# Filtering for Texture Classification: A Comparative Study

Trygve Randen, *Member*, *IEEE*, and John Håkon Husøy, *Member*, *IEEE*

**Abstract**—In this paper, we review most major filtering approaches to texture feature extraction and perform a comparative study. Filtering approaches included are Laws masks, ring/wedge filters, dyadic Gabor filter banks, wavelet transforms, wavelet packets and wavelet frames, quadrature mirror filters, discrete cosine transform, eigenfilters, optimized Gabor filters, linear predictors, and optimized finite impulse response filters. The features are computed as the local energy of the filter responses. The effect of the filtering is highlighted, keeping the local energy function and the classification algorithm identical for most approaches. For reference, comparisons with two classical nonfiltering approaches, co-occurrence (statistical) and autoregressive (model based) features, are given. We present a ranking of the tested approaches based on extensive experiments.

**Index Terms**—Texture classification, image processing, filtering, survey.

## 1 Introduction

Classification and segmentation of texture content in digital images has received considerable attention during the past decades and numerous approaches have been presented [1], [2]. Tuceryan and Jain [2] identifies five major categories of features for texture identification; statistical, geometrical, structural, model-based, and signal processing features. It is our impression that the statistical, model-based, and signal processing techniques are the most commonly used. The focus of this paper will be on signal processing approaches.

A common denominator for most signal processing approaches is that the textured image is submitted to a linear transform, filter, or filter bank, followed by some energy measure. Due to the inherent similarities between these approaches, they will all be referred to as filtering approaches in this paper.

Although similar in concept, several quite different filtering schemes have been presented in the literature. One of the pioneering techniques was the approach by Laws [3], where a bank of band pass filters was applied. Subsequent works have focused on different filter bank families, different subband decompositions, and on optimization of the filters for texture feature separation. The large number of different test images applied, along with different system setups, have made comparison of the filtering approaches based on published results very difficult.

A few comparisons between texture feature extraction schemes have been presented. Weszka et al. [4] compared the Fourier power spectrum, second order gray level statistics, co-occurrence statistics, and gray level run length statistics features. They concluded that the co-occurrence features were the best of these features. The co-occurrence features were also the best features in a study by Conners and Harlow [5], when compared with run length difference, gray level difference density, and power spectrum.

Du Buf et al. [6], on the other hand, reported that several features had roughly the same performance. They evaluated co-occurrence features, fractal dimension, transform and filter bank features, dominant local frequency and orientation features, number of gray level extrema per unit area, and curvilinear integration features. Filtering features have been compared to the co-occurrence features in some more studies, with different conclusions. Strand and Taxt [7] concluded that the co-occurrence features were performing best, while Laws [3], Pietikäinen et al. [8], and Clausi and Jernigan [9] had the opposite conclusion. Different setups, different test images, and different filtering methods may be the reasons for the contradicting results. Ojala et al. [10] compared gray-level difference, Laws filter, covariance, local binary patterns, and complementary feature pairs. They concluded that the gray level difference features were performing best and noted a particularly poor performance of the Laws features.

We note that no extensive evaluation of the various filtering approaches has been performed. The aim of this paper is to provide such a comparative study. The system setup will be as illustrated in Fig. 1. In order to make the results comparable, the focus will be on the filtering part of this system, keeping the other components as similar as possible. For reference, we also compare the filtering approaches with two classes of popular, nonfiltering texture features, model-based and statistical features. In our experiments, we compare the classification results for some images with two to 16 textures with borders ranging from simple to difficult. Special emphasis is put on making the results realistic, thus the design and test data sets are disjoint. Interestingly enough, this is far

- *T. Randen is with Schlumberger Geco-Prakla, P.O. Box 330, 4001 Stavanger, Norway.*
  *E-mail: t.randen@ieee.org.*
- *J.H. Husøy is with Stavanger College, Stavanger, Norway.*
  *E-mail: John.H.Husoy@tn.his.no.*

*Manuscript received 13 Feb. 1998; revised 11 Dec. 1998. Recommended for acceptance by K. Bowyer.*
*For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 108099.*

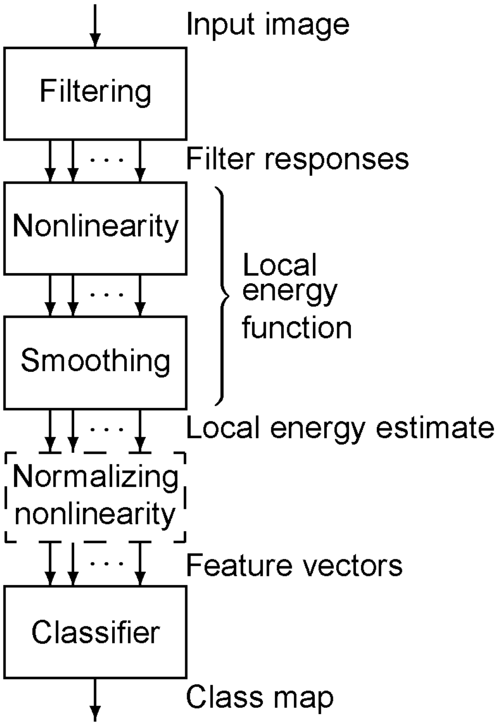


Fig. 1. Experimental setup.

from always the case in experiments presented in the literature on texture classification.

This paper is organized as follows: In Section 2, we give an introduction to how texture features are extracted by filtering, along with a review of the different filtering approaches. In Section 3, we present the experimental results for the test of the various filtering approaches.

## 2 Feature Extraction Approaches and System Setup

Before we proceed with the description of the different filtering approaches, we will give a description of the elements of the system in Fig. 1. Consider the simple synthetic textured image in Fig. 2a. This image consists of two textures generated by sinusoids. The left half of the image has one low-frequency sinusoid and the right half has a high-frequency sinusoid superimposed on the low frequency one.

For illustrational purposes, consider a horizontal line through this image, Fig. 2b. The first operation of the system illustrated in Fig. 1 is filtering. Assume that the image is filtered with a filter stopping the low frequency sinusoid and passing the high frequency sinusoid. A line in the resulting image is illustrated in Fig. 2c. In this case, we see that the filter response for the left texture has low energy, and the right texture high. However, we still can not classify the image by its pixel values alone without significant classification errors.

Next, a local energy function is applied, consisting of a nonlinearity (Fig. 2d), basically rectifying[1] the filter response and smoothing (Fig. 2e). The resulting feature image is given in Fig. 2f, and this feature image can be classified with success. The optional second nonlinearity box in Fig. 1 is not illustrated in this experiment and will be treated later.

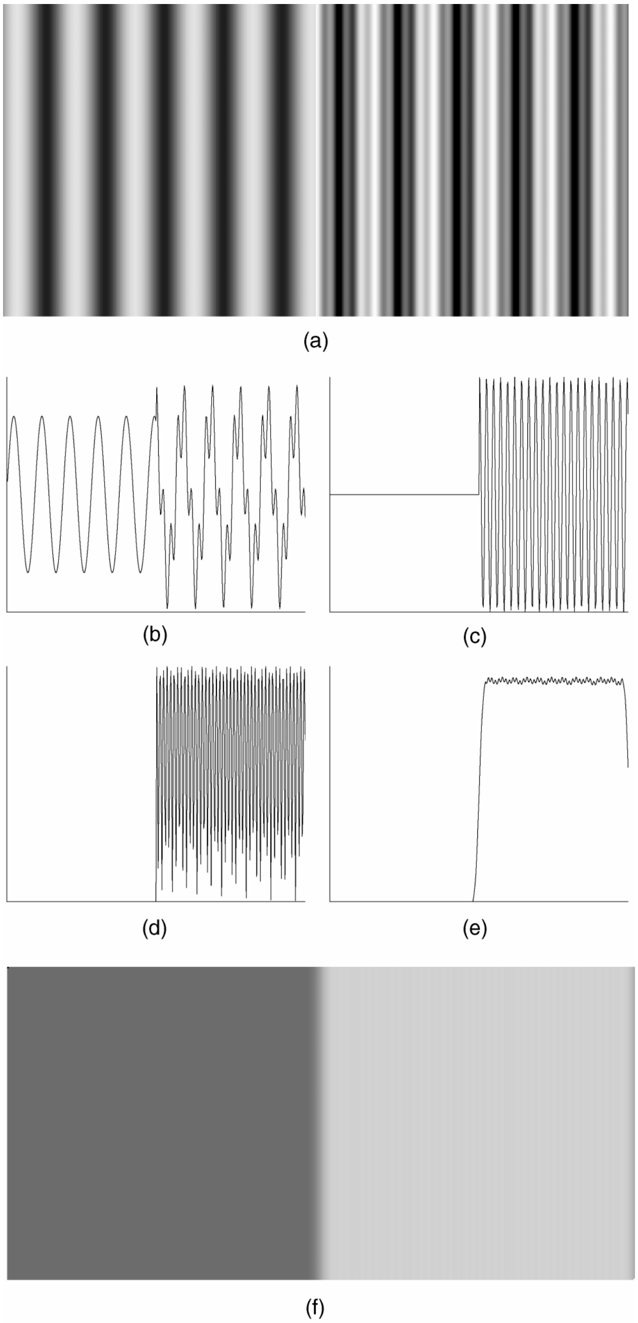


Fig. 2. Illustration of a typical texture feature extraction process with filters. (a) Two synthetic textures, (b) a horizontal scan line through the image, (c) filtered, (d) nonlinear transform, (e) smoothing, and (f) the resulting feature image.

In the literature, different choices for all the elements of Fig. 1 have been reported. In this section, we will review some of the approaches found in the literature. We do, however, believe that the most important element is the filter or filter bank. The scope of this paper is to examine the performance of different choices for the filtering step. In order to isolate this problem to keep the results comparable, we will keep the other elements of the setup as similar as possible.

[1] Rectification here is understood as the operation of transforming negative amplitudes to the corresponding positive amplitudes.

## 2.1 Local Energy Function and Post Processing

The objective of the local energy function (see Figs. 1 and 2) is to estimate the energy in the filter output in a local region. Unfortunately, accurate edge preservation and accurate energy estimation are conflicting goals. For edge localization, high spatial resolution is desired, while for energy estimation, high spatial frequency resolution is desired. These desires need to be balanced in the smoothing filter.

Commonly applied smoothing filters in the local energy function are rectangular [3], [7], [11], [12], [13] and Gaussian [9], [14], [15], [16], [17], [18], [19]. Experience has taught us that the Gaussian filter is the far better choice and will consequently be used in our experiments.

How can the size of the smoothing filter be appropriately determined? If we want to estimate the local energy of low spatial frequencies, the smoothing filter will have to be large, while we can allow smaller smoothing filters for higher frequencies. Hence, for a band-limited filter output, we may set the smoothing filter size as a function of the band center frequency. In our experiments, we compute the radial spatial center frequency, $f_0$, for all band pass filters, and apply a separable Gaussian low-pass filter with the unit pulse response,

$$h_G(n) = \frac{1}{\sqrt{2\pi}\sigma_s} e^{-\frac{1}{2}\frac{n^2}{\sigma_s^2}} \tag{1}$$

in each dimension, where

$$\sigma_s = \frac{1}{2\sqrt{(2)}f_0}.$$

This smoothing function was initially suggested by Jain and Farrokhnia [20]. For subsampled filter responses, a modification of this scheme described by Randen and Husøy [17] is used.

Another element of the local energy function is the nonlinearity. Commonly applied nonlinearities are the magnitude $|\cdot|$ [3], [8], [9], [12], [14], [15], [16], [21], [22], the squaring $(\cdot)^2$ [4], [11], [13], [18], [23], [24], [25], [26], and the rectified sigmoid, $|\tanh(\alpha\cdot)|$ [7], [17], [19], [20]. The magnitude and squaring nonlinearities are parameter free, whereas the rectified sigmoid nonlinearity requires tuning of the saturation parameter, $\alpha$. This may be an advantage if it is easy to tune this parameter, or a disadvantage otherwise. An appropriate saturation parameter is dependent on the dynamic range of the input image.

The second nonlinearity illustrated in Fig. 1 is not commonly used. Unser [27] proposed and tested several combinations of the first and second nonlinearity for texture segmentation. He concluded that squaring in combination with a logarithmic normalizing nonlinearity was the best combination. This combination will consequently be used in most experiments in this paper. Unser did not test the rectified sigmoid nonlinearity. However, due to the issue of appropriate saturation parameter determination, this nonlinearity will primarily not be used here.

## 2.2 Classification

The output from the local energy function is a set of images, one image per filter. These images are the feature images that will form the basis for the classification. They are used to form feature vectors, where each feature image corresponds to one element in the feature vectors. Each feature vector corresponds to one or a few image pixels. Classification is the task of assigning class labels to these feature vectors.

### *2.2.1 Classification Scheme*

Numerous classification approaches are possible and are described in several text books, e.g., [28], [29], [30]. The choice of classifier will not be examined in detail, but classifiers will be used for illustrating the usefulness of the proposed feature extraction schemes. Examples of classification schemes reported in the literature are Bayes classifiers assuming multivariate Gaussian feature distributions [11], [15], [16], [23], Fisher transformation [4], [26], nearest neighbor classifiers [7], [10], [22], classification trees [26], feed-forward neural networks [19], and the Learning Vector Quantization scheme [17]. In some cases, even thresholding [8], [18], [25], [31] and simple extremum picking [14], [32], [33] are applicable as classification schemes. Yet, other researchers [6], [9], [12], [20], [21], [34] report their results using clustering schemes, i.e., unsupervised schemes. It is beyond the scope of this article to test all these classification schemes. We have chosen to use the "Type One Learning Vector Quantizing" (LVQ) supervised classifier of Kohonen [35] for most experiments.

Training and using complex statistical or neural network classifiers like the LVQ is clearly both time consuming and puts heavy requirements on training data quality and size. As will become clear some of the feature extraction schemes are specially designed to avoid this complexity and use maximum picking or thresholding instead. Since these schemes probably will never be used together with a complex statistical or neural network classifier, it would not be appropriate to test them with such classifiers.

### *2.2.2 Training Data*

In order to have a reliable test of the classification, it is necessary to have separate test and training feature vectors. Despite this, it is quite common in texture classification to pick the training feature vectors as a subset of the test vectors [4], [7], [17], [19], [20] or as a not completely disjoint set [10], [23], [24], [26]. This practice has major implications on the performance of the classification system and yields results not attainable in a realistic environment. Farrokhnia [19] suggests using 6 percent of the feature vectors for classifier training in the supervised experiments. To illustrate the effect of this choice, experiments with the texture image of Fig. 11a were conducted. In these experiments, the classifiers were trained using two different approaches:

1) Using 6 percent of the test feature vectors, i.e., about 800 vectors per class, for training the LVQ.
2) Picking training and test data from nonintersecting regions of the source texture images, without using any edge specific information. About 800 feature vectors were used for training also in this case.

Otherwise, the experiments were identical. A total of 46 feature extraction techniques were applied (see Tables 3 and 4 for details). The average results were 4.9 percent classification error when using a subset of the test features as training data and 17.6 percent error when using separate training data. As expected, we see a significant increase in

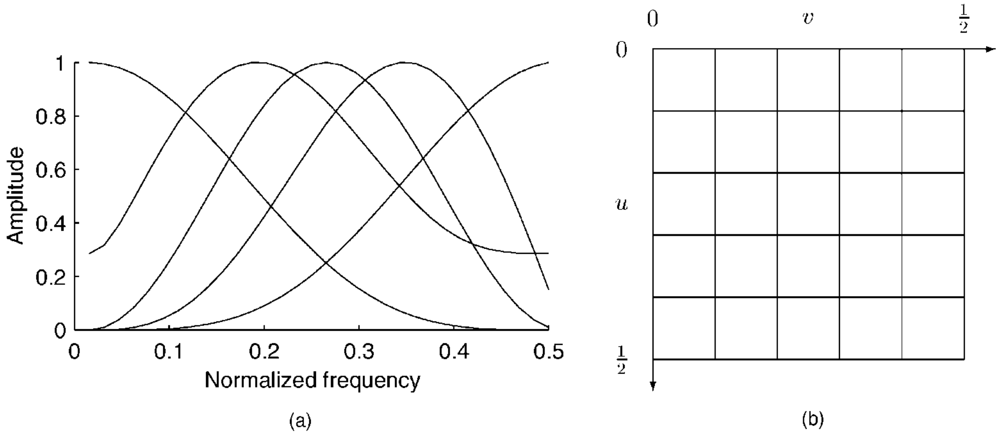


Fig. 3. (a) Subband split of the one-dimensional equivalent separable Laws [3] filter masks (normalized). (b) The resulting two-dimensional frequency band split. The axis labels are vertical and horizontal normalized spatial frequencies.

classification error, primarily due to increased edge inaccuracy, when using separate training data. In only 12 of the 53 cases the classification error was less than doubled when moving to a separate training data set.

Overoptimistic results, however, is not the only problem with a nonseparate training set. The relative performance degradation was very dependent on the method. For example, with co-occurrence features [36] the error rate was 7.6 percent with a nonseparate training set and 9.7 percent with separate. The results with a dyadic Gabor filter bank [20] were 3.8 percent and 14.5 percent, respectively. Hence, if we were comparing these methods based on nonseparate training data, the conclusion on their relative performances would be *the opposite* of the conclusion when comparing them with the more realistic (separate) training data case.

## 2.3 Heuristically Designed Filter Banks

The basic assumption for most filtering approaches is that the energy distribution in the frequency domain identifies a texture. Hence, if the frequency spectrum of the textured image is decomposed into a sufficient number of subbands, the spectral energy signatures of different textures are different. Utilizing this, several filter bank approaches and related schemes have been proposed. Some major approaches are discussed here. Parameter choices for the approaches with respect to the experiments are also discussed.

### 2.3.1 Laws Filter Masks

One of the first approaches to filtering for texture identification was presented in the work by Laws [3]. Laws suggested using a bank of separable filters, five in each dimension, i.e., a total of 25 filters. The filter masks suggested were $h_1$ = [1, 4, 6, 4, 1], $h_2$ = [–1, –2, 0, 4, 1], $h_3$ = [–1, 0, 2, 0, –1], $h_4$ = [–1, 2, 0, –2, 1], and $h_5$ = [1, –4, 6, –4, 1]. The resulting 25 subbands are illustrated in Fig. 3b, with the one-dimensional equivalent in Fig. 3a.

### 2.3.2 Ring and Wedge Filters

Assuming that texture is discriminated by spatial frequency and orientation, Coggins and Jain [12] suggested using seven dyadically spaced ring filters and four wedge-shaped orientation filters for feature extraction. The filters are designed in the two-dimensional spatial frequency domain, giving the amplitude responses of Fig. 4. The rings and wedges have Gaussian cross sections.

### 2.3.3 Dyadic Gabor Filter Bank

Jain and Farrokhnia [20] suggested a bank of Gabor filters, i.e., Gaussian shaped band-pass filters, with dyadic coverage of the radial spatial frequency range and multiple orientations. This choice was justified by the relation to models for the early vision of mammals as well as the filters' joint optimum resolution in time and frequency.

The basic even-symmetric Gabor filter oriented at 0° is a band-pass filter with unit pulse response

$$h(k,l) = e^{-\frac{1}{2}\left(\frac{k^2}{\sigma_x^2}+\frac{l^2}{\sigma_y^2}\right)}\cos\left(2\pi f_0 k\right), \tag{2}$$

where $f_0$ is the radial center frequency. Other orientations are obtained by rotating the reference coordinate system, $(k, l)$. This filter has an infinite unit pulse response, but in practical experiments it is approximated by a finite length filter.

Five radial frequencies are suggested [20] (for images of size 256 × 256) and four orientations. In the tests presented here, the same number of filters are used, irrespective of the size of the input image. The discrete radial center frequencies

$$\frac{\sqrt{2}}{2^6}, \frac{\sqrt{2}}{2^5}, \frac{\sqrt{2}}{2^4}, \frac{\sqrt{2}}{2^3} \text{ and } \frac{\sqrt{2}}{2^2} \tag{3}$$

and orientations 0°, 45°, 90°, and 135° are used (see Fig. 5).

### 2.3.4 Wavelet Transform, Packets, and Frames

A transform like the discrete wavelet transform corresponds to a critically sampled filter bank with particular filter parameters and subband decompositions [37]. Wavelet transform approaches are consequently filter bank approaches. The application of the discrete wavelet transform, and variants thereof for texture identification has received considerable attention in the literature. The use for texture analysis was pioneered by Mallat [38], who applied a "standard" wavelet transform for feature extraction, i.e., critically decimated with dyadic subband structure. The work by Chang and Kuo [24], however, indicates that *texture* features are most prevalent in intermediate frequency

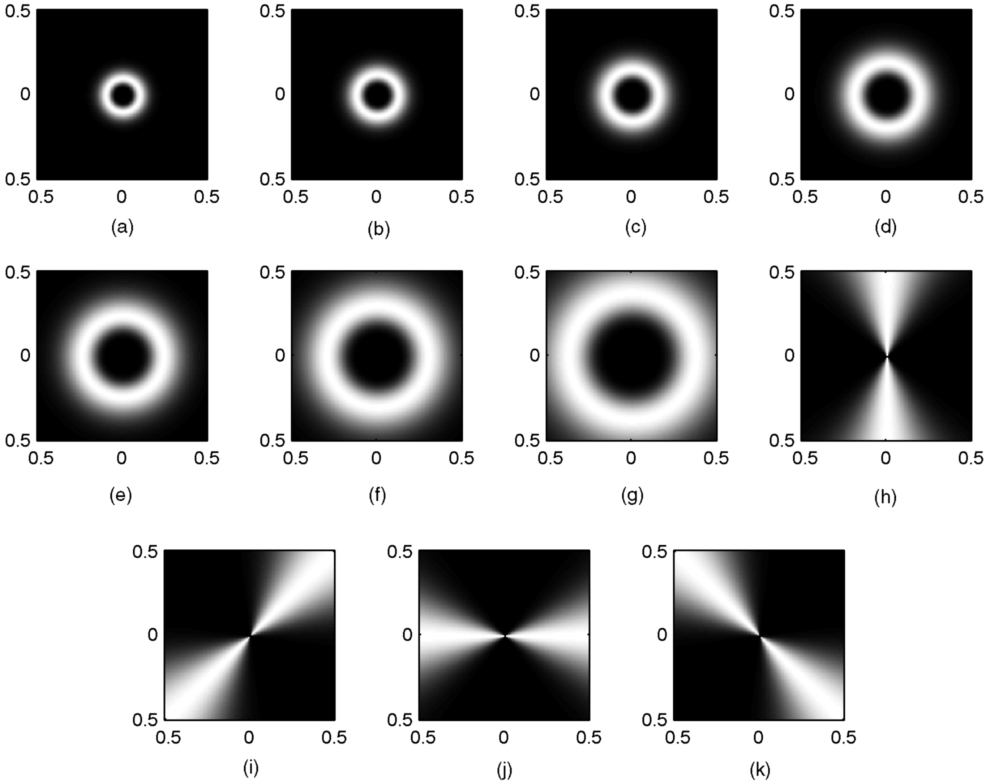


Fig. 4. Amplitude responses of the ring and wedge filters [12]. The axis labels are normalized spatial frequencies.

bands, thus that the octave band decomposition is not optimal. The trend probably therefore seems to be a concentration on the wavelet *packet* transform [21], [24], [26], which basically is the wavelet transform with subband decompositions not restricted to be dyadic.

The discrete wavelet transform and the discrete wavelet packet transform are critically sampled multi rate filter banks. However, critically sampled filter banks typically imply inaccurate texture edge localization [17]. The use of over-complete wavelet representations, i.e., wavelet frames [11], is a remedy for alleviating this problem. Improved results with over-complete representations have been reported [11], [17], see also Section 3.

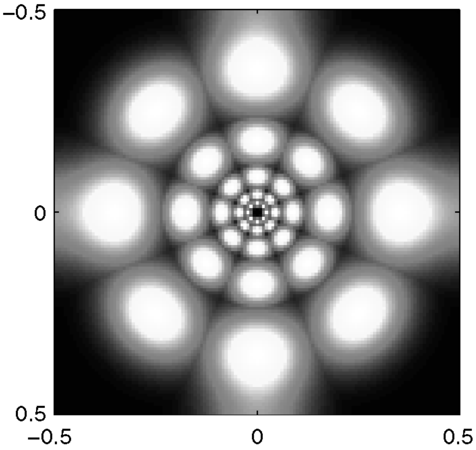


Fig. 5. The frequency response of the dyadic bank of Gabor filters. The maximum amplitude response over all filters is plotted. Each filter is represented by one center-symmetric pair of lobes in the illustration. The axes are in normalized spatial frequencies.

Evaluation of absolutely all approaches to texture analysis using wavelet representations is beyond the scope of the experiments. The attention will be restricted to the Daubechies family of wavelets [39]. The subband decompositions illustrated in Fig. 6 will be evaluated. Using both full rate and critical subsampling and these decompositions, both wavelet transform, wavelet packet, and wavelet frame techniques are covered. The responses of the applied filters classes are illustrated in Fig. 7.

### *2.3.5 Discrete Cosine Transform (DCT)*

The discrete cosine transform is popular in image coding due to good performance and fast implementation [40]. It is, for instance, the backbone in the JPEG compression standard. Ng et al. [41] suggest using a 3 × 3 DCT for texture feature extraction. They furthermore suggest excluding the low-frequency component of the DCT, thus yielding eight features.

Image transforms are equivalent to critically sampled filter banks. The above approach is tested in a filter bank implementation, without critical sampling. The filter bank is separable, determined by the one-dimensional filter masks $h_1 = [1, 1, 1]$, $h_2 = [1, 0, -1]$, $h_3 = [1, -2, 1]$. The amplitude responses are illustrated in Fig. 8.

### *2.3.6 Quadrature Mirror Filters (QMF)*

Another class of filters, quadrature mirror filters (QMF), were used for texture analysis by Randen and Husøy [17]. This is, like the class of wavelet filters, a very broad class of filters, incorporating both infinite impulse response (IIR) and finite

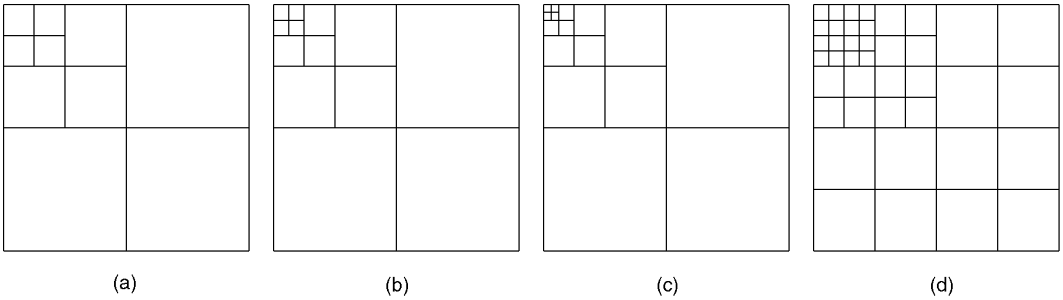


Fig. 6. Subband decompositions evaluated in this paper. Decompositions (a-c) are dyadic (octave band), while (d) is not. The axes are the same as in Fig. 3b.

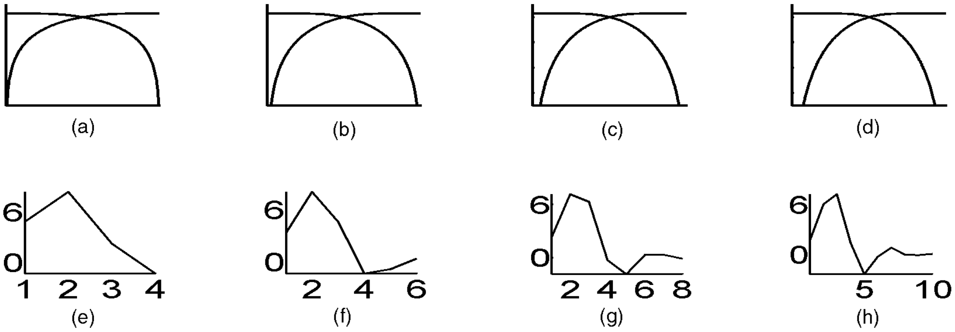


Fig. 7. The two-channel filter bank dB amplitude responses for the wavelet filters (a) “Daubechies 4,” (b) “Daubechies 6,” (c) “Daubechies 8,” (d) “Daubechies 10,” and their respective unit pulse responses (e)-(h).

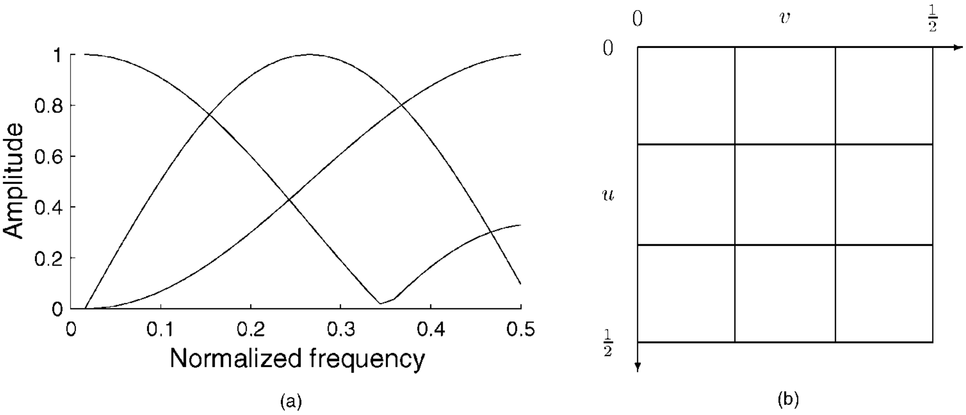


Fig. 8. (a) One-dimensional equivalent of the amplitude response of the filter bank corresponding to a separable 3 × 3 Discrete Cosine Transform (normalized). (b) The resulting two-dimensional frequency band split. The axis labels are vertical and horizontal normalized spatial frequencies.

impulse response (FIR) filters. In this study, we will analyze a few filters applied to the subband decompositions in Fig. 6. The filters to be analyzed are the 8, 16, and 32-tap FIR filters “f8a,” “f16b,” and “f32d” [42] and the IIR filters “F_2_1_09” and “F_2_1_smpl” [43]. The two-channel amplitude responses (with a dB axis) for these filters are given in Fig. 9. The two IIR filters require a low number of multiplications and additions [43], yielding low computational complexity in the feature extraction. For “F_2_1_smpl” all multiplications are by powers of two and can be implemented by bit-wise shifts. Like with the wavelet filters, these filters will be tested both at critical rate (subsampled) and full rate (not subsampled).

### 2.3.7 Tree-Structured Gabor Filter Bank

Motivated by the success of the dyadic Gabor filter bank [20] and the arbitrary QMF decompositions [17], we suggested using nondyadic Gabor filter banks [17]. In this approach, the filter bank is decomposed with the decompositions of Fig. 6. The subband filter of Section 2.3.3 is used with only minor modifications [17].

## 2.4 Optimized Filters and Filter Banks

The heuristically designed filter banks[2] have been reported to yield successful results in numerous cases. However, most of the heuristically designed filter banks imply large numbers of features. Consequently, the computational complexities are *typically* large in both feature extraction and classification. It may, therefore, be desirable to attempt to optimize the filtering operation. Optimizing filters or filter banks may yield low feature vector dimensionality,

[2] By a heuristically designed filter bank is meant a filter bank that is not optimized by some explicit criterion related to the texture classification.

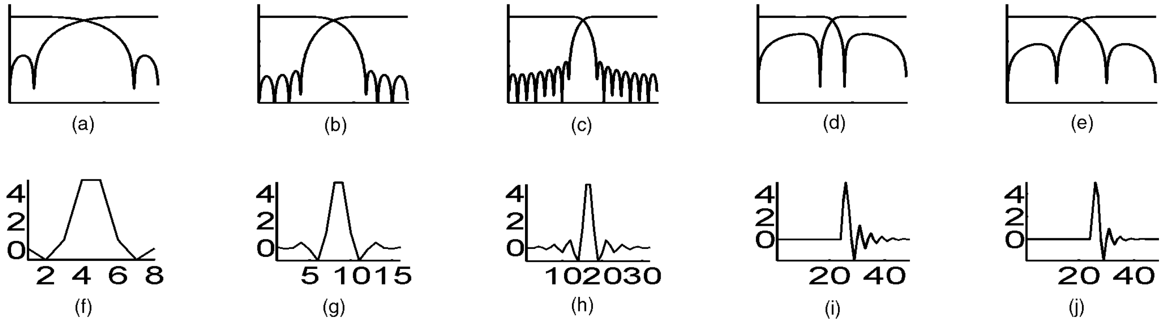


Fig. 9. The two-channel dB amplitude responses for the quadrature mirror filters (QMF) (a) "f8a," (b) "f16b," (c) "f32d," (d) "F_2_1_09," and (e) "F_2_1_smpl" and their respective low-pass prototype unit pulse responses (f)-(j).

maximized feature separation, and in some cases more simple classifiers.

As will be evident from the presentation, some of the optimization approaches are inherently restricted to two-texture problems, while others are applicable also to multi-texture problems. Furthermore, some of the approaches yield one filter, while others yield banks of filters. However, common for all approaches is that the filters are optimized with respect to some criterion related to the texture classification task.

### 2.4.1 Eigenfilter

An early attempt at optimizing the filter function for texture feature extraction was presented in 1983 by Ade [13]. Ade suggests using eigenfilters derived from the autocorrelation functions of the textures. From each texture, the 9 × 9 matrix

$$\mathbf{R}_{xx} = \begin{bmatrix} E\left[x_{m,n}x_{m,n}\right] & \cdots & E\left[x_{m,n}x_{m,n+2}\right] & \cdots & E\left[x_{m,n}x_{m+2,n+2}\right] \\ \vdots & & \vdots & & \vdots \\ E\left[x_{m,n+2}x_{m,n}\right] & \cdots & E\left[x_{m,n+2}x_{m,n+2}\right] & \cdots & E\left[x_{m,n+2}x_{m+2,n+2}\right] \\ \vdots & & \vdots & & \vdots \\ E\left[x_{m+2,n+2}x_{m,n}\right] & \cdots & E\left[x_{m+2,n+2}x_{m,n+2}\right] & \cdots & E\left[x_{m+2,n+2}x_{m+2,n+2}\right] \end{bmatrix},$$

where $x_{m,n}$ is the image and $E[\cdot]$ is the expectation operator is estimated and the eigenvectors and eigenvalues are computed. Each 9 × 1 eigenvector corresponds thus to a 3 × 3 filter mask. Then, the image is filtered with the 3 × 3 filters corresponding to the principal eigenvectors. One set of filters is designed for each texture and all filters are applied to the composite image.

In the original approach, the filters corresponding to the eigenvalues summing up to at least 99 percent of the total eigenvalue sum were selected. This leads to a significant number of filters, typically in the range 5 to 9 per texture. Since the test images in Fig. 11 have up to 16 textures, a considerable number of filters would be constructed. This implies very significant computational complexities for the feature extraction and classification systems. Therefore, the total number of filters was restricted to maximally $50/N_T$ per texture, where $N_T$ is the number of textures. Hence, the maximum number of filters for any image is 50.

The filters designed by this technique are not band-pass filters. Consequently, using a smoothing operator dependent on the center frequency, as suggested in Section 2.1, is not appropriate. Hence, a fixed Gaussian smoothing filter with $\sigma_s = 8$ is used. This selection of filter size was based on numerous experiments.

It is worth noting that this approach is optimized with respect to image *representation*—it is closely related to the Karhunen-Loeve transform [40]. However, optimized representation does not imply optimized *discrimination*.

### 2.4.2 Prediction Error Filter

Yet another filtering scheme optimized with respect to the *representation* is the linear prediction error filtering scheme of Randen and Husøy [32], [33]. The filters are designed to give the least squared prediction error for each texture, thus the number of filters is the same as the number of textures. A linear predictor in image processing is an equation of the form

$$\hat{x}(m,n) = -\sum_{(k,l)\in\mathcal{N}} \theta(k,l)x(m-k,n-l), \qquad (4)$$

predicting a pixel value on the basis of neighboring pixels in the neighborhood $\mathcal{N}$ using the coefficients $\theta(k, l)$. The filter is optimized by minimizing $\sum_{m,n}\left(\hat{x}(m,n)-x(m,n)\right)^2$. The least squares error (LSE) predictor for a texture will be the predictor yielding maximum similarity between the texture and the predictor output, in the squared error sense.

Generally, the LSE predictors for different textures are different. Hence, for any texture the predictor yielding minimum mean error energy will be the predictor designed with respect to that texture. This may be used for classification, since the predictor yielding minimum local prediction error energy most likely corresponds to the underlying texture. In the experiments reported here, the smoothing filter is a Gaussian low pass filter with bandwidth given by $\sigma_s = 8$, just like with the eigenfilters. Classification is obtained by assigning each pixel to the class corresponding to the minimum prediction error. In the experiments herein, the region of support of the predictor is circular with radius four pixels.

Like the eigenfilter approach, this approach is only optimized with respect to image *representation*.

### 2.4.3 Optimized Representation Gabor Filter Bank

In order to tune the filters to the characteristics of the underlying textures, Bovik et al. [14] suggested using narrow-band Gabor filters. The filters' central frequencies are tuned to the spectral peaks of the textures. That is, for each texture the central frequencies of the corresponding Gabor filters are selected as the frequencies corresponding to the principal spectral peaks of the texture. Bovik et al. suggest a manual procedure for determining the spectral peaks. However, this procedure should easily be extendable to be fully automatic [14]. The manual procedure was implemented, selecting two filters per texture. The smoothing filter was tuned with respect to the center frequency of the selected filter.

We note that also this approach is optimized with respect to image *representation*, thus giving no guarantee for good feature *separation*.

### 2.4.4 Optimized Two-Class Gabor Filter

A Gabor filter design scheme yielding filters optimized with respect to feature *separation* has been suggested by Dunn and Higgins [44]. The scheme is designed to classify problems involving two textures. The Gabor filter center frequency giving features yielding minimum modeled classification error is designed. The optimal center frequency is determined by evaluating a large range of center frequencies (using the Fourier transform) and selecting the best candidate. The user is required to select the bandwidth, $\sigma$, of the filter, see (2). Gabor filters with $\sigma$-values 2, 4, 8, and 16 were evaluated. These $\sigma$-values are suggested by Weldon et al. [15]. Furthermore, as suggested by Weldon et al., smoothing Gaussian filters with spatial widths given by $\sigma_s = 2\sigma$ (see Section 2.1) were applied.

Inherently, this filter design approach is based on classification of the feature image by a threshold classifier. Such a simple classifier is less complex to design and use than classifiers like the LVQ. Furthermore, a large parameter size typically requires more training data vectors. Hence since a threshold classifier has only one parameter, the needed amount of training data is relatively low.

### 2.4.5 Optimized Multi-Class Gabor Filter Bank

The previous approach to optimized filter design was limited to two-texture problems with one filter. Furthermore, the filter bandwidth had to be determined heuristically. These issues are addressed by Weldon et al. [15], [16]. In their approach, a bank of Gabor filters is designed for problems involving an arbitrary number of textures. The user is only required to select the number of filters to be used. The approach is based on the feature extraction model developed by Dunn and Higgins [44]. However, a modified criterion function, incorporating the effect of the filter size on the edge accuracy, is used. This approach furthermore allows more than two textures and more than one filter.

The filter size is determined by evaluating a number of filter sizes and selecting the best candidate. Different filter sizes are allowed for the different filters in the filter bank. The $\sigma$ and $\sigma_s$ values, (2) and (1), suggested by Weldon et al. [15] were also used in these experiments, that is, $\sigma = 2, 4, 8, 16$, and $\sigma_s = 2\sigma$.

### 2.4.6 Optimized Two-Texture FIR Filters

The Gabor filter is a band pass filter with a Gaussian shaped pass band. The only free parameters of such a filter are the radial center frequency, the orientation, and the bandwidths. By allowing more free parameters, it should be possible to adapt the filter better to the underlying textures. Mahalanobis and Singh [31] suggest a two-texture design approach yielding FIR filters with maximum ratio between the extracted mean feature values. That is, filters maximizing the criterion

$$J_{MS} = \frac{\mu_{v_1}}{\mu_{v_2}},$$

are designed, where $\mu_{v_i}$ is the mean feature value for texture *i*.

Randen and Husøy [18] show that this criterion is insufficient for some texture pairs, and develop an optimization scheme allowing other criterion functions. Optimizations with respect to the criteria

$$J_U = \frac{\left(\mu_{v_1} - \mu_{v_2}\right)^2}{\mu_{v_1}\mu_{v_2}}$$

and

$$J_F = \frac{\left(\mu_{v_1} - \mu_{v_2}\right)^2}{\sigma_{v_1}^2 + \sigma_{v_2}^2},$$

where $\sigma_{v_i}^2$ is the feature variance, are shown to yield better results. The criterion $J_U$ was originally suggested by Unser [23] for designing optimized texture transforms and $J_F$ was originally suggested by Fisher [29] for linear classifier design.

For $J_U$, a closed form solution was presented, while for $J_F$ an approximate closed form solution and a gradient search solution are presented. Only the closed form solutions will be evaluated here.

Inherently, all these filters assume a threshold classifier, as was also the case with the optimized Gabor filters. Like the eigenfilters, filters designed by these approaches are in general not band pass filters, thus we use a Gaussian smoothing filter with $\sigma_s = 8$.

The size of the filter mask is also of importance for the results. A symmetric region of support is intuitively preferential for accurate edge localization. In order to have a symmetric region of support, an odd sized mask is required. Preliminary experiments indicate that mask sizes 5 × 5 and up are adequate for the given resolution. A few texture pairs have required larger than 5 × 5 mask sizes. In the experiments, filters with mask sizes 7 × 7 are optimized.

### 2.4.7 Optimized FIR Filter Bank

The FIR two-texture optimization with respect to the criterion functions $J_{MS}$ and $J_U$ have been extended to multiple-texture multiple-filter solutions [18]. The solution is based on one filter separating one group of textures from another, and then subsequent filters separating the textures within a group, hierarchically.

Note that this technique is also designed with thresholding as the inherent classifier, making the classifier design and application computationally simple. The same smoothing is used with these filters as with the two-texture filters.

#### 2.4.8 Back Propagation Designed Mask

Neural networks have proved to be of practical utility in numerous applications and have also been applied to texture feature extraction. Jain and Karu [45] suggest extracting features and classifying using a feed forward neural network trained by the back-propagation rule. In this approach, the input nodes cover a neighborhood of image pixels. The inputs are weighted and summed in the nodes in the network, thus parts of the network may be formulated as a filter or filter bank. Furthermore, the application of the nonlinearities in the network resembles the nonlinearity in the local energy function. Hence, this approach is very similar to filtering approaches. The scheme may be modified to utilize the effect of the smoothing operation in the local energy function [45]. However, this generalization is not used in the experiments due to implementational and complexity issues.

A three-layer neural network is used and the neural network parameters (numbers of nodes in the layers, etc.) used are the same as the ones used in [45]. Since this approach has filtering, nonlinearity, smoothing, and classification all in one, the standard setup (Fig. 1) is not used for these operations. Furthermore a 5 × 5 median filter is applied as proposed [45] to the resulting class map to reduce speckle-like classification errors.

### 2.5 Classical Nonfiltering Approaches

The focus of this paper is on filtering and local energy estimation for texture recognition. An extensive coverage of non-filtering approaches will not be given. However, due to the fundamental importance and wide range of applications of some nonfiltering approaches, a brief review is necessary. Furthermore, to provide a reference framework with techniques familiar to readers who are not familiar with filtering, comparisons with two classical nonfiltering approaches are provided. Results with co-occurrence (statistical) and autoregressive (model-based) features are presented in Section 3.

#### 2.5.1 Statistical Features

In statistical approaches, the textures are described by statistical measures. One commonly applied and referenced method is the co-occurrence method, introduced by Haralick [36]. In the co-occurrence method, the relative frequencies of gray level pairs of pixels at certain relative displacements are computed and stored in a matrix, the *co-occurrence matrix* **P**. For $G$ gray levels in the image, **P** will be of size $G \times G$. If $G$ is large, the number of pixel pairs contributing to each element, $p_{ij}$, in **P** will be low, and the statistical significance poor. On the other hand, if the number of gray levels is low, much of the texture information may be lost in the image quantization. Ohanian and Dubes [22] reported that $G = 8$ was an appropriate choice for subimages of size 32 × 32. These parameters have been chosen for the experiments in this paper. Since the task is supervised segmentation, overlapping subimages will be used. As suggested by other researchers [7], [22], the nearest neighbor pairs at orientations 0°, 45°, 90°, and 135° will be used in the experiments.

Haralick [36] suggests 14 features describing the two-dimensional probability density function $p_{ij}$. Four features that are commonly used [7], [22], [36] have been selected. These are the Angular Second Moment (ASM), Contrast (Con), Correlation (Cor), and Entropy (Ent). They are given by

$$\mathrm{ASM} = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} p_{ij}^2 \tag{5}$$

$$\mathrm{Con} = \sum_{n=0}^{G-1} N^2 \left\{ \sum_{|i-j|=n} p_{ij} \right\} \tag{6}$$

$$\mathrm{Cor} = \frac{1}{\sigma_x \sigma_y} \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} ij p_{ij} - \mu_x \mu_y \tag{7}$$

$$\mathrm{Ent} = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} p_{ij} \log p_{ij}, \tag{8}$$

where $\mu_x$, $\mu_y$, $\sigma_x$, and $\sigma_y$ are the means and the standard deviations corresponding to the distributions

$$p_i^{(x)} = \sum_{j=0}^{G-1} p_{ij} \tag{9}$$

$$p_j^{(y)} = \sum_{i=0}^{G-1} p_{ij}. \tag{10}$$

We extract each of these four features at each of the four different orientations, thus the feature vectors are 16-dimensional.

#### 2.5.2 Model-Based Features

Another major class of texture features is the model-based features. With model-based features, some image model is assumed, its parameters estimated for a subimage, and the model parameters, or attributes derived from them, are used as features.

As an example of this class of features, the multiresolution autoregressive (AR) features introduced by Mao and Jain [34] are used. The autoregressive model for an image $x(m, n)$ can be expressed as

$$x(m,n) = \sum_{(k,l)\in\mathcal{N}} \theta(k,l)\, x(m-k, n-l) + \sigma_\varepsilon \varepsilon(m,n), \tag{11}$$

where $\mathcal{N}$ is the model neighborhood, $\theta(k, l)$ are the model parameters, and $\sigma_\varepsilon \varepsilon(m, n)$ the model error term. The feature vectors are composed by all the model parameters from the three neighborhoods in Fig. 10. Furthermore, the error en-

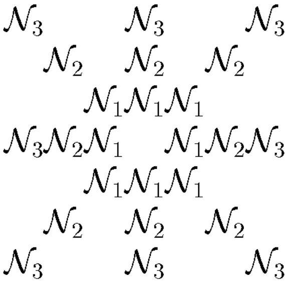
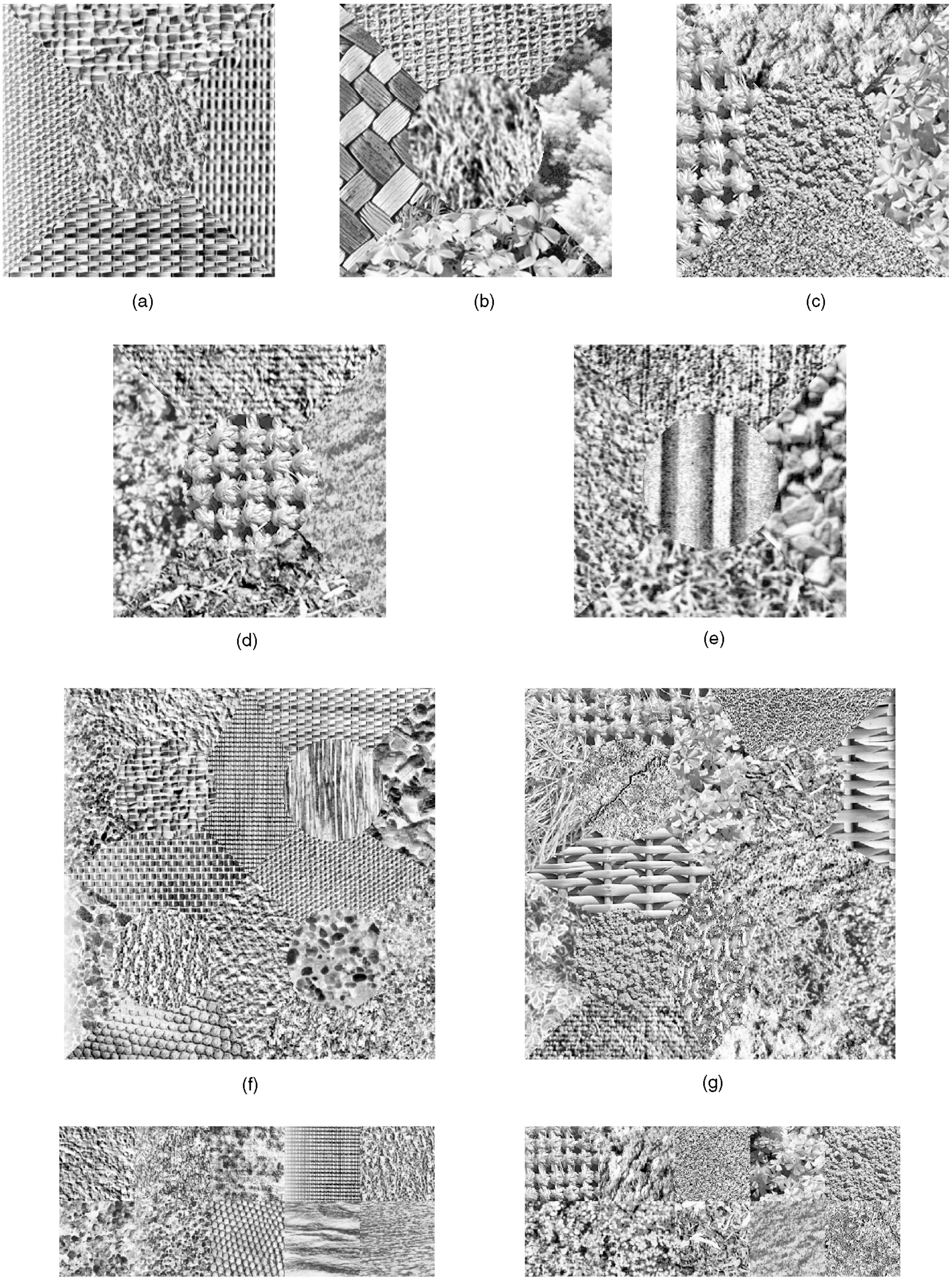

$\mathcal{N}_3$ $\mathcal{N}_3$ $\mathcal{N}_3$
$\mathcal{N}_2$ $\mathcal{N}_2$ $\mathcal{N}_2$
$\mathcal{N}_1\mathcal{N}_1\mathcal{N}_1$
$\mathcal{N}_3\mathcal{N}_2\mathcal{N}_1$ $\mathcal{N}_1\mathcal{N}_2\mathcal{N}_3$
$\mathcal{N}_1\mathcal{N}_1\mathcal{N}_1$
$\mathcal{N}_2$ $\mathcal{N}_2$ $\mathcal{N}_2$
$\mathcal{N}_3$ $\mathcal{N}_3$ $\mathcal{N}_3$

Fig. 10. Neighborhood sets $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ for the AR features. Each $\mathcal{N}_i$ corresponds to one relative pixel position.



(a) (b) (c)

(d) (e)

(f) (g)

(h) (i)

Fig. 11. Composite texture images used in our experiments.

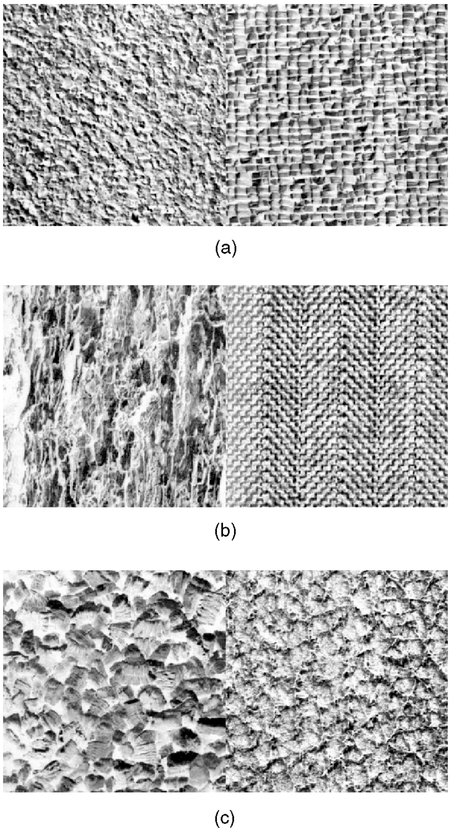
ergy, $\sigma_{\mathcal{E}}$, is used as a feature. The parameters are determined in 25 × 25 pixel overlapping windows. In order to overcome problems with too high feature variances, the features are smoothed. The smoothing filter applied in the experiments here was a Gaussian low-pass filter with $\sigma_S = 3$, see (1).

## 3 Experiments

In our experiments, we evaluate the different feature extraction approaches by performing supervised segmentation on several test images of varying complexity, Figs. 11 and 12. As the feature quality criterion, we use the classification error, "the most common measure of performance for a recognition system" [22].

All filter optimization and classifier training are performed on 256 × 256 subimages of the texture images that are *not* part of the test images. For the classifier design, about 16,000 features were extracted from these 256 × 256 images individually and used for training.

The local energy function used is the one described in Section 2.1, i.e., squaring, Gaussian smoothing, and taking the logarithm. For subsampled filters, the modification of



(a)

(b)

(c)

Fig. 12. The 256 × 512 two-texture images used in our experiments.

Randen and Husøy [17] is used. For filters not having a narrow pass-band, Gaussian low-pass filters with $\sigma_s$ = 8 (empirically determined) are used. As described in Section 2.2, the classifier is LVQ [35] in most cases. However, with the optimized Gabor filters [44], Section 2.4.4 and the optimized FIR filters, Sections 2.4.6 and 2.4.7, thresholding is used. With the optimized prediction error filters, Section 2.4.2, the image is classified by assigning the class label associated with the minimum error predictor. The classifiers and energy functions applied are summarized in Table 1. The method identifiers used in this and subsequent tables are defined in Table 2.

A normalization by a global normalization factor is applied for each feature. The global normalization factors are determined as the factors yielding unity variance for the image in Fig. 11a.

The presentation of the results is divided in three parts. First, in Section 3.1 a discussion of the test images is given. Next, in Section 3.2, heuristically designed, i.e., fixed filter banks are considered. In Section 3.3, critically sampled filter banks (including wavelet transform / packets) are discussed. Finally, in Section 3.4, results with the optimized filters and filter banks are discussed.

### 3.1 Test Images

We have used three different and commonly used texture sources for our test images; the Brodatz album [47], the MIT Vision Texture database [48], and the MeasTex Image Texture Database [49]. Consequently, we use images captured using different equipment and under different conditions. We have selected textures with a granularity which we have judged to be appropriate for the sizes of the test images. Since we train and test on separate portions of the texture images we have selected texture images which are visually stationary, i.e., their visual properties do not change too much over the image.

All textures are gray-scale images when presented to the methods. The dynamic ranges are represented by eight bits per sample. In order to make the textures nondiscriminable for the local mean gray level or local variance, the images have each separately been globally histogram equalized prior to being used.

We have tested simple two-texture images, complicated five-texture images, images with simple borders but as many as ten textures, and images with very complex borders and as many as 16 textures. Since all feature extractors extract features using some window or neighborhood, the complexity and shape of the borders is of major interest. The ability to cope with edges is of course an important feature and the test images should be appropriate in this respect.

The 256 × 256 5-texture images are:

- Fig. 11a, consisting of D77, D84, D55, D53, and D24 from [47],
- Fig. 11b, consisting of Fabric.0000, Fabric.0017, Flowers.0002, Leaves.0006, and Leaves.0013 from [48],
- Fig. 11c, consisting of Fabric.0009, Fabric.0016, Fabric.0019, Flowers.0005, and Food.0005 from [48],
- Fig. 11d, consisting of Fabric.0007, Fabric.0009, Leaves.0003, Misc.0002, and Sand.0000 from [48], and
- Fig. 11e, consisting of Asphalt.0000, Concrete.0001, Grass.0002, Misc.0002, and Rock.0005 from [49].

The 512 × 512 16-texture images are:

- Fig. 11f, consisting of D3, D4, D5, D6, D9, D21, D24, D29, D32, D33, D54, D55, D57, D68, D77, and D84 from [47] and
- Fig. 11g, consisting of Fabric.0007, Fabric.0009, Fabric.0013, Fabric.0014, Fabric.0016, Flowers.0005, Food.0005, Grass.0001, Leaves.0003, Leaves.0008, Leaves.0012, Metal.0000, Metal.0002, Misc.0002, Sand.0000, and Stone.0004 from [48].

The 256 × 640 10-texture images are:

- Fig. 11h, consisting of D4, D9, D19, D21, D24, D28, D29, D36, D37, and D38 from [47] and
- Fig. 11i, consisting of Fabric.0009, Fabric.0016, Fabric.0019, Flowers.0005, Food.0005, Leaves.0003, Misc.0000, Misc.0002, Sand.0000, and Stone.0004 from [48].

Finally, we also have some 256 × 512 texture pair images which are:

- Fig. 12a, consisting of D4 and D84 from [47],
- Fig. 12b, consisting of D12 and D17 from [47], and
- Fig. 12c, consisting of D5 and D92 from [47].

We note that although Fig. 12c *looks* simple, it was very difficult to discriminate for several of the approaches.

TABLE 1
DETAILS ON THE SMOOTHING FILTERS AND CLASSIFIERS USED IN THE EXPERIMENTS

| **Identifier** | **Smoothing filter** | **Classifier** |
|---|---|---|
| Laws filters | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| Ring/wedge filters | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ and $\sigma_s = 16$ | LVQ |
| Dyadic Gabor filter bank | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| Gabor filter bank ($\alpha$) | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| DCT | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| Co-occurrence | N/A | LVQ |
| AR | $\sigma_s = 3$ | LVQ |
| Daubechies $n$ | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| f16b ($\alpha$), f16b ($\alpha$), and f32d ($\alpha$) | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| F_2_1_smpl ($\alpha$) and F_2_1_09 ($\alpha$) | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| Eigenfilter | $\sigma_s = 8$ | LVQ |
| Opt. repr. Gabor filter bank | $\sigma_s = \frac{1}{2\sqrt{2}f_0}$ | LVQ |
| Prediction error filter | $\sigma_s = 8$ | Minimum value selection |
| Optimal Gabor filter $\sigma =?$ | $\sigma_s = 2\sigma$ | Threshold |
| $n$-filter opt. Gabor filter bank | $\sigma_s = 2\sigma$ | LVQ |
| $J_{MS}$ | $\sigma_s = 8$ | Threshold |
| $J_U$ | $\sigma_s = 8$ | Threshold |
| $J_F$ | $\sigma_s = 8$ | Threshold |
| BackProp. NN, mask size $n$ | N/A | Feed-forward neural net |

*For the smoothing filters, the $\sigma_s$-parameter of a Gaussian low-pass filter is given. Any $f_0$ is the subband center frequency and $\sigma$ is the parameter of a Gabor filter in the filter bank.*

TABLE 2
SUMMARY OF IDENTIFIERS USED TO IDENTIFY THE METHODS IN THE RESULT TABLES

| **Identifier** | **Max number of classes** | **Description** | **References** | **Number of features** |
|---|---|---|---|---|
| Laws filters | $\infty$ | Section 2.3.1 | [3] | 25 |
| Ring/wedge filters | $\infty$ | Section 2.3.2 | [12] | 11 |
| Dyadic Gabor filter bank | $\infty$ | Section 2.3.3 | [20] | 20 |
| Gabor filter bank ($\alpha$) | $\infty$ | Section 2.3.7<br>Decomposition of Figure 6($\alpha$) | [17] | (a) 10, (b) 13, (c) 16, (d) 40 |
| DCT | $\infty$ | Section 2.3.5 | [41] | 8 |
| Co-occurrence | $\infty$ | Section 2.5.1 | [36] | 16 |
| AR | $\infty$ | Section 2.5.2 | [2] | 15 |
| Daubechies $n$ ($\alpha$) | $\infty$ | Daubechies $n$-tap wavelet,<br>Decomposition of Figure 6($\alpha$)<br>Section 2.3.4 | [39, 24, 21]<br><br>[26, 11] | (a) 10, (b) 13, (c) 16, (d) 40 |
| f8a ($\alpha$),<br>f16b ($\alpha$), or<br>f32d ( $\alpha$) | $\infty$ | f8a, f16b, or f32d QMF-filter,<br>Decomposition of Figure 6($\alpha$)<br>Section 2.3.6 | [17] | (a) 10, (b) 13, (c) 16, (d) 40 |
| F_2_1_smpl ($\alpha$) or<br>F_2_1_09 ($\alpha$) | $\infty$ | F_2_1_09 or F_2_1_smpl QMF-filter. Decomposition of Figure 6 ($\alpha$). Section 2.3.6 | [17] | (a) 10, (b) 13, (c) 16, (d) 40 |
| Eigenfilter | $\infty$ | Section 2.4.1 | [13] | $\leq 50$ |
| Opt. repr. Gabor filter bank | $\infty$ | Section 2.4.3 | [14, 46] | 2 per texture |
| Prediction error filter | $\infty$ | Section 2.4.2 | [33] | 1 per texture |
| Optimal Gabor filter $\sigma =?$ | 2 | Section 2.4.4 | [44] | 1 |
| $n$-filter opt. Gabor filter bank | $\infty$ | Section 2.4.5 | [15, 16] | $n$ |
| $J_{MS}$ | $2/\infty$ | Sections 2.4.6, 2.4.6, and 2.4.7 | [31] | Variable |
| $J_U$ | $2/\infty$ | Sections 2.4.6 and 2.4.7 | [33, 18] | Variable |
| $J_F$ | 2 | Section 2.4.6 | [33, 18] | Variable |
| BackProp. NN, mask size $n$ | $\infty$ | Section 2.4.8 | [45] | N/A |

From the experiments, we will see the need for a wide variety of test images when assessing the relative performances of texture classification methods. If we look at for example the optimal representation Gabor filter bank, it is the best optimized approach for Fig. 11a and fails completely for Fig. 11b.

Another observation is that the texture pair of Fig. 12c is rather difficult to discriminate for several approaches. It

TABLE 3
CLASSIFICATION ERRORS FOR DIFFERENT HEURISTICALLY DESIGNED TEXTURE FEATURE EXTRACTORS

| **Heuristical Method** | Test image (figure number) | | | | | | | | | | | | Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 11(a) | 11(b) | 11(c) | 11(d) | 11(e) | 11(f) | 11(g) | 11(h) | 11(i) | 12(a) | 12(b) | 12(c) | |
| Laws filters | 9.7 | 25.7 | 32.4 | 27.3 | 25.7 | 48.3 | 54.3 | 41.9 | 87.9 | 24.8 | 3.3 | 8.6 | 32.5 |
| Ring/wedge filters | 14.6 | 35.5 | 28.9 | 35.5 | 22.4 | 43.8 | 67.8 | 44.5 | 89.5 | 3.0 | 4.5 | 18.4 | 34.0 |
| Dyadic Gabor filt.bk. | 10.7 | 34.8 | 22.6 | 25.2 | 24.6 | 60.1 | 58.2 | 32.3 | 88.4 | 3.6 | 2.0 | 11.3 | 31.1 |
| Gabor filter bank (d) | 8.2 | 34.0 | 25.8 | 36.9 | 28.4 | 54.8 | 71.5 | 39.7 | 88.9 | 6.5 | 1.2 | 15.6 | 34.3 |
| DCT | 13.2 | 27.0 | 25.5 | 37.8 | 22.6 | 40.9 | 49.0 | 38.2 | 86.8 | 6.4 | 2.2 | 2.5 | 29.3 |
| Daubechies 4 (d) | 8.7 | 22.8 | 25.0 | 23.4 | 21.8 | 38.2 | 45.2 | 40.9 | 89.5 | 5.7 | 0.6 | 8.2 | 27.5 |
| f16b (d) | 8.7 | 18.9 | 23.3 | 18.4 | 17.2 | 36.4 | 41.7 | 39.8 | 90.5 | 8.1 | 0.8 | 8.2 | 26.0 |
| Co-occurrence | 9.9 | 27.0 | 26.1 | 51.1 | 35.7 | 49.6 | 55.4 | 35.3 | 84.9 | 1.9 | 4.8 | 3.3 | 32.1 |
| AR | 19.6 | 19.4 | 23.0 | 23.9 | 24.0 | 58.0 | 46.4 | 56.7 | 88.6 | 11.3 | 1.9 | 3.0 | 31.3 |
| Average | 11.5 | 27.2 | 25.9 | 31.1 | 24.7 | 47.8 | 54.4 | 41.0 | 88.3 | 7.9 | 2.4 | 8.8 | 30.9 |

must be kept in mind that 50 percent error would be attained by randomly picking a class assignment. Hence, 15-25 percent error is significant.

We also observe that the 10-texture image of Fig. 11i gives close to random performance for all approaches (random picking of a class assignment would yield 10 percent chance of correct class assignment, i.e., 90 percent error). In essence, none of the approaches are giving any useful results for this image. We do not have any explanation of the poor performance on this image, but include it to stimulate research on how to improve texture classification methods.

## 3.2 Experiments with the Heuristically Designed Filter Banks

The results for most of the heuristically designed feature extractors are presented in Table 3. For the wavelet frame experiments only one variant is listed in Table 3. The same is the case with the QMF and arbitrarily decomposed Gabor filter bank experiments. More detailed results from these filter classes are given in Tables 4 and 5.

In order to get an increased insight from the experiments, there are multiple "dimensions" in the results. Some approaches allow variations of the filter responses and subband decomposition, some are fixed on filter response, some on decomposition, and some on both. The number of feature images also vary much, from eight to 40 features, having considerable impact on the computational complexity and memory requirements. The complexity, subband decomposition, and filter response issues will be discussed later, but first an overall discussion of the results is given.

### 3.2.1 Overall Discussion

No clear hierarchy of classification performances is observed in Table 3. For some images, some approaches are good, for different images, others are good. No methods are consistently poor, but many methods have examples with poor relative performance.

Considering Table 3, evidently the "old" Laws and ring / wedge filters are not very good. They never stand out as clear winners, and particularly the Laws approach fail completely in the case of the simple image in Fig. 12a. One possible explanation might be that the frequency spectra of the filters are too smooth, giving too poor separation of the frequency components of the textures. Furthermore, we also have an indication that the basic assumption behind the ring / wedge filters that frequency and orientation can be treated separately is not true.

The commonly referred dyadic Gabor filter bank show rather poor performance for some of the test images, Figs. 11b, 11d, 11f and Fig. 12c. Similar comments apply to the arbitrary Gabor filter bank. Compared to the best approaches, also these approaches have smooth frequency responses of the channel filters. This leads to poor separation of the frequency content in the image, which might account for the moderate performance.

TABLE 4
CLASSIFICATION ERRORS FOR DIFFERENT WAVELET, GABOR, AND QMF FULL RATE FILTER BANKS

| **Full rate Basis** | Decomposition (fig. #) | | | |
|---|---|---|---|---|
| | 6(a) | 6(b) | 6(c) | 6(d) |
| Gabor filter bk. | 41.7 | 37.4 | 46.5 | 34.3 |
| Daubechies 4 | 34.0 | 33.6 | 37.8 | 27.5 |
| Daubechies 6 | 33.8 | 34.8 | 40.2 | 28.2 |
| Daubechies 8 | 33.9 | 36.7 | 42.7 | 29.7 |
| Daubechies 10 | 34.7 | 38.6 | 45.2 | 32.3 |
| f8a | 33.1 | 33.2 | 37.2 | 26.3 |
| f16b | 32.7 | 33.0 | 36.2 | 26.0 |
| f32d | 32.9 | 33.9 | 33.7 | 27.0 |
| F_2_1_smpl | 32.7 | 32.6 | 35.7 | 26.6 |
| F_2_1_09 | 33.5 | 34.0 | 39.7 | 27.7 |

*For details on the individual experiments, see Table 8. The numbers represent the mean classification errors over all test images.*

TABLE 5
CLASSIFICATION ERRORS FOR DIFFERENT WAVELET TRANSFORMS/PACKETS AND CRITICALLY DECIMATED QMF FILTERS

| **Decimated Basis** | Decomposition (fig. #) | | | |
|---|---|---|---|---|
| | 6(a) | 6(b) | 6(c) | 6(d) |
| Daubechies 4 | 40.1 | 38.3 | 41.6 | 32.4 |
| Daubechies 6 | 38.5 | 37.4 | 41.4 | 32.2 |
| Daubechies 8 | 39.1 | 39.8 | 42.9 | 33.5 |
| Daubechies 10 | 38.9 | 40.6 | 46.2 | 35.6 |
| f8a | 38.6 | 36.0 | 38.0 | 30.2 |
| f16b | 36.6 | 34.9 | 36.6 | 29.3 |
| f32d | 35.5 | 34.9 | 35.8 | 29.1 |
| F_2_1_smpl | 36.6 | 35.9 | 40.2 | 32.1 |
| F_2_1_09 | 38.3 | 37.9 | 42.1 | 33.8 |

*For details on the individual experiments, see Table 9. The numbers represent the mean classification errors over all test images.*

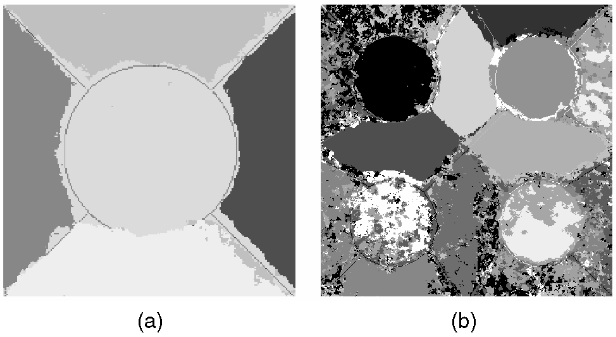


Fig. 13. Classification results for the QMF "f16b" filter bank for the textures in Figs. 11a and 11f.

The DCT approach did also show poor relative performance for a few images, Figs. 11a and 11d and Fig. 12a. On the overall results, however, it was only beaten by the best QMF and wavelet approaches with 40-dimensional feature spaces. Compared to the 8-feature DCT approach this is remarkable.

With the QMF and wavelet approaches, only the best decomposition/filter response combinations are shown in Table 3. More results are given in Table 4. The QMF and wavelet frame approaches are among the best for most images. Examples of the classification results for one QMF filter bank (f16b (d)) are shown in Fig. 13. These results correspond to the textures in Figs. 11a and 11f. The QMF f16b (d) is the overall best approach.

Also the popular co-occurrence method gives rather poor performance, giving actually the worst results in three cases, Figs. 11d and 11e and Fig. 12b. The AR features were slightly better for most images, but they were giving the worst results in two cases, Figs. 11a and 11h. The best filtering approaches yield classifications that are as good as or better than the co-occurrence and AR methods in practically all cases.

To summarize, Laws and ring/wedge filters were never found to perform best, while each of the other approaches had at least one test image where it performed best. The best average results and the highest number of best cases is observed for the QMF filter bank f16b (d). Hence, we may conclude that f16b (d) is the winner and the Laws and ring/wedge filters are the losers. If computational complexity is an important issue however, the DCT approach is the winner due to its good overall performance and low complexity. We may also conclude that the popular Gabor filter and co-occurrence features are clearly not superior.

Finally, remember that it is a quite common practice in texture classification to pick training data from the test features as discussed in Section 2.2.2. This practice has not been used in these experiments. In order to simulate a real application environment, all classifier training has been performed on features from separate subimages. Furthermore, no edge or texture border specific training data are picked. These issues must be kept in mind when comparing the results with previously published results. For a further analysis of the impact of this choice, see Section 2.2.

### 3.2.2 Filter Response

Table 4 gives the numbers for the performances of the wavelet frames, QMF, and arbitrary Gabor filter banks. Careful examination of the results tells us that there are consistent trends with respect to the performances of various filter responses. The most clear conclusion from the results is that the performance of the Gabor filters is inferior to the wavelet and QMF filter classes. This despite the fact that the Gabor filter is the preferred filter in several works [9], [14], [15], [16], [20], [25], [44], [50].

Next, we may also see an overall detoriation in the results when we move from shorter Daubechies wavelet bases to longer bases. Note that the numbers in the identifiers for the QMF and wavelet filters represent number of coefficients in the prototype low-pass filter. A high number typically corresponds to a wide impulse response but a crisp frequency response, and vice versa, see Figs. 9 and 7.

What do we learn from these experiments? Intuitively, we would desire a filter with a compact representation in the spatial domain (due to edge effects), and good frequency response crispness and high stop band attenuation (in order to discriminate the textures' spectra). However, the poor performance of the Gabor filter (which has optimal joint resolution in the spatial and the frequency domains) indicates that optimal *joint* resolution is no ultimate goal.

Can we then tell whether spatial resolution is more important than frequency resolution, or vice versa? The QMF filters have significantly crisper frequency responses than the Gabor filters, but also wider impulse responses. Since these filters give significantly better results than the Gabor filters, we may conclude that crisp frequency response is an important issue in texture classification. On the other hand, the variations in the different QMF filter crispnesses do not have a major performance impact.

As we can see, the QMF filters are overall giving better results than the Daubechies wavelets. This is probably attributable to the significant asymmetric unit pulse responses of the Daubechies wavelets. A nonsymmetric filter response will lead to a consistent edge detection error and higher classification errors. As we might expect, the Daubechies wavelets also give more error the longer the impulse response, i.e., the larger the asymmetry.

### 3.2.3 Subband Decomposition

Except for the images where the methods fail (like Fig. 11i), the choice of subband decomposition is very significant. The clear winner of the decompositions is the decomposition of Fig. 6d. This is the overall best decomposition for all filters. This is also the decomposition with most subbands, i.e., highest feature dimensionality: 40.

It might seem like the expected conclusion that the decomposition with most features yields the best results. This would indeed be the case if we did not use separate test and training datasets. However, it should be kept in mind that more features makes it more difficult for the classifier to generalize, thus more features actually presents an increased challenge to the classification.

Decompositions of Figs. 6a-6c are all dyadic, also known as wavelet decompositions. The more levels of the dyadic decomposition, the more the low frequency band is

TABLE 6
CLASSIFICATION ERRORS FOR DIFFERENT OPTIMIZED TEXTURE FEATURE EXTRACTORS

| **Optimized approaches** | Test image (figure number) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | 11(a) | 11(b) | 11(c) | 11(d) | 11(e) | 11(f) | 11(g) | 11(h) | 11(i) | 12(a) | 12(b) | 12(c) | Mean |
| Eigenfilter | 12.1 | 27.0 | 25.0 | 29.7 | 23.4 | 56.1 | 57.9 | 37.1 | 88.0 | 4.0 | 3.7 | 4.7 | 30.7 |
| Opt. repr. Gabor filter bank | 7.2 | 60.5 | 36.4 | 27.4 | 42.2 | 37.2 | 75.3 | 41.9 | 89.5 | 5.5 | 5.8 | 36.2 | 38.8 |
| Prediction error filter | 14.7 | 31.1 | 27.0 | 34.0 | 42.5 | 34.7 | 48.7 | 44.6 | 85.6 | 0.9 | 4.6 | 31.7 | 33.3 |
| Optimal Gabor filter $\sigma = 2$ | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 10.4 | 3.6 | 10.8 | N/A |
| Optimal Gabor filter $\sigma = 4$ | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 5.3 | 3.3 | 11.6 | N/A |
| Optimal Gabor filter $\sigma = 8$ | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 5.7 | 5.2 | 11.4 | N/A |
| Optimal Gabor filter $\sigma = 16$ | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 14.0 | 13.7 | 24.2 | N/A |
| 4-filter opt. Gabor filter bank | 24.7 | 54.5 | 34.2 | 33.3 | 34.0 | 54.0 | 74.0 | 52.8 | 88.9 | 4.3 | 3.8 | 18.7 | 39.8 |
| 6-filter opt. Gabor filter bank | 16.3 | 34.5 | 33.9 | 33.6 | 29.3 | 49.4 | 61.0 | 50.3 | 89.5 | 3.8 | 3.8 | 17.1 | 35.2 |
| 10-filter opt. Gabor filter bank | 16.1 | 29.4 | 31.3 | 34.5 | 27.8 | 40.7 | 54.7 | 39.9 | 87.8 | 4.1 | 3.9 | 12.8 | 31.9 |
| $J_{MS}$ | 16.9 | 36.3 | 32.7 | 41.1 | 43.0 | 47.3 | 51.1 | 59.7 | 85.0 | 0.7 | 8.3 | 28.6 | 37.6 |
| $J_U$ | 12.7 | 33.0 | 26.5 | 34.3 | 43.4 | 45.6 | 46.5 | 35.9 | 86.1 | 0.7 | 2.1 | 5.1 | 31.0 |
| $J_F$ | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 0.7 | 2.1 | 5.1 | N/A |
| BackProp. NN mask size 11 | 47.4 | 65.0 | 71.4 | 67.2 | 55.7 | 61.1 | 74.0 | 69.3 | 93.3 | 32.1 | 30.0 | 43.6 | 59.2 |
| BackProp. NN mask size 21 | 72.4 | 68.7 | 69.0 | 74.8 | 70.0 | 84.9 | 83.5 | 76.5 | 90.8 | 30.8 | 28.8 | 48.6 | 66.6 |
| Average | 24.1 | 44.0 | 38.7 | 41.0 | 41.1 | 51.1 | 62.7 | 50.8 | 88.4 | 8.2 | 8.2 | 20.7 | 40.4 |

*See Table 2 for an overview of the method descriptions.*

split—and the more features. Since we may expect textures to have their characteristic features in higher frequency bands, we would not expect that increasing the depth of the decomposition would increase the performance. This is indeed seen on the results as well, where we can see a consistent lack of improvement when the depth is increased. In fact as the depth becomes as high as five (Fig. 6c), the performance *drops* consistently probably due to the mentioned problems for the classifier to generalize.

### 3.2.4 Complexity Issues

When comparing the results, complexity issues should also be kept in mind. Most of the filtering approaches are significantly less computationally complex than the co-occurrence and AR methods. For the filtering approaches, filtering and classification are the main contributors to the total complexity of the system. The DCT approach has low filtering complexity (short separable filter masks with fast implementation schemes) and low feature dimensionality, generally yielding relatively low classifier complexity. The Laws, ring and wedge, Gabor, and some of the wavelet and QMF filter banks have high filtering complexities, whereas the IIR QMF filters have low complexities. Low feature count yields low classification complexity. In our setup the complexity contribution from the LVQ classifier and the feature extractors were of the same order. Hence classifier complexity is an important component in the overall complexity.

### 3.2.5 Summary

From Tables 3 and 4, no single feature extraction method is *consistently* superior. The main insight we may learn from this, is that when designing a filter bank for a particular texture problem, it is advantageous to test multiple approaches. We do, however, see patterns in the results that may exclude several of the approaches from such a study.

An inherent assumption made when filter banks are used for texture feature extraction is that the spectral signatures of different textures are different,[3] see Section 2. For problems with many textures or textures with subtle spectral differences, it is reasonable to assume that the spectral decomposition has a significant effect. This intuitive idea is supported by the experiments, and we see great importance of the filter classes and decompositions for difficult two-texture images and for most multi-texture images.

In many circumstances, the effort in testing numerous filter banks may be too large. For these applications, one filter bank should be recommended. This filter bank should have a reasonably good performance in most cases. One such choice is the bank with the QMF filter f16b using the sub-band decomposition in Fig. 6d. However, in several cases this 40-dimensional filter bank may be too computationally complex. A viable alternative with significantly lower feature dimensionality is the DCT approach, which we would recommend if lower complexity is required.

## 3.3 Experiments with Critically Sampled Filter Banks

Considering the full rate QMF and wavelet frame approaches, Table 4, relative to the results from the critically

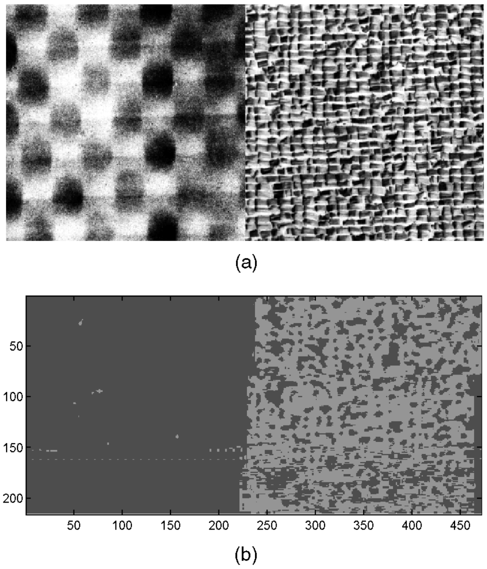


Fig. 14. (a) Texture pair D8-D84 [47] and (b) supervised segmentation by the back propagation designed mask [45].

[3] Analog discussions do also apply to other representation spaces than the frequency space.

TABLE 7
FEATURE DIMENSIONALITY FOR THE DIFFERENT FEATURE EXTRACTION SCHEMES

| **Heuristical Method** | Test image (figure number) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 11(a) | 11(b) | 11(c) | 11(d) | 11(e) | 11(f) | 11(g) | 11(h) | 11(i) | 12(a) | 12(b) | 12(c) |
| Laws filters | 25 | 25 | 25 | 25 | 25 | 25 | 25 | 25 | 25 | 25 | 25 | 25 |
| Ring/wedge filters | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| Dyadic Gabor filt.bk. | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| Decomposition of Fig. 6(a) | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| Decomposition of Fig. 6(b) | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 |
| Decomposition of Fig. 6(c) | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| Decomposition of Fig. 6(d) | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 |
| DCT | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| Co-occurrence | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| AR | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 |
| Eigenfilter | 40 | 37 | 45 | 44 | 36 | 48 | 48 | 50 | 50 | 18 | 16 | 16 |
| Opt. repr. Gabor filter bank | 10 | 10 | 10 | 10 | 10 | 32 | 32 | 20 | 20 | 4 | 4 | 4 |
| Prediction error filter | 5 | 5 | 5 | 5 | 5 | 16 | 16 | 10 | 10 | 2 | 2 | 2 |
| Optimal Gabor filter | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 1 | 1 | 1 |
| $n$-filter opt. Gabor filter bank | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ | $n$ |
| $J_{MS}$ | 4 | 4 | 4 | 4 | 7 | 46 | 21 | 12 | 12 | 1 | 1 | 1 |
| $J_U$ | 4 | 4 | 4 | 4 | 7 | 25 | 31 | 15 | 9 | 1 | 1 | 1 |
| $J_F$ | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 1 | 1 | 1 |
| BackProp. NN | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 15 |

sampled QMF and wavelet transform / packet approaches, Table 5, a few conclusions may be drawn. First of all, the filter responses for the critically sampled QMF filters, the wavelet transforms and the wavelet packets are simply subsampled versions of the ones obtained by the full rate or over-complete counterparts. We see that, as expected, using decimated filter outputs overall degrades the results. The major cause is worse edge resolution due to the subsampling. However, the *significantly decreased computational complexity* [33] should be kept in mind.

Furthermore, we see that the trend in the results is the same with the critically sampled approaches as it was with the full rate approaches. That is, image, filter class, and decomposition combinations that yielded poor relative results in full rate do also yield poor results when critically sampled, and vice versa.

## 3.4 Experiments with Optimized Filters

The results from the optimized filtering experiments are presented in Table 6. Some of the optimization approaches are only applicable for two-texture cases, and the multi-texture table entries for these are marked by “N / A.”

Despite the fact that the training times for the *back propagation designed masks* were significant, the results were not useful for *any* of the images in this test. Useful results were only obtained for some very simple texture pairs. One example is shown in Fig. 14. Improvements using a modified scheme have been reported [45], but has for implementational and complexity reasons not been tested here.

The *eigenfilter* approach is the overall best optimized approach. However, it must be remembered that the number of filters is relatively high with this approach. For two-texture images, the number of filters is typically in the range 16-18, while for multi-texture images it is in the range 36-50. Hence, the argument of low complexity versus the heuristic filters is not applicable. For reference we have made a list of number of features for the different feature extraction approaches in Table 7.

The *optimized representation Gabor* filter bank was the only optimized filtering approach doing a really good job on the image of Fig. 11a. However, for most of the other test images, it is only giving similar, and even in some cases (Figs. 11b, 11e, and 11g and Fig. 12c) significantly worse results than the best optimized filtering approaches, despite high feature dimensionality. We assume that the problems experienced with this approach are due to the fact that it is optimized with respect to representation, not discrimination.

The best result for the image in Fig. 11f in this study was obtained with the *prediction error filters*. However, despite these nice results, a remarkably poor result was obtained with the image of Fig. 12c and relatively poor results with the images of Figs. 11a and 11e. Hence, this approach is not very robust.

It is interesting to observe the performances of the three approaches optimizing filters with respect to representation, i.e., the eigenfilters, the optimized representation Gabor filter bank and the prediction error filters. For the latter two we do observe some cases with *breakdown* in performance, particularly for Fig. 11e and Fig. 12c, most likely corresponding to cases where textures are too similar for discrimination. Close examination of the confidence matrices[4] for these cases support this assumption, i.e., a few of the textures were severely misclassified while other textures had nice classification results. When the eigenfilter approach do not have the same problems with breakdowns, this might be due to the high number of features and the versatile classification algorithm which compensate for the problems.

The results from the *optimized Gabor filters* and *optimized Gabor filter banks* are fairly good, but the results indicate problems with similar textures—especially when using only one or a few filters. In particular, the results for the

[4] The confidence matrix shows how the misclassification is distributed among the textures in the image, i.e., how many percent of texture 1 was classified as belonging to texture class 2, etc.

TABLE 8
DETAILED SUMMARY OF THE CLASSIFICATION ERRORS FOR DIFFERENT WAVELET, GABOR, AND QMF FULL RATE FILTER BANKS

| **Full rate** | Test image (figure number) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | 11(a) | 11(b) | 11(c) | 11(d) | 11(e) | 11(f) | 11(g) | 11(h) | 11(i) | 12(a) | 12(b) | 12(c) | Mean |
| Gabor filter bk. (a) | 14.9 | 37.2 | 42.9 | 54.8 | 38.2 | 60.4 | 87.6 | 45.9 | 86.9 | 9.2 | 2.0 | 20.9 | 41.7 |
| Gabor filter bk. (b) | 13.0 | 46.0 | 27.9 | 39.3 | 33.9 | 60.1 | 72.2 | 43.8 | 88.0 | 6.9 | 1.5 | 16.5 | 37.4 |
| Gabor filter bk. (c) | 18.6 | 56.2 | 39.4 | 75.8 | 37.7 | 62.7 | 78.4 | 53.7 | 90.3 | 7.9 | 9.5 | 27.5 | 46.5 |
| Gabor filter bk. (d) | 8.2 | 34.0 | 25.8 | 36.9 | 28.4 | 54.8 | 71.5 | 39.7 | 88.9 | 6.5 | 1.2 | 15.6 | 34.3 |
| Daubechies 4 (a) | 11.6 | 26.1 | 35.8 | 36.6 | 27.8 | 48.8 | 60.9 | 41.2 | 88.8 | 15.5 | 1.3 | 13.4 | 34.0 |
| Daubechies 4 (b) | 12.7 | 30.8 | 27.2 | 31.7 | 32.0 | 46.7 | 54.9 | 44.5 | 88.8 | 19.2 | 1.0 | 13.5 | 33.6 |
| Daubechies 4 (c) | 23.8 | 46.2 | 33.4 | 33.6 | 40.3 | 60.1 | 59.5 | 47.0 | 85.9 | 13.1 | 4.0 | 6.5 | 37.8 |
| Daubechies 4 (d) | 8.7 | 22.8 | 25.0 | 23.4 | 21.8 | 38.2 | 45.2 | 40.9 | 89.5 | 5.7 | 0.6 | 8.2 | 27.5 |
| Daubechies 6 (a) | 10.7 | 27.8 | 36.8 | 36.6 | 26.9 | 46.4 | 59.6 | 43.2 | 88.5 | 16.1 | 0.9 | 12.3 | 33.8 |
| Daubechies 6 (b) | 15.9 | 34.1 | 27.9 | 31.2 | 32.9 | 50.8 | 56.8 | 47.0 | 88.4 | 20.3 | 0.7 | 11.4 | 34.8 |
| Daubechies 6 (c) | 23.4 | 52.2 | 39.4 | 37.4 | 44.9 | 63.3 | 61.1 | 50.1 | 87.5 | 11.9 | 4.6 | 6.4 | 40.2 |
| Daubechies 6 (d) | 10.7 | 22.1 | 30.0 | 21.3 | 21.7 | 39.7 | 46.1 | 44.3 | 90.8 | 4.7 | 0.6 | 6.1 | 28.2 |
| Daubechies 8 (a) | 14.7 | 29.0 | 35.2 | 36.1 | 25.5 | 46.5 | 60.1 | 44.8 | 89.8 | 14.8 | 1.0 | 9.3 | 33.9 |
| Daubechies 8 (b) | 19.9 | 36.3 | 34.8 | 31.7 | 34.9 | 53.2 | 61.4 | 51.9 | 90.0 | 17.8 | 0.4 | 8.0 | 36.7 |
| Daubechies 8 (c) | 34.4 | 50.0 | 44.8 | 38.1 | 48.8 | 68.9 | 62.1 | 53.5 | 88.5 | 13.0 | 3.3 | 6.9 | 42.7 |
| Daubechies 8 (d) | 12.1 | 25.0 | 32.5 | 24.4 | 24.0 | 43.8 | 47.7 | 45.0 | 90.5 | 4.9 | 1.3 | 4.7 | 29.7 |
| Daubechies 10 (a) | 18.2 | 31.5 | 33.9 | 35.2 | 27.3 | 47.3 | 60.8 | 46.4 | 90.2 | 15.4 | 0.9 | 9.0 | 34.7 |
| Daubechies 10 (b) | 24.3 | 39.9 | 40.0 | 32.4 | 39.0 | 57.4 | 63.9 | 51.9 | 88.4 | 17.9 | 0.5 | 7.4 | 38.6 |
| Daubechies 10 (c) | 39.3 | 51.5 | 48.6 | 44.1 | 53.8 | 72.8 | 63.4 | 56.9 | 87.9 | 14.5 | 2.4 | 7.2 | 45.2 |
| Daubechies 10 (d) | 15.9 | 28.1 | 38.1 | 29.1 | 27.6 | 48.7 | 50.6 | 48.5 | 90.7 | 4.4 | 1.2 | 5.1 | 32.3 |
| f8a (a) | 8.9 | 23.6 | 37.4 | 37.3 | 24.3 | 41.5 | 60.2 | 46.8 | 89.6 | 15.5 | 0.5 | 11.7 | 33.1 |
| f8a (b) | 11.4 | 29.6 | 29.0 | 27.9 | 27.1 | 45.1 | 57.5 | 48.3 | 90.8 | 19.7 | 0.4 | 11.2 | 33.2 |
| f8a (c) | 29.9 | 40.3 | 31.7 | 36.7 | 31.2 | 55.8 | 60.3 | 46.9 | 87.6 | 15.0 | 6.8 | 3.9 | 37.2 |
| f8a (d) | 7.2 | 21.1 | 23.7 | 18.6 | 18.6 | 37.5 | 43.2 | 40.1 | 90.9 | 6.5 | 0.6 | 7.2 | 26.3 |
| f16b (a) | 9.5 | 24.0 | 35.5 | 36.1 | 22.0 | 42.4 | 58.1 | 47.0 | 89.9 | 16.0 | 0.3 | 11.2 | 32.7 |
| f16b (b) | 13.2 | 29.4 | 28.0 | 23.7 | 26.9 | 45.9 | 56.6 | 48.9 | 90.1 | 21.6 | 0.4 | 10.9 | 33.0 |
| f16b (c) | 28.4 | 39.2 | 32.7 | 32.1 | 29.0 | 55.1 | 57.6 | 46.6 | 87.2 | 16.7 | 6.2 | 4.1 | 36.2 |
| f16b (d) | 8.7 | 18.9 | 23.3 | 18.4 | 17.2 | 36.4 | 41.7 | 39.8 | 90.5 | 8.1 | 0.8 | 8.2 | 26.0 |
| f32d (a) | 9.9 | 24.3 | 37.6 | 35.7 | 21.9 | 44.7 | 59.0 | 47.2 | 90.8 | 14.6 | 0.2 | 8.5 | 32.9 |
| f32d (b) | 16.2 | 31.6 | 30.7 | 24.6 | 28.1 | 49.8 | 55.7 | 49.5 | 91.2 | 20.2 | 0.4 | 9.1 | 33.9 |
| f32d (c) | 20.9 | 31.0 | 27.9 | 24.1 | 29.0 | 49.6 | 60.1 | 48.0 | 87.9 | 17.9 | 1.6 | 5.8 | 33.7 |
| f32d (d) | 10.8 | 19.7 | 23.8 | 16.8 | 19.1 | 40.3 | 42.6 | 43.2 | 91.6 | 9.0 | 0.5 | 6.8 | 27.0 |
| F_2_1_smpl (a) | 8.9 | 25.4 | 35.3 | 35.7 | 23.8 | 40.7 | 58.0 | 47.2 | 90.3 | 15.3 | 1.5 | 10.4 | 32.7 |
| F_2_1_smpl (b) | 15.0 | 30.2 | 29.6 | 25.0 | 22.5 | 43.8 | 55.1 | 46.8 | 90.6 | 20.1 | 2.4 | 10.4 | 32.6 |
| F_2_1_smpl (c) | 22.8 | 41.9 | 38.9 | 27.0 | 27.7 | 60.8 | 52.9 | 42.9 | 89.0 | 13.8 | 4.3 | 6.9 | 35.7 |
| F_2_1_smpl (d) | 12.3 | 19.7 | 20.6 | 18.3 | 20.3 | 37.1 | 41.9 | 41.7 | 90.5 | 7.2 | 2.3 | 7.4 | 26.6 |
| F_2_1_09 (a) | 11.2 | 25.3 | 35.7 | 35.7 | 27.1 | 44.3 | 58.7 | 47.1 | 90.7 | 14.8 | 1.2 | 10.6 | 33.5 |
| F_2_1_09 (b) | 21.2 | 29.6 | 27.7 | 25.6 | 27.7 | 47.0 | 57.5 | 49.4 | 89.0 | 21.2 | 2.3 | 10.1 | 34.0 |
| F_2_1_09 (c) | 33.9 | 45.5 | 41.3 | 35.8 | 34.1 | 58.8 | 56.4 | 49.6 | 89.2 | 18.6 | 6.2 | 7.6 | 39.7 |
| F_2_1_09 (d) | 15.7 | 20.1 | 22.5 | 18.4 | 23.2 | 39.8 | 43.6 | 44.2 | 89.8 | 6.7 | 2.1 | 6.0 | 27.7 |
| Average | 17.2 | 33.6 | 33.5 | 33.2 | 30.3 | 50.7 | 58.2 | 46.6 | 89.2 | 13.1 | 2.2 | 10.0 | 34.8 |

*The averages are summarized in Table 4.*

images of Figs. 11a and 12c are noted. On the other hand, the optimized Gabor filters yield one-dimensional feature spaces for two textures. Hence, class labels may be assigned to the feature values by thresholding. Thresholding has a very low computational complexity compared to classifiers like the LVQ. Furthermore, training the threshold classifier means determining *one* scalar parameter per feature image. This is clearly a simpler task than training the parameters of, say, the LVQ classifier.

Of the optimized FIR filters, the clearly best and most versatile criterion is the $J_U$ criterion, while the $J_{MS}$ gives poorer performance for most images. The $J_F$ criterion does not give any improvement over the $J_U$ criterion for any of the texture pairs and it is not applicable to multi-texture images. For two-texture images, the $J_U$-optimized filters are the winners of this study. Overall, they outperform all other approaches, optimized and heuristic, irrespective of number of features for texture-pair images. Since these approaches yield only one feature image that may be classified by thresholding, they are also among the least computational demanding for two-texture images.

Even for multi-texture images, filters optimized with respect to $J_U$ are among the best optimized filters, except for the image in Fig. 11e. Compared to the heuristic approaches on the other hand, the performance for the multi-textured images is not very good. However we note that these schemes are designed to utilize thresholding as the classification scheme also for multi-texture images, and that the filtering and classification scheme for multi-texture images is hierarchical. Hence, the computational complexity is very low which, in some cases, may make this approach viable still.

TABLE 9
DETAILED SUMMARY OF THE CLASSIFICATION ERRORS FOR DIFFERENT WAVELET TRANSFORMS/PACKETS AND CRITICALLY DECIMATED QMF FILTERS

| **Decimated** | Test image (figure number) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | 11(a) | 11(b) | 11(c) | 11(d) | 11(e) | 11(f) | 11(g) | 11(h) | 11(i) | 12(a) | 12(b) | 12(c) | Mean |
| Daubechies 4 (a) | 22.8 | 31.3 | 43.8 | 46.4 | 31.1 | 58.1 | 68.7 | 49.9 | 87.9 | 22.5 | 2.8 | 15.5 | 40.1 |
| Daubechies 4 (b) | 22.6 | 32.8 | 32.0 | 36.2 | 34.1 | 52.5 | 66.3 | 53.0 | 87.5 | 25.7 | 1.1 | 15.6 | 38.3 |
| Daubechies 4 (c) | 22.7 | 39.2 | 37.0 | 47.9 | 39.8 | 66.5 | 71.4 | 52.6 | 86.5 | 23.4 | 2.4 | 9.7 | 41.6 |
| Daubechies 4 (d) | 11.9 | 26.0 | 34.1 | 32.7 | 26.9 | 41.5 | 57.3 | 48.0 | 88.6 | 11.0 | 1.3 | 10.0 | 32.4 |
| Daubechies 6 (a) | 17.6 | 32.1 | 40.1 | 44.5 | 30.6 | 56.2 | 67.5 | 47.1 | 88.3 | 22.2 | 2.7 | 12.9 | 38.5 |
| Daubechies 6 (b) | 15.2 | 36.4 | 34.4 | 38.5 | 32.1 | 55.2 | 64.7 | 47.4 | 87.2 | 23.3 | 1.8 | 13.2 | 37.4 |
| Daubechies 6 (c) | 29.7 | 38.8 | 43.0 | 40.8 | 37.2 | 65.8 | 63.4 | 56.4 | 86.6 | 19.7 | 3.6 | 12.1 | 41.4 |
| Daubechies 6 (d) | 12.3 | 25.3 | 36.7 | 30.3 | 23.9 | 44.8 | 55.9 | 46.6 | 90.6 | 11.2 | 1.4 | 7.9 | 32.2 |
| Daubechies 8 (a) | 21.4 | 33.0 | 37.4 | 46.3 | 31.9 | 56.7 | 68.8 | 51.0 | 88.4 | 21.5 | 2.2 | 10.6 | 39.1 |
| Daubechies 8 (b) | 27.2 | 37.4 | 36.2 | 36.0 | 37.5 | 58.2 | 67.7 | 53.2 | 89.3 | 25.9 | 1.3 | 8.1 | 39.8 |
| Daubechies 8 (c) | 32.6 | 50.6 | 48.1 | 42.3 | 41.3 | 62.6 | 64.2 | 54.9 | 87.0 | 21.7 | 1.8 | 7.3 | 42.9 |
| Daubechies 8 (d) | 13.9 | 30.8 | 37.6 | 33.1 | 27.5 | 46.8 | 55.0 | 49.7 | 90.6 | 8.5 | 1.3 | 6.8 | 33.5 |
| Daubechies 10 (a) | 21.0 | 32.9 | 41.3 | 43.4 | 30.1 | 56.4 | 68.3 | 52.5 | 90.4 | 18.8 | 2.9 | 8.5 | 38.9 |
| Daubechies 10 (b) | 26.0 | 38.3 | 42.2 | 36.9 | 40.2 | 59.1 | 69.5 | 56.9 | 88.5 | 19.8 | 0.8 | 8.4 | 40.6 |
| Daubechies 10 (c) | 36.5 | 55.5 | 47.7 | 51.5 | 46.8 | 65.7 | 71.6 | 58.1 | 89.6 | 20.4 | 1.6 | 9.1 | 46.2 |
| Daubechies 10 (d) | 18.9 | 32.7 | 39.8 | 33.4 | 29.3 | 53.9 | 60.1 | 50.1 | 91.7 | 8.2 | 1.5 | 7.3 | 35.6 |
| f8a (a) | 20.6 | 29.0 | 39.0 | 50.2 | 28.6 | 51.4 | 68.8 | 48.5 | 88.9 | 23.8 | 1.7 | 12.6 | 38.6 |
| f8a (b) | 16.3 | 29.2 | 30.7 | 36.2 | 27.7 | 54.6 | 65.4 | 44.4 | 89.4 | 24.4 | 1.3 | 12.0 | 36.0 |
| f8a (c) | 25.5 | 37.2 | 36.4 | 36.8 | 33.1 | 56.2 | 67.4 | 43.6 | 92.1 | 17.9 | 1.1 | 8.3 | 38.0 |
| f8a (d) | 9.4 | 22.8 | 28.7 | 27.9 | 22.8 | 40.5 | 54.2 | 44.7 | 90.4 | 11.6 | 0.6 | 8.8 | 30.2 |
| f16b (a) | 17.2 | 28.0 | 36.8 | 45.7 | 24.7 | 50.2 | 65.8 | 48.2 | 89.0 | 22.2 | 1.0 | 10.7 | 36.6 |
| f16b (b) | 15.2 | 28.9 | 31.1 | 32.8 | 26.8 | 51.5 | 62.7 | 43.3 | 90.5 | 24.2 | 0.8 | 11.0 | 34.9 |
| f16b (c) | 25.4 | 34.5 | 37.0 | 32.2 | 30.9 | 54.4 | 63.9 | 45.6 | 89.1 | 17.5 | 1.0 | 7.8 | 36.6 |
| f16b (d) | 10.8 | 22.6 | 29.3 | 23.0 | 19.8 | 39.1 | 51.0 | 45.1 | 91.8 | 10.5 | 0.5 | 7.8 | 29.3 |
| f32d (a) | 16.4 | 27.3 | 35.3 | 44.2 | 23.7 | 48.9 | 64.0 | 46.7 | 89.0 | 19.4 | 0.6 | 10.0 | 35.5 |
| f32d (b) | 16.9 | 29.2 | 30.4 | 32.7 | 26.1 | 51.0 | 60.9 | 42.3 | 91.5 | 27.0 | 0.8 | 10.4 | 34.9 |
| f32d (c) | 16.8 | 32.0 | 35.1 | 29.1 | 29.9 | 52.4 | 66.5 | 45.8 | 92.2 | 20.4 | 0.8 | 8.5 | 35.8 |
| f32d (d) | 11.6 | 20.5 | 28.1 | 22.3 | 22.8 | 41.0 | 50.1 | 43.4 | 92.0 | 10.4 | 0.9 | 6.4 | 29.1 |
| F_2_1_smpl (a) | 15.5 | 26.6 | 38.7 | 44.7 | 25.1 | 51.1 | 65.7 | 48.3 | 90.0 | 22.3 | 2.4 | 9.5 | 36.6 |
| F_2_1_smpl (b) | 16.5 | 28.1 | 35.3 | 34.6 | 26.6 | 52.9 | 62.5 | 47.0 | 92.3 | 22.4 | 3.1 | 9.7 | 35.9 |
| F_2_1_smpl (c) | 26.1 | 42.4 | 43.2 | 37.6 | 34.9 | 58.6 | 65.4 | 51.8 | 87.4 | 19.4 | 6.7 | 9.0 | 40.2 |
| F_2_1_smpl (d) | 17.6 | 24.6 | 31.4 | 26.1 | 27.4 | 41.4 | 53.1 | 49.9 | 91.2 | 11.7 | 3.9 | 6.9 | 32.1 |
| F_2_1_09 (a) | 17.1 | 28.4 | 44.3 | 44.6 | 30.9 | 52.9 | 67.6 | 50.9 | 89.7 | 20.6 | 2.3 | 9.9 | 38.3 |
| F_2_1_09 (b) | 19.8 | 30.5 | 36.5 | 37.3 | 35.6 | 54.3 | 64.8 | 48.2 | 90.5 | 22.7 | 3.2 | 11.4 | 37.9 |
| F_2_1_09 (c) | 26.4 | 43.8 | 53.7 | 37.0 | 37.8 | 61.3 | 67.3 | 54.6 | 87.0 | 18.4 | 7.1 | 11.3 | 42.1 |
| F_2_1_09 (d) | 21.4 | 26.1 | 34.6 | 28.6 | 31.5 | 45.3 | 57.0 | 49.1 | 91.0 | 10.6 | 3.4 | 7.4 | 33.8 |
| Average | 20.8 | 33.5 | 38.7 | 38.0 | 31.9 | 54.0 | 63.8 | 50.3 | 89.2 | 18.8 | 2.4 | 9.8 | 37.6 |

*The averages are summarized in Table 5.*

## 4 CONCLUSIONS

We have seen how various filtering approaches yield different results for different images. No single approach did perform best or very close to the best for *all* images, thus no single approach may be selected as the clear "winner" of this study. In addition to the classification error issue is the issue of computational complexity. If we take the classification error and computational complexity into consideration, the following conclusions may be drawn:

- Much of the focus in the filter *optimization* approaches is on a low feature count, thus many of the optimization schemes yield nice computational characteristics. For multi-textured images this however comes at the cost of an increase in the classification error, in our experiments about 3.5 percent average increase from the best heuristic to the best optimized approach.
- For two-textured images, the conclusion is very clear. The best results were obtained with the filters optimized with respect to $J_U$ or $J_F$. Since none of the other approaches are less computationally complex they should be the clear choice.
- For multi-textured images the most robust and *overall* best performance was achieved with the f16b (d) filter bank—having an overall error rate of only 26 percent. This is however a high-complexity 40-dimensional feature extractor which puts high requirements on the system. If the complexity can be allowed however, this is the system of choice.
- Otherwise the choice is between the f16b (d) critically sampled filter bank and the full rate DCT approach. They both have about 29 percent overall error. The f16b (d) critically sampled filter bank will lead to only 1/8 the number of feature samples compared to the DCT approach, thus classification will be less complex.
- If even these systems are too complex, the next choice will be an optimized approach. The best optimized approach—the Eigenfilter approach—is not an option. Its feature dimensionality, and thus its complexity, is in the range of the f16b (d) full rate filter

bank. This leaves us with the second best optimized approach, the $J_U$-optimized filter bank, or the fourth best approach, the prediction error filters. These approaches have overall error rates of 31.0 percent and 33.3 percent, respectively. Both of these approaches are characterized by a generally low feature dimensionality and extraordinary simple classification schemes. The third-best scheme, the 10-filter optimized Gabor filter bank is outperformed by the DCT approach regarding both complexity and overall error rate.

- Based on the overall impression from the experiments reported here, we would not give a *general* recommendation to use any of the other approaches. We do however recognize that for specific images some of the other techniques give the best results. It might therefore be advisable to *test* the Gabor filter bank, AR, and co-occurrence features in the case that they *may* perform well for the specific dataset at hand. However the co-occurrence and AR schemes do have a significant computational complexity, which should be kept in mind.

In addition to these recommendations, we would also like to make some comments regarding system setups that are commonly presented in the literature.

- The commonly used dyadic decomposition [7], [9], [19], [20], [38] (also known as the wavelet transform decomposition or octave band decomposition) is generally *not* superior to other decompositions. Our results indicate that it is actually inferior to the decomposition of Fig. 6d, supporting similar findings of Chang and Kuo [24]. Furthermore, we see that increasing the depth (i.e., adding more subbands) of the dyadic decomposition do not improve the classification results.
- The very popular Gabor filter [7], [9], [14], [15], [16], [19], [20], [22], [25] is outperformed in most cases. Hence, there is no evidence that the Gabor filter should be preferred.
- Experiments without true separate test and training datasets can not be trusted. We have presented experiments showing that testing with partially the same data that were used in training the classifier leads us to the opposite conclusion than we got when we tested with separate test and training data.

## 5 Directions for Future Research

The experiments reported in this paper have taken weeks, maybe months of computer time on powerful modern workstations. Of course, the software is prototype software, made with an emphasis on short development time and high flexibility. Yet the time consumption is an indication of the problem with several of the approaches—they are very computationally complex. The computer hardware industry is giving us some relief on this issue, but there's a long way yet until the most complex approaches reach what a user would deem as acceptable interactive performance on a standard desktop computer system.

A very useful direction for future research is therefore the development of powerful texture measures that can be extracted and classified with a low computational complexity. The experiments reported here show that the current least complex schemes do not yet have quite the same nice overall classification rate as the best of the more complex ones. In our opinion this is the most important direction for future research in texture analysis.

Finally, we would also like to point the readers attention towards the implicit directions for future research presented in the conclusions of Section 4.

## Acknowledgments

The authors would like to thank Kalle Karu and Jianchang Mao, both formerly at Michigan State University, for providing software for their approaches.

The texture mosaics used in the experiments are available in the electronic version of this paper. The test images, along with software, are also available on an "as is" basis via the web pages of the Signal Processing Group at Stavanger College, http://www.ux.his.no/sigproc/. The software is partially available as MATLAB code and partially as C code based on the HIPS image processing library.

## References

[1] T.R. Reed and J.M.H. du Buf, "A Review of Recent Texture Segmentation and Feature Extraction Techniques," *CVGIP: Image Understanding*, vol. 57, pp. 359–372, May 1993.

[2] M. Tuceryan and A.K. Jain, "Texture Analysis," *Handbook Pattern Recognition and Computer Vision*, C.H. Chen, L.F. Pau, and P.S.P. Wang, eds., ch. 2, pp. 235–276. Singapore: World Scientific, 1993.

[3] K.I. Laws, "Rapid Texture Identification," *Proc. SPIE Conf. Image Processing for Missile Guidance*, pp. 376–380, 1980.

[4] J.S. Weszka, C.R. Dyer, and A. Rosenfeld, "A Comparative Study of Texture Measures for Terrain Classification," *IEEE Trans. Systems, Man, Cybernetics*, vol. 6, pp. 269–285, Apr. 1976.

[5] R.W. Conners and C.A. Harlow, "A Theoretical Comparison of Texture Algorithms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 2, pp. 204–221, May 1980.

[6] J.M.H. du Buf, M. Kardan, and M. Spann, "Texture Feature Performance for Image Segmentation," *Pattern Recognition*, vol. 23, no. 3/4, pp. 291–309, 1990.

[7] J. Strand and T. Taxt, "Local Frequency Features for Texture Classification," *Pattern Recognition*, vol. 27, no. 10, pp. 1,397–1,406, 1994.

[8] M. Pietikäinen, A. Rosenfeld, and L.S. Davis, "Experiments with Texture Classification Using Averages of Local Pattern Matches," *IEEE Trans. Systems, Man, Cybernetics*, vol. 13, no. 3, pp. 421–426, 1983.

[9] D. Clausi and M. Jernigan, "Towards a Novel Approach for Segmentation of SAR Sea Ice Imagery," *Proc. 26th Symp. Remote Sensing of Environment and 18th Ann. Symp. Canadian Remote Sensing Society*, Mar. 1995.

[10] T. Ojala, M. Pietikäinen, and D. Harwood, "A Comparative Study of Texture Measures with Classification Based on Feature Distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[11] M. Unser, "Texture Classification and Segmentation Using Wavelet Frames," *IEEE Trans. Image Processing*, vol. 4, pp. 1,549–1,560, Nov. 1995.

[12] J.M. Coggins and A.K. Jain, "A Spatial Filtering Approach to Texture Analysis," *Pattern Recognition Letters*, vol. 3, no. 3, pp. 195–203, 1985.

[13] F. Ade, "Characterization of Texture by 'Eigenfilter,'" *Signal Processing*, vol. 5, no. 5, pp. 451–457, 1983.

[14] A.C. Bovik, M. Clark, and W.S. Geisler, "Multichannel Texture Analysis Using Localized Spatial Filters," *IEEE Trans. Pattern Analysis and Machine Intelligence.*, vol. 12, pp. 55–73, Jan. 1990.

[15] T.P. Weldon and W.E. Higgins, "Design of Multiple Gabor Filters for Texture Segmentation," *Proc. Int'l Conf. Acoustic Speech, Signal Proc.*, Atlanta, Ga., pp. 2,243–2,246, May 1996.

[16] T.P. Weldon and W.E. Higgins, “Integrated Approach to Texture Segmentation Using Multiple Gabor Filters,” *Proc. Int’l Conf. Image Processing*, Lausanne, Switzerland, pp. 955–958, Sept. 1996.

[17] T. Randen and J.H. Husøy, “Multichannel Filtering for Image Texture Segmentation,” *Optical Eng.*, vol. 33, pp. 2,617–2,625, Aug. 1994.

[18] T. Randen and J.H. Husøy, “Texture Segmentation Using Filters with Optimized Energy Separation,” *IEEE Trans. Image Processing*, vol. 8, no. 4, Apr. 1999.

[19] F. Farrokhnia, “Multi-Channel Filtering Techniques for Texture Segmentation and Surface Quality Inspection,” PhD thesis, Michigan State Univ., 1990.

[20] A.K. Jain and F. Farrokhnia, “Unsupervised Texture Segmentation Using Gabor Filters,” *Pattern Recognition*, vol. 24, no. 12, pp. 1,167–1,186, 1991.

[21] A. Laine and J. Fan, “Texture Classification by Wavelet Packet Signatures,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, pp. 1,186–1,190, Nov. 1993.

[22] P.P. Ohanian and R.C. Dubes, “Performance Evaluation for Four Classes of Textural Features,” *Pattern Recognition*, vol. 25, no. 8, pp. 819–833, 1992.

[23] M. Unser, “Local Linear Transforms for Texture Measurements,” *Signal Processing*, vol. 11, no. 1, pp. 61–79, 1986.

[24] T. Chang and C.-C.J. Kuo, “Texture Analysis and Classification with Tree-Structured Wavelet Transform,” *IEEE Trans. Image Processing*, vol. 2, no. 4, pp. 429–441, 1993.

[25] A. Teuner, O. Pichler, and B.J. Hosticka, “Unsupervised Texture Segmentation of Images Using Tuned Matched Gabor Filters,” *IEEE Trans. Image Processing*, vol. 4, pp. 863–870, June 1995.

[26] N. Saito and R.R. Coifman, “Local Discriminant Bases and Their Applications,” *J. Mathematical Imaging and Vision*, vol. 5, no. 4, pp. 337–358, 1995.

[27] M. Unser and M. Eden, “Nonlinear Operators for Improving Texture Segmentation Based on Features Extracted by Spatial Filtering,” *IEEE Trans. Systems, Man, Cybernetics*, vol. 20, pp. 804–815, 1990.

[28] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[29] K. Fukunaga, *Statistical Pattern Recognition*, 2nd ed. San Diego, Calif.: Academic Press, 1990.

[30] R. Schalkoff, *Pattern Recognition: Statistical, Structural and Neural Approaches*. New York: Wiley, 1992.

[31] A. Mahalanobis and H. Singh, “Application of Correlation Filters for Texture Recognition,” *Applied Optics*, vol. 33, no. 11, pp. 2,173–2,179, 1994.

[32] T. Randen and J.H. Husøy, “Texture Segmentation with Optimal Linear Prediction Error Filters,” *Piksel’n*, vol. 11, pp. 25–28, Sept. 1994. Also available at http://www.ux.his.no/~tranden/.

[33] T. Randen, “Filter and Filter Bank Design for Image Texture Recognition,” PhD thesis, Norwegian Univ. of Science and Technology, Trondheim, Norway, Oct. 1997. Also available at http://www.ux.his.no/~tranden/.

[34] J. Mao and A.K. Jain, “Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models,” *Pattern Recognition*, vol. 25, no. 2, pp. 173–188, 1992.

[35] T. Kohonen, “The Self-Organizing Map,” *Proc. IEEE*, vol. 78, pp. 1,464–1,480, Sept. 1990.

[36] R. Haralick, K. Shanmugam, and I. Dinstein, “Textural Features for Image Classification,” *IEEE Trans. Systems, Man, Cybernetics*, vol. 3, pp. 610–621, Nov. 1973.

[37] A.N. Akansu and R.A. Haddad, *Multiresolution Signal Decomposition*. San Diego, Calif.: Academic Press, 1992.

[38] S.G. Mallat, “A Theory for Multiresolution Signal Decomposition: The Wavelet Representation,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674–693, July 1989.

[39] I. Daubechies, *Wavelets*. Philadelphia, Pa.: S.I.A.M, 1992.

[40] T.A. Ramstad, S.O. Aase, and J.H. Husøy, *Subband Compression of Images—Principles and Examples*. North Holland: ELSEVIER Science Publishers BV, 1995.

[41] I. Ng, T. Tan, and J. Kittler, “On Local Linear Transform and Gabor Filter Representation of Texture,” *Proc. Int’l Conf. Pattern Recognition*, pp. 627–631. Int’l Assoc. for Pattern Recognition, 1992.

[42] J.D. Johnston, “A Filter Family Designed for Use in Quadrature Mirror Filter Banks,” *Proc. Int’l Conf. Acoustic Speech, Signal Processing*, Denver, Colo., pp. 291–294, 1980.

[43] J.H. Husøy, “Low Complexity Subband Coding of Still Images and Video,” *Optical Eng.*, vol. 30, pp. 904–911, July 1991.

[44] D.F. Dunn and W.E. Higgins, “Optimal Gabor Filters for Texture Segmentation,” *IEEE Trans. Image Processing*, vol. 4, pp. 947–964, July 1995.

[45] A.K. Jain and K. Karu, “Learning Texture Discrimination Masks,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, pp. 195–205, Feb. 1996.

[46] A.C. Bovik, “Analysis of Multichannel Narrow-Band Filters for Image Texture Segmentation,” *IEEE Trans. Signal Processing*, vol. 39, pp. 2,025–2,043, Sept. 1991.

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

[48] MIT Vision and Modelling Group, http://www.media.mit.edu/vismod/, 1998.

[49] MeasTex Image Texture Database, http://www.cssip.elec.uq.edu.au/~guy/meastex/meastex.html, 1998.

[50] D.F. Dunn, W.E. Higgins, and J. Wakeley, “Texture Segmentation Using 2-D Gabor Elementary Functions,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, pp. 130–149, Feb. 1994.

**Trygve Randen** received his BEng and MSc degrees in 1991 and 1993 from Rogaland University Center, Norway, and his DrIng in 1997 from the Norwegian University of Science and Technology. He is a research scientist at Schlumberger Geco-Prakla, Stavanger, Norway. His main research interest is image and multidimensional signal processing for the interpretation and processing of geophysical data. He is a member of the IEEE, IAPR, SEG, EAGE, the Norwegian Society for Image Processing and Pattern Recognition (NOBIM), and the Norwegian Society for Signal Processing (NORSIG), where he is currently vice chairman.

**John Håkon Husøy** received his MSc and DrIng degrees in electrical engineering in 1981 and 1991, respectively, from the Norwegian Institute of Technology (NTH), Trondheim, Norway. From 1982 to 1986, he worked for the Canadian Department of Transportation, Ottawa, Ontario, and subsequently for Northern Telecom, Ottawa. The following two years were spent at the Chr. Michelsen Institute, Bergen, Norway. From 1988 to 1992, he was with the Department of Electrical and Computer Engineering at NTH. In 1992, Dr. Husøy was appointed professor of Electrical and Computer Engineering at Rogaland University Center (now Stavanger College). His research interests include image compression, digital filtering, bioelectrical signal processing, adaptive algorithms and image analysis. He is a member of the IEEE, EURASIP, IAPR, NOBIM, and NORSIG.