 8(4):472–481, 1987.

[59] J. M. Keller and S. Chen. Texture description and segmentation through fractal geometry. *Computer Vision, Graphics, Image Process.*, 45:150–166, 1989.

[60] A. Khotanzad and J. Y. Chen. Unsupervised segmentation of textured images by edge detection in multidimensional features. *IEEE Trans. Pattern Anal. Machine Intell.*, 11(4):414–421, 1989.

[61] K. I. Laws. Textured image segmentation. Technical Report USCIPI-940, Image Process. Inst., University of Southern California, 1980.

[62] M. D. Levine. *Vision in Man and Machine.* McGraw-Hill, 1985.

[63] R. P. Lippmann. An introduction to computing with neural nets. *IEEE ASSP Magazine*, pages 4–22, April 1987.

[64] R. P. Lippmann. Pattern classification using neural networks. *IEEE Communications Magazine*, pages 47–64, November 1989.

[65] J. Malik and P. Perona. A computational model of texture segmentation. In *Proc. IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, pages 326–332, San Diego, CA, 1989.

[66] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *J. Opt. Soc. Amer. A*, 7(5):923–932, 1990.

[67] S. G. Mallat. Multifrequency channel decomposition of images and wavelet models. *IEEE Trans. Acoust., Speech, Signal Process.*, 37(12):2091–2110, 1989.

[68] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Pattern Anal. Machine Intell.*, 11(7):674–693, 1989.

[69] B. S. Manjunath, T. Simchony, and R. Chellappa. Stochastic and deterministic networks for texture segmentation. *IEEE Trans. Acoust., Speech, Signal Process.*, 38(6):1039–1049, 1990.

[70] S. Marcelja. Mathematical description of the responses of simple cortical cells. *J. Opt. Soc. Amer.*, 70:1297–1300, 1980.

[71] D. Marr. *Vision.* Freeman, San Francisco, 1982.

[72] D. L. Milgram. Region extraction using convergent evidence. *Computer Graphics, Image Process.*, 11:1–12, 1979.

[73] G. W. Milligan and M. C. Cooper. A study of standardization of variables in cluster analysis. *J. Classification*, 5:181–204, 1988.

[74] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry.* MIT Press, 1969.

[75] T. Pavlidis and Y. Liow. Integrating region growing and edge detection. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(3):225–233, 1990.

[76] A. P. Pentland. Fractal-based description of natural scenes. *IEEE Trans. Pattern Anal. Machine Intell.*, 6(6):661–674, 1984.

[77] A. Perry and D. G. Lowe. Segmentation of textured images. In *Proc. IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, pages 326–332, San Diego, CA, 1989.

[78] D. A. Pollen and S. F. Ronner. Visual cortical neurons as localized spatial frequency filters. *IEEE Trans. Syst., Man, Cybern.*, 13(5):907–916, 1983.

[79] M. Porat and Y. Y. Zeevi. The generalized Gabor scheme of image representation in biological and machine vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 10(4):452–468, 1988.

[80] T. R. Reed and H. Wechsler. Segmentation of textured images and Gestalt organization using spatial/spatial-frequency representations. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(1):1–12, 1990.

[81] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, volume 1, pages 318-362. MIT Press, Cambridge, MA, 1986.

[82] M. Sachs, J. Nachimas, and R. J. Spatial-frequency channels in human vision. *J. Opt. Soc. Amer.*, 61:1176–1186, 1971.

[83] D. J. Sakrison. On the role of the observer and a distortion measure in image transmission. *IEEE Trans. Communications*, 25(11):1251–1267, 1977.

[84] L. H. Siew, R. M. Hodgson, and E. J. Wood. Texture measures for carpet wear assessment. *IEEE Trans. Pattern Anal. Machine Intell.*, 10(1):92–105, 1988.

[85] J. Sklansky. Image segmentation and feature extraction. *IEEE Trans. Syst., Man, Cybern.*, 8(4):237–247, 1978.

[86] H. Tamura, S. Mori, and T. Yamawaki. Textural features corresponding to visual perception. *IEEE Trans. Syst., Man, Cybern.*, 8(6):460–473, 1978.

[87] T. N. Tan and A. G. Constantinides. Texture analysis based on a human visual model. In *Proc. IEEE Int. Conf. on Acoust., Speech, Signal Process.*, pages 2091–2110, Albuquerque, New Mexico, April 1990.

[88] W. S. Torgerson. *Theory and Methods of Scaling.* Wiley, New York, 1958.

[89] M. Tuceryan and A. K. Jain. Texture segmentation using voronoi polygons. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(2):211–216, 1990.

[90] M. R. Turner. Texture discrimination by Gabor functions. *Biol. Cybern.*, 55:71–82, 1986.

[91] L. Van Gool, P. Dewaele, and A. Oosterlinck. Texture analysis anno 1983. *Computer Vision, Graphics, Image Process.*, 29:336–357, 1985.

[92] H. Voorhees. Finding texture boundaries in images. Technical Report AI-TR-986, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1987.

[93] H. Voorhees and T. Poggio. Computing texture boundaries from images. *Nature*, 333(6171):364–367, 1988.

[94] S. R. Yhann and T. Y. Young. A multiresolution approach to texture segmentation using neural networks. In *Proc. Int. Conf. on Pattern Recognition*, volume 1, pages 513–517, Atlantic City, NJ, June 1990.

[95] R. A. Young. The Gaussian derivative theory of spatial vision: Analysis of cortical cell receptive field line-weighting profiles. Technical Report GMR-4920, General Motors Research Center, Warren, MI, 1985.