Chapter 13

A Galaxy of Texture Features

Xianghua Xie and Majid Mirmehdi

Department of Computer Science, University of Bristol
Bristol BS8 1UB, England
E-mail: {xie,majid}@cs.bris.ac.uk

The aim of this chapter is to give experienced and new practitioners in image analysis and computer vision an overview and a quick reference to the "galaxy" of features that exist in the field of texture analysis. Clearly, given the limited space, only a corner of this vast galaxy is covered here! Firstly, a brief taxonomy of texture analysis approaches is outlined. Then, a list of widely used texture features is presented in alphabetical order. Finally, a brief comparison of texture features and feature extraction methods based on several literature surveys is given.

13.1. Introduction

The aim of this chapter is to give the reader a comprehensive overview of texture features. This area is so diversive that it is impossible to cover it fully in this limited space. Thus only a list of widely used texture features are presented here. However, before that, we will first look at how these features can be used in texture analysis. With reference to several survey papers,[1–6] we categorise these texture features into four families: statistical features, structural features, signal processing based features, and model based features. It is worth noting that this categorisation is not a crisp classification. There are techniques that combine features from different categories for texture analysis, e.g. Ref. 7 applies statistical co-occurrence measurements on wavelet transformed detail images. At the end of this chapter, a very brief comparison of texture features and feature extraction methods will be given based on literature survey.

13.1.1. *Statistical features*

Statistical texture features measure the spatial distribution of pixel values. They are well rooted in the computer vision literature and have been extensively applied to various tasks. Texture features are computed based on the statistical distribution of image intensities at specified relative pixel positions. A large number of these features have been proposed, ranging from first order statistics to higher order statistics depending on the number of pixels for each observation.

The image histogram is a first order statistical feature that is not only computationaly simple, but also rotational and translation invariant; it is thus commonly used in various vision applications, e.g. image indexing and retrieval. Second order statistics examine the relationship between a pair of pixels across the image domain, for example through autocorrelation. One of the most well-known second order statistical feature for texture analysis is the co-occurrence matrix.[8] Several statistics, such as energy and entropy, can be derived from the co-occurrence matrix to characterise textures. Higher order statistical features explore pixel relationships beyond pixel pairs and they are generally less sensitive to image noise.[9,10] Gray level run length[11] and local binary pattern (LBP)[12] can also be considered higher order statistical features.

13.1.2. *Structural features*

From the structural point of view, texture is characterised by *texture primitives* or texture elements, and the spatial arrangement of these primitives.[4] Thus, the primary goals of structural approaches are firstly to extract texture primitives, and secondly to model or generalise the spatial placement rules. The texture primitive can be as simple as individual pixels, a region with uniform graylevels, or line segments. The placement rules can be obtained through modelling geometric relationships between primitives or learning their statistical properties.

A few example works are as follows. Zucker[13] proposed that natural textures can be treated as ideal patterns that have undergone certain transformations. The placement rule is defined by a graph that is isomorphic to a regular or semi-regular tessellation which is transformable to generate variant natural textures. Fu[14] considered a texture as a string of a language defined by a tree grammar which defines the spatial placement rules, and its terminal symbols are the texture primitives that can be individual pixels, connected or isolated. Marr[15] proposed a symbolic description, the primal sketch, to represent spatial texture features, such as edges, blobs, and bars. In Ref. 16, Julesz introduced the concept of textons as fundamental image structures, such as elongated blobs, bars, crosses, and terminators (more details later in this chapter). The textons were considered as atoms of pre-attentive human visual perception. The idea of describing texture using local image patches and placement rules has also been practiced in texture synthesis, e.g. Ref. 17.

13.1.3. *Signal processing based features*

Most signal processing based features are commonly extracted by applying filter banks to the image and computing the energy of the filter responses. These features can be derived from the spatial domain, the frequency domain, and the joint spatial/spatial-frequency domain.

In the spatial domain, the images are usually filtered by gradient filters to extract edges, lines, isolated dots, etc. Sobel, Robert, Laplacian, Laws filters have been routinely used as a precursor to measuring edge density. In Ref. 18, Malik and Perona used a bank of differences of offset Gaussian function filters to model pre-attentive texture perception in human vision. Ade[19] proposed eigenfilters, a set of masks obtained from the Karhunen-Lóeve (KL) transform[20] of local image patches, for texture representation.

Many other features are derived by applying filtering in the frequency domain, particularly when the associated kernel in the spatial domain is difficult to obtain. The image is transformed into the Fourier domain, multiplied with the filter function and then re-transformed into the spatial domain saving on the spatial convolution operation. Ring and wedge filters are some of the most commonly used frequency domain filters, e.g. Ref. 21. D'Astous and Jernigan[22] used peak features, such as strength and area, and power distribution features, such as power spectrum eigenvalues and circularity, to discriminate textures.

The Fourier transform has relatively poor spatial resolution, as Fourier coefficients depend on the entire image. The classical way of introducing spatial dependency into Fourier analysis is through the windowed Fourier transform. If the window function is Gaussian, the windowed Fourier transform becomes the well-known Gabor transform. Psychophysiological findings of multi-channel, frequency and orientation analysis in the human vision system have strongly motivated the use of Gabor analysis, along with other multiscale techniques. Turner[23] and Clark and Bovik[24] first proposed the use of Gabor filters in texture analysis.

Carrying similar properties to the Gabor transform, wavelet transform representations have also been widely used for texture analysis, e.g. Refs. 25, 7, 26 and 27. Wavelet analysis uses approximating functions that are localised in both spatial and spatial-frequency domain. The input signal is considered as the weighted sum of overlapping wavelet functions, scaled and shifted. These functions are generated from a basic wavelet (or mother wavelet) by dilation and translation.

Dyadic transformation is one of the most commonly used, however, its frequency and orientation selection are rather coarse. Wavelet packet decomposition,[28] as a generalisation of the discrete wavelet transform, is one of the extensions to improve the selectivity where at each stage of the transform, the signal is split into low-pass and high-pass orthogonal components. The low-pass is an approximation of the input signal, while the high-pass contains the missing signals from the approximation. Finer frequency selectivity can be further obtained by dropping the constraints of orthogonal decomposition.

13.1.4. *Model based features*

Model based methods include, among many others, fractal models,[29] autoregressive models,[30,31] random field models,[32] and the epitome model.[33] They generally use stochastic and generative models to represent images, with the estimated model parameters as texture features for texture analysis.

The fractal model is based on the observation of self-similarity and has been found useful in modelling natural textures. Fractal dimension and lacunarity are the two most popular fractal features. However, this model is generally considered not suitable for representing local image structures. Random field models, including autoregressive models, assume that local information is sufficient to achieve a good global image representation. One of the major challenges is to efficiently estimate the model parameters. The establishment between Markov random field and Gibbs distribution, which takes simpler form, provided tractable statistical analysis using random field theories. Recently, Jojic *et al.* proposed a generative model called *epitome* which is a miniature of the original image and extracts its essential textural and shape characteristics. This model also relies on the local neighbourhood.

13.2. Texture Features and Feature Extraction Methods

In this section, a number of of commonly used texture features are presented in alphabetical order. Additionally, we also outline several feature extraction methods. In what follows $\mathbf{I}$ denotes a $w \times h$ image in which individual pixels are addressed by $\mathbf{I}(x, y)$, however, when convenient, other appropriate terminology may also be used. Each feature or feature extraction method has one or more symboles after it signifying which categories it can be associated with, where $\spadesuit$ denotes statistical approach, $\heartsuit$ denotes structural approach, $\diamondsuit$ represents signal processing approach, and $\clubsuit$ represents model based approach.

(1) **Autocorrelation** (♤ − −−)
The autocorrelation feature is derived based on the observation that some textures are repetitive in nature, such as textiles. It measures the correlation between the image itself and the image translated with a displacement vector, $\mathbf{d} = (dx, dy)$ as:

$$\rho(\mathbf{d}) = \frac{\sum_{x=0}^{w}\sum_{y=0}^{h} \mathbf{I}(x,y)\mathbf{I}(x+dx, y+dy)}{\sum_{x=0}^{w}\sum_{y=0}^{h} \mathbf{I}^2(x,y)}. \qquad (13.1)$$

Textures with strong regularity will exhibit peaks and valleys in the autocorrelation measure. This second order statistic is clearly sensitive to noise interference. Higher order statistics, e.g. Refs. 34 and 10, have been investigated, for example, Huang and Chan[10] used fourth-order cumulants to extract harmonic peaks and demonstrated the method's ability to localise defects in textile images.

(2) **Autoregressive** model (− − −♧)
The autoregressive model is usually considered as an instance of the Markov Random Field model. Similar to autocorrelation, autoregressive models also exploit the linear dependency among image pixels. The basic autoregressive model for texture analysis can be formulated as:[30]

$$g(\mathbf{s}) = \mu + \sum_{\mathbf{d}\in\Omega} \theta(\mathbf{d}) g(\mathbf{s}+\mathbf{d}) + \varepsilon(\mathbf{s}), \qquad (13.2)$$

where $g(\mathbf{s})$ is the gray level value of a pixel at site $\mathbf{s}$ in image $\mathbf{I}$, $\mathbf{d}$ is the displacement vector, θ is a set of model parameters, μ is the bias that depends on the mean intensity of the image, $\varepsilon(\mathbf{s})$ is the model error term, and Ω is the set of neighbouring pixels at site $\mathbf{s}$. A commonly used second order neighbourhood is a pixel's 8-neighbourhood. These model parameters can be considered as a characterisation of a texture, thus, can be used as texture features.
Autoregressive models have been applied to texture synthesis,[35] texture segmentation,[1] and texture classification.[30] Selection of the neighbourhood size is one of the main design issues in autoregressive models. Multiresolution methods have been used to alleviate the associated difficulties, such as in Ref. 30.

(3) **Co-occurrence matrices** (♤ − −−)
Spatial graylevel co-occurrence matrices (GLCM)[8] are one of the most well-known and widely used texture features. These second order statistics are accumulated into a set of 2D matrices, $\mathbf{P}(r, s|\mathbf{d})$, each of which measures the spatial dependency of two graylevels, r and s, given a displacement vector $\mathbf{d} = (d, \theta) = (dx, dy)$. The number of occurrences (frequencies) of r and

s, separated by distance $\mathbf{d}$, contributes the (r, s)th entry in the co-occurrence matrix $\mathbf{P}(r, s|\mathbf{d})$. A co-occurrence matrix is given as:

$$\mathbf{P}(r, s|\mathbf{d}) = ||\{((x_1, y_1), (x_2, y_2)) : \mathbf{I}(x_1, y_1) = r, \mathbf{I}(x_2, y_2) = s\}|| \tag{13.3}$$

where $(x_1, y_1), (x_2, y_2) \in w \times h$, $(x_2, y_2) = (x_1 \pm dx, y_1 \pm dy)$ and $||.||$ is the cardinality of a set. Texture features, such as energy, entropy, contrast, homogeneity, and correlation, are then derived from the co-occurrence matrix. Example successful applications on texture analysis using co-occurrence features can be found in Refs. 8, 36 and 37.

Co-occurrence matrix features can suffer from a number of shortcomings. It appears there is no generally accepted solution for optimising $\mathbf{d}$.[6,38] The number of graylevels is usually reduced in order to keep the size of the co-occurrence matrix manageable. It is also important to ensure the number of entries of each matrix is adequate to be statistically reliable. For a given displacement vector, a large number of features can be computed, which implies dedicated feature selection procedures.

(4) **Difference of Gaussians filter** $(- - \diamond -)$

This is the one of the most common filtering techniques to extract texture features. Smoothing an image using different Gaussian kernels followed by computing their difference is used to highlight image features, such as edges at different scales. As Gaussian smoothing is low pass filtering, difference of Gaussians is thus effectively band pass filtering. Its kernel (see Fig. 13.1) can be simply defined as:

$$DoG = G_{\sigma_1} - G_{\sigma_2}, \tag{13.4}$$

where G_{σ_1} and G_{σ_2} are two different Gaussian kernels. Difference of Gaussians is often used as an approximation of Laplacian of Gaussian. By varying σ_1 and σ_2, we can extract textural features at particular spatial frequencies. Note this filter is not orientation selective. Example applications can be found in the scale-space primal sketch[39] and SIFT feature selection.[40]

(5) **Difference of offset Gaussians filters** $(- - \diamond -)$

This is another simple filtering technique which provides useful texture features, such as edge orientation and strength. Similar to difference of Gaussians filters, the filter kernel is obtained by subtracting two Gaussian functions. However, the centre of these two Gaussian functions are displaced by a vector $\mathbf{d} = (dx, dy)$:

$$DooG_\sigma(x, y) = G_\sigma(x, y) - G_\sigma(x + dx, y + dy). \tag{13.5}$$

Figure 13.2 shows an example difference of offset Gaussians filter kernel in

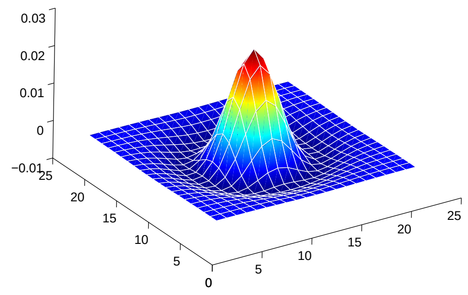

Fig. 13.1. 3D visualisation of a difference of Gaussians filter kernel.

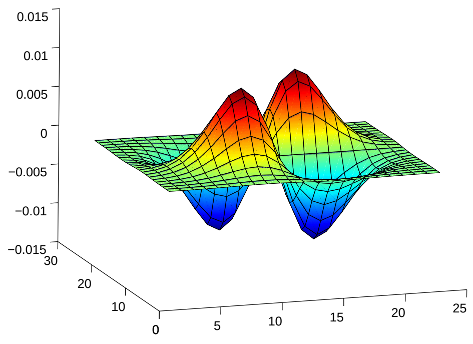

Fig. 13.2. 3D visualisation of a difference of offset Gaussian filters kernel.

3D. An example application of these filters to texture analysis can be found in Ref. 18.

(6) **Derivative of Gaussian filters** (− − ◇−)
Edge orientation or texture directionality is one of the most important cues to understand textures. Derivative filters, particularly derivative of Gaussian filters, are commonly applied to highlight texture features at different orientations. By varying their kernel bandwidth, these filters can also selectively highlight texture features at different scales. Given a Gaussian function

$G_\sigma(x,y)$, its first derivatives in x and y directions are:

$$D_x(x,y) = -\frac{x}{\sigma^2}G_\sigma(x,y), \;\; D_y(x,y) = -\frac{y}{\sigma^2}G_\sigma(x,y). \tag{13.6}$$

Convolving an image with Gaussian derivative kernels is equivalent to smoothing the image using a Gaussian kernel and then computing its derivatives. These oriented filters have been widely used in texture analysis, for example in Refs. 41 and 42. Also see "Steerable filter".

(7) **Eigenfilter** (− − ◇−)

Most filters used for texture analysis are non-adaptive, i.e. the filters are predefined and often not directly associated with the textures. However, eigenfilter is an exception. The eigenfilter was first introduced to texture analysis by Ade.[19] Eigenfilters are considered adaptive as they are data dependent and they can highlight the dominant features of the textures. The filters are usually generated through Karhunen-Loève transform. In Ref. 19, the eigenfilters are extracted from autocorrelation functions. Let $\mathbf{I}^{(x,y)}$ be the original image without any displacement, and $\mathbf{I}^{(x+n,y)}$ be the shifted image along the x direction by n pixel(s). For example, if n takes a maximum value of 2, the eigenvectors and eigenvalues are computed from this 9×9 autocorrelation matrix:

$$\begin{bmatrix} E[\mathbf{I}^{(x,y)}\mathbf{I}^{(x,y)}] & \dots & E[\mathbf{I}^{(x,y+2)}\mathbf{I}^{(x,y)}] & \dots & E[\mathbf{I}^{(x+2,y+2)}\mathbf{I}^{(x,y)}] \\ \vdots & & \vdots & & \vdots \\ E[\mathbf{I}^{(x,y+2)}\mathbf{I}^{(x,y)}] & \dots & E[\mathbf{I}^{(x,y+2)}\mathbf{I}^{(x,y+2)}] & \dots & E[\mathbf{I}^{(x,y+2)}\mathbf{I}^{(x+2,y+2)}] \\ \vdots & & \vdots & & \vdots \\ E[\mathbf{I}^{(x+2,y+2)}\mathbf{I}^{(x,y)}] & \dots & E[\mathbf{I}^{(x+2,y+2)}\mathbf{I}^{(x,y+2)}] & \dots & E[\mathbf{I}^{(x+2,y+2)}\mathbf{I}^{(x+2,y+2)}] \end{bmatrix}, \tag{13.7}$$

where $E[.]$ denotes expectation. The 9×1 eigenvectors are rearranged in the spatial domain resulting in 3×3 eigenfilters. The number of eigenfilters selected can be determined by thresholding the sum of eigenvalues. The filtered images, usually referred to as basis images, can be used to reconstruct the original image. Due to their orthogonality, they are considered as an optimised representation of the image. Example applications can be found in Refs. 19 and 43.

(8) **Eigenregion** (♤♡ − −)

Eigenregions are geometrical features that encompass area, location, and shape properties of an image.[44] They are generated based on image prior-segmentation and principal component analysis. The images are firstly segmented and the regions within are downsampled to much smaller patches, such as 5×5. Then principal components are obtained from these simplified image regions and used for image classification. A similar approach has been presented in Ref. 45 for image segmentation.

(9) **Epitome model** (– – –♧)
The epitome, as described in Ref. 33, is a small, condensed representation of a given image containing its primitive shapes and textural elements. The mapping from the epitome to its original pixels is hidden, and several images may share the same epitome by varying the hidden mapping. In this model, raw pixel values are used to characterise textural and colour properties, instead of popular filtering responses. The epitome is derived using a generative model. It is assumed that image patches from the original (large) images are produced from the epitome by copying pixel values from it with added Gaussian noise. Thus, as a learning process various sizes of patches from the image are taken and are forced into the epitome, a much smaller image, by examining the best possible match. The epitome is then updated accordingly when new image patches are sampled. This process iteratively continues until the epitome is stabilised. Figure 13.3 shows an example image and two epitomes at different sizes. We can see that the epitomes are relatively compact representations of the image.

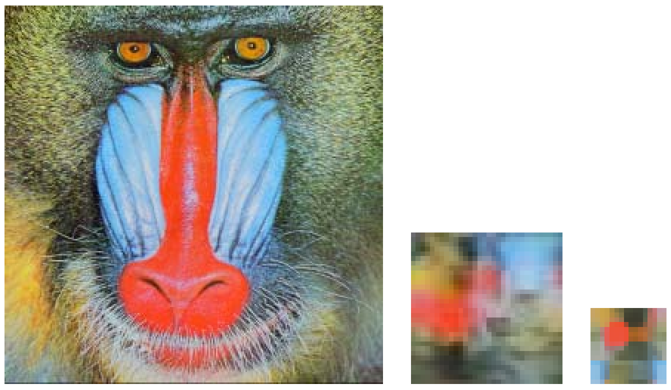

Fig. 13.3. Epitome - from left: Original colour image, its 32×32 epitome, and its 16×16 epitome (generated with the software provided by the authors in Ref. 33).

The authors of the epitome model have demonstrated its ability in texture segmentation, image denoising, and image inpainting.[33] Stauffer[46] also used epitomes to measure the similarity between pixels and patches to perform image segmentation. Cheung *et al.*[47] further extended the epitome model for video analysis.

(10) **Fractal model** (– – –♧)

Fractals, initially proposed by Mandelbrot,[29] are geometric primitives that are self-similar and irregular in nature. Fragments of a fractal object are exact or statistical copies of the whole object and they can match the whole by stretching and shifting.

Fractal dimension is one of the most important features in the fractal model as a measure of complexity or irregularity. Several methods have been developed to estimate the fractal dimension. Pentland[48] used the Fourier power spectral density to estimate the fractal dimension for image segmentation. The image intensity is modelled as 3D fractal Brownian motion surfaces. Gangepain and Roques-Carmes[49] proposed the box-counting method which was later improved by Voss[50] and Keller *et al.*[51] Super and Bovik[52] proposed the use of Gabor filters to estimate the fractal dimension in textured images. Lacunarity is another important measurement in fractal models. It measures the structural variation or inhomogeneity and can be calculated using the gliding-box algorithm.[53]

(11) **Gabor filters** (− − ◇−)

Gabor filters are used to model the spatial summation properties of simple cells in the visual cortex and have been adapted and popularly used in texture analysis, for example see Refs. 23,24,54 and 55. They have been long considered as one of the most effective filtering techniques to extract useful texture features at different orientations and scales. Gabor filters can be categorised into two components: a real part as the symmetric component and an imaginary part as the asymmetric component. The 2D Gabor function can be mathematically formulated as:

$$G(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)\right] \exp(2\pi j u_0 x), \tag{13.8}$$

where σ_x and σ_y define the Gaussian envelope along the x and y directions respectively, u_0 denotes the radial frequency of the Gabor function, and $j = \sqrt{-1}$. Figure 13.4 shows the frequency response of the dyadic Gabor filter bank with the centre frequencies $\{2^{-\frac{11}{2}}, 2^{-\frac{9}{2}}, 2^{-\frac{7}{2}}, 2^{-\frac{5}{2}}, 2^{-\frac{3}{2}}\}$, and orientations $\{0°, 45°, 90°, 135°\}$.[56]

(12) **Gaussian Markov random field (GMRF)** – see “Random field models”.

(13) **Gaussian pyramid features** (− − ◇−)

Extracting features in multiscale is an efficient way of analysing image texture. The Gaussian pyramid is one of the simplest multiscale transforms. Let us denote $\mathbf{I}^{(n)}$ as the n^{th} level image of the pyramid, l as the total number of levels, and $S^{\downarrow}$ as the down-sampling operator. We then have

$$\mathbf{I}^{(n+1)} = S^{\downarrow} G_{\sigma}(\mathbf{I}^{(n)}), \qquad \forall n, n = 1, 2, ..., l-1, \tag{13.9}$$

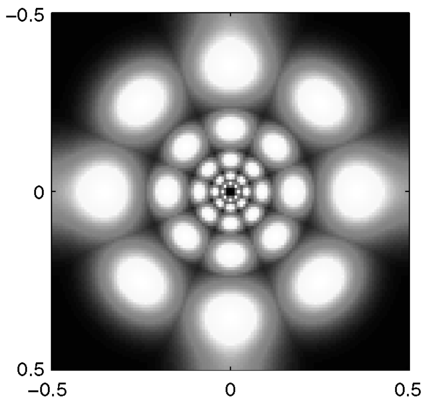

Fig. 13.4. The frequency response of the dyadic bank of Gabor filters. The maximum amplitude response over all filters is plotted. Each filter is represented by one centre-symmetric pair of lobes in the illustration. The axes are in normalised spatial frequencies (image reproduced with permission from Ref. 56).

where G_σ denotes the Gaussian convolution. The finest scale layer is the original image, $\mathbf{I}^{(1)} = \mathbf{I}$. As each level is a low pass filtered version of the previous level, the low frequency information is repeatedly represented in Gaussian pyramid.

(14) **Gray level difference matrix** (♤ – ––)
Gray level difference statistics are considered a subset of the co-occurrence matrix.[57] They are based on the distribution of pixel pairs separated by $\mathbf{d} = (dx, dy)$ having gray level difference k:

$$\mathbf{P}(k|\mathbf{d}) = \|\{((x_1, y_1), (x_2, y_2)) : |\mathbf{I}(x_1, y_1) - \mathbf{I}(x_2, y_2)| = k\}\|, \tag{13.10}$$

where $(x_2, y_2) = (x_1 \pm dx, y_1 \pm dy)$. Various properties then can be extracted from this matrix, such as angular second moment, contrast, entropy, and mean, for texture analysis purposes.

(15) **Gibbs random field** – see “’Random field models’

(16) **Histogram features** (♤ – ––)
Commonly used histogram features include range, mean, geometric mean, harmonic mean, standard deviation, variance, and median. Despite their simplicity, histogram techniques have proved their worth as a low cost, low level approach in various applications, such as Ref. 58. They are invariant to translation and rotation, and insensitive to the exact spatial distribution of the colour pixels. Table 13.1 lists some similarity measurements of two distributions, where r_i and s_i are the number of events in bin i for the first and second

Table 13.1. Some histogram similarity measurements.

Measurement	Formula
L_1 norm	$L_1 = \sum_{i=1}^{n} \lvert r_i - s_i \rvert$
L_2 norm	$L_2 = \sqrt{\sum_{i=1}^{n} (r_i - s_i)^2}$
Mallows or EMD distance	$M_p = \left(\frac{1}{n} \sum_{i=1}^{n} \lvert r_{(i)} - s_{(i)} \rvert^p \right)^{1/p}$
Bhattacharyya distance	$B = -\ln \sum_{i=1}^{n} \sqrt{r_i s_i}$
Matusita distance	$M = \sqrt{\sum_{i=1}^{n} (\sqrt{r_i} - \sqrt{s_i})^2}$
Divergence	$D = \sum_{i=1}^{n} \left((r_i - s_i) \ln \frac{r_i}{s_i} \right)$
Histogram intersection	$H = \frac{\sum_{i=1}^{n} \min(r_i, s_i)}{\sum_{i=1}^{n} r_i}$
Chi-square	$\chi^2 = \sum_{i=1}^{n} \frac{(r_i - s_i)^2}{r_i + s_i}$
Normalised correlation coefficient	$r = \frac{\sum_{i=1}^{n} (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\sum_{i=1}^{n} (r_i - \bar{r})^2} \sqrt{\sum_{i=1}^{n} (s_i - \bar{s})^2}}$

datasets, respectively, $\bar{r}$ and $\bar{s}$ are the mean values, n is the total number of bins, and $r_{(i)}$ and $s_{(i)}$ denote the sorted (ascending order) indices. Note EMD is the Earth Mover's Distance.

(17) **Laplacian of Gaussian** $(- - \diamond -)$

Laplacian of Gaussian is another simple but useful multiscale image transformation. The transformed data contains basic but also useful texture features. The 2D Laplacian of Gaussian with zero mean and Gaussian standard deviation σ is defined as:

$$LoG_\sigma(x, y) = -\frac{1}{\pi\sigma^4} \left[1 - \frac{x^2 + y^2}{2\sigma^2} \right] e^{-\frac{x^2+y^2}{2\sigma^2}} . \tag{13.11}$$

Figure 13.5 plots a 3D visualisation of such a function. Laplacian of Gaussian calculates the second spatial derivative of an image, and is closely related to the difference of Gaussians function. It is often used in low level feature extraction, e.g. Ref. 59.

(18) **Laplacian pyramid** $(- - \diamond -)$

Decomposing an image so that redundant information is minimised and characteristic features are thus preserved and highlighted is a common way of analysing textures. The Laplacian pyramid was applied by Burt and Adelson[60] to image compression to remove redundancy. Compared to the Gaussian pyramid, the Laplacian pyramid is a much more compact representation. Each level of a Laplacian pyramid contains the difference between a low pass filtered version and an upsampled "predication" from coarser level. It can be formulated as:

$$\mathbf{I}_L^{(n)} = \mathbf{I}_G^{(n)} - S \uparrow \mathbf{I}_G^{(n+1)}, \tag{13.12}$$

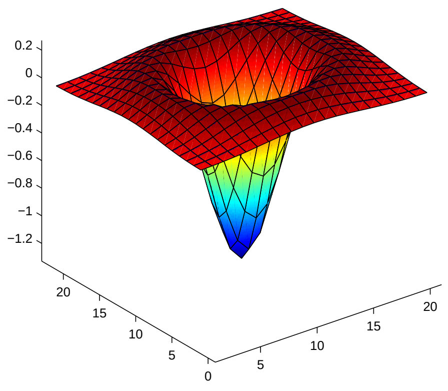

Fig. 13.5. A 3D visualisation of a Laplacian of Gaussian filter kernel.

where $\mathbf{I}_L^{(n)}$ denotes the nth level in a Laplacian pyramid, $\mathbf{I}_G^{(n)}$ denotes the nth level in a Gaussian pyramid of the same image, and $S\uparrow$ represents upsampling using nearest neighbours.

(19) **Laws operators** ($\spadesuit - \diamondsuit -$)
These texture energy measures were developed by Laws[61] and are considered as one of the first filtering approaches to texture analysis. The Laws texture energy measures are computed first by applying a bank of separable filters, followed by a nonlinear local window based transform. The most commonly used five element kernels are as follows:

$$\begin{array}{lclrrrrrl} \mathrm{L5} & = & [& 1 & 4 & 6 & 4 & 1 &] \\ \mathrm{E5} & = & [& -1 & -2 & 0 & 2 & 1 &] \\ \mathrm{S5} & = & [& -1 & 0 & 2 & 0 & -1 &] \\ \mathrm{W5} & = & [& -1 & 2 & 0 & -2 & 1 &] \\ \mathrm{R5} & = & [& 1 & -4 & 6 & -4 & 1 &], \end{array} \qquad (13.13)$$

where the initial letters denote Level, Edge, Spot, Wave, and Ripple, respectively. From these five 1D operators, a total of 25 2D Laws operators can be generated by convolving a vertical 1D kernel with a horizontal 1D kernel, for example convolving the vertical L5 with a horizontal W5.

(20) **Local binary patterns (LBP)** ($\spadesuit\heartsuit - -$)
The LBP operator was first introduced by Ojala *et al.*[12] as a shift invariant complementary measure for local image contrast. It uses the graylevel of the centre pixel of a sliding window as a threshold for surrounding neighbour-

hood pixels. Its value is given as a weighted sum of thresholded neighbouring pixels.

$$\mathcal{L}_{P,R} = \sum_{p=0}^{P-1} sign(g_p - g_c)2^p, \tag{13.14}$$

where g_c and g_p are the graylevels of centre pixel and neighbourhood pixels respectively, P is the total number of neighbourhood pixels, R denotes the radius, and $sign(.)$ is a sign function such that

$$sign(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ otherwise.} \end{cases} \tag{13.15}$$

Figure 13.6 shows an eight-neighbours LBP calculation. A simple local contrast measurement, $C_{P,R}$, is derived from the difference between the average gray levels of pixels brighter than centre pixel and those darker then centre pixel, i.e. $C_{P,R} = \sum_{p=0}^{P-1}(sign(g_p - g_c)g_p/M - sign(g_c - g_p)g_p/(P - M))$ where M denote the number of pixels that brighter than the centre pixel. It is calculated as a complement to the LBP value in order to characterise local spatial relationships, together called LBP/C.[12] Two-dimensional distributions of the LBP and local contrast measures are used as texture features.

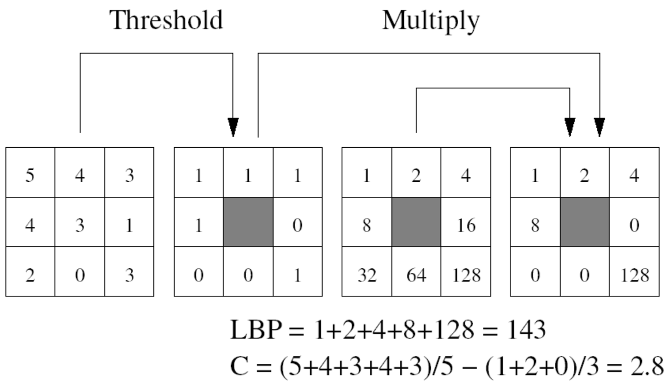

Fig. 13.6. Calculating LBP code and a contrast measure (*image reproduced with permission from Ref. 62*).

The LBP operator with a radial symmetric neighbourhood is invariant with respect to changes in illumination and image rotation (for example, compared to co-occurrence matrices), and computationally simple.[62] Ojala *et al.* demonstrated good performance for LBP in texture classification.

(21) **Markov random field (MRF)** – see "Random field models".

(22) **Oriented pyramid** (− − ◇−)
An oriented pyramid decomposes an image into several scales and different orientations. Unlike the Laplacian pyramid where there is no orientation information in each scale, in an oriented pyramid each scale represents textural energy at a particular direction. One way of generating an orientated pyramid is by applying derivative filters to a Gaussian pyramid or directional filters to a Laplacian pyramid, i.e. further decompose each scale. For an example of an oriented pyramid see Ref. 63. Also see "Steerable pyramids".

(23) **Power spectrum** (− − ◇−)
The power spectrum depicts the energy distribution in the frequency domain. It is commonly generated using the discrete form of the Fourier transform: [64]

$$F(u,v) = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \mathbf{I}(x,y) e^{-2\pi i(\frac{ux}{w} + \frac{vy}{h})}. \tag{13.16}$$

Then the power spectrum is obtained by computing the complex modulus (magnitude) of the Fourier transform, i.e. $P(u,v) = |F(u,v)|^2$. The radial distribution of energy in the power spectrum reflects the coarseness of the texture, and the angular distribution relates to the directionality. For example in Figure 13.7 the horizontal orientation of the texture features is reflected in the vertical energy distribution in the spectrum image. Thus, one can use these energy distributions to characterise textures. Commonly used techniques include applying ring filters, wedge filters, and peak extraction algorithms.

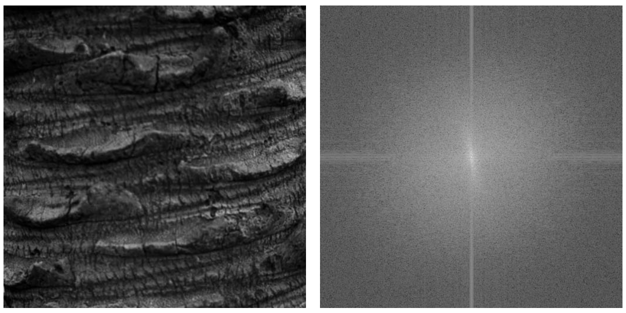

Fig. 13.7. Power spectrum image of a texture image - from left: the original image, and its Fourier spectrum image from which texture features can be computed.

(24) **Primal sketch** (−♡ − ♧)
Primal sketch attempts to extract distinctive image primitives as well as describe their spatial reletionship. Its concept was first introduced by Marr[65] as a symbolic representation of an image. It is considered as a representation of

image primitives or textons, such as bars, edges, blobs, and terminators. An image primitive extraction process is usually necessary, followed by a process of pursuing the sketch. Then, statistics, such as amount of different types of primitives, element orientation, distribution of size parameters, distribution of contrast of primitives, and spatial density of elements, can be extracted from the primal sketch for texture analysis.[66] Recently in Ref. 67, Guo *et al.* integrated sparse coding theory and the MRF concept as a primal sketch. The image was divided into sketchable regions, modelled using sparse coding, and non-sketchable regions, where the MRF based model was adopted. Textons were collected from the sketchable parts of the image. Figure 13.8 gives an example of a primal sketch with each element represented by bar or circles.

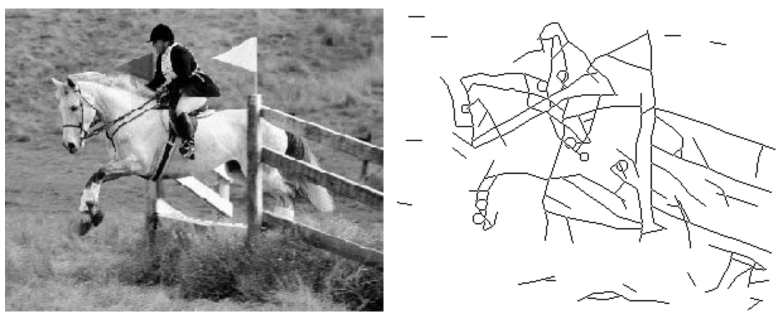

Fig. 13.8. An example of "primal sketch" - from left: The original image and its primal sketch with each element represented by a bar or a circle (images reproduced with permission from Ref. 67).

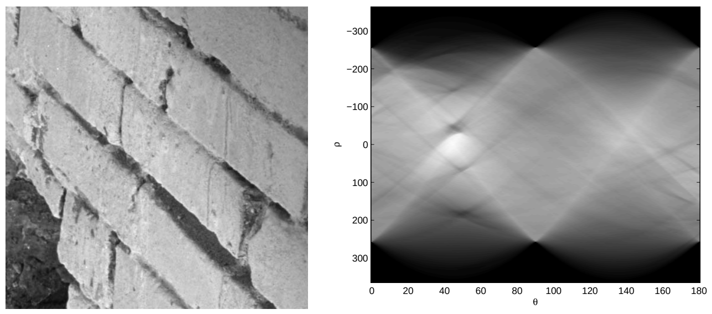

Fig. 13.9. An example of the Radon transform - from left: Original image and a visualisation of its Random transform.

(25) **Radon transform** (♤ − −−)

The Radon transform is an integral of a function over a set of all lines. A 2D

Radon transform of an image $\mathbf{I}(x, y)$ can be defined as:

$$R[\mathbf{I}(x,y)](\rho,\theta) = \sum_x \sum_y \mathbf{I}(x,y)\delta(\rho - x\cos\theta - y\sin\theta), \tag{13.17}$$

where θ is the angle between a line and the y-axis and ρ is the perpendicular distance of that line from the origin, which is the centre of the image. It can be used to detect linear trends in an image.[68] Thus, directional textures will exhibit “hot spots” in their Radon transform space. In Ref. 68, the Radon transform was used to find the dominant texture orientation which was later compensated to achieve rotational invariancy in texture classification. An example of the Radon transform of a texture is given in Fig. 13.9. The Radon transform is closely related to the Fourier, Hough, and Trace transforms.

(26) **Random field models** (− − −♣)
Markov Random Field (MRF) is a conditional probability model which provides a convenient way to model local spatial interactions among entities such as pixels. The establishment of the equivalence between MRFs and Gibbs distribution provided tractable means for statistical analysis as Gibbs distribution takes a much simpler form. Since then, MRFs have been applied to various applications, including texture synthesis[69] and texture classification.[35]
In MRF models, an image is represented by a finite rectangular lattice within which each pixel is considered as a site. Neighbouring sites then form cliques and their relationships are modelled in the neighbourhood system. Let the image $\mathbf{I}$ be represented by a finite rectangular $M \times N$ lattice $\mathcal{S} = \{s = (i, j) | 1 \leq i \leq M, 1 \leq j \leq N\}$, where s is a site in $\mathcal{S}$. A Gibbs distribution takes the following form

$$P(\mathbf{x}) = \frac{1}{Z} e^{-\frac{1}{T} U(\mathbf{x})}, \tag{13.18}$$

where T is a constant analogous to temperature, $U(\mathbf{x})$ is an energy function and Z is a normalising constant or *partition function* of the system. The energy is defined as a sum of *clique potentials* $V_c(\mathbf{x})$ over all possible cliques C:

$$U(\mathbf{x}) = \sum_{c \in C} V_c(\mathbf{x}). \tag{13.19}$$

If $V_c(\mathbf{x})$ is independent of the relative position of the clique c, the Gibbs random field (GRF) is said to be homogeneous. A GRF is characterised by its global property (the Gibbs distribution) whereas an MRF is characterised by its local property (the Markovianity).[32] Different distributions can be obtained by specifying the potential functions, such as Gaussian MRF (GMRF)[70] and the FRAME model.[71]

(27) **Random walk** (♤ – ––)
In Ref. 72, Kidode and Wechsler proposed a random walk procedure for texture analysis. The random walkers are moving in unit steps in one of the four given directions. The moving probabilities for a random walker at a given pixel to its four-connected neighbours are defined as a function of the underlying pixels. A very recent work on random walk based image segmentation can be found in Ref. 73, in which an image is treated as a graph with a fixed number of vertices and edges. Each edge is assigned a weight which corresponds to the likelihood a random walker will cross it. The user is required to select a certain number of seeds according to the number of regions to be segmented. Each unseeded pixel is assigned a random walker. The probilities for the random walker to reach those seed points are used to perform pixel clustering and image segmentation.

(28) **Relative extrema** (♤ – ––)
Relative extrema measures extract minimum and maximum values in a local neighbourhood. In Ref. 74, Mitchell *et al.* used relative frequency of the local gray level extremes to perform texture analysis. The number of extrema extracted from each scan line and their related threshold were used to characterise textures. This simple approach is a particularly useful trade-off in real-time applications.

(29) **Ring filter** (– – ♢–)
The Ring filter can be used to analyse texture energy distribution in the power spectrum as given in Eq. (13.16). In polar coordinates, it is defined as:

$$P(r) = 2\sum_{\theta=0}^{\pi} P(r,\theta), \tag{13.20}$$

where r denotes radius and θ is the angle. Figure 13.10 shows an example of a ring filter. The distribution of $P(r)$ indicates the coarseness of a texture. Also see the "Wedge filter".

(30) **Run lengths** (♤ – ––)
The gray level run length was introduced by Galloway in Ref. 11. A run is defined as consecutive pixels with the same gray level, collinear in the same direction. The number of pixels in a run is referred to as run length, and the frequency at which such a run occurs is known as run length value. Let $\mathbf{P}_\theta(i, j)$ be the run length matrix, each element of which records the frequency that j pixels with the same gray level i continue in the direction θ. Some of the statistics commonly extracted from run length matrices for texture analysis are listed in Table 13.2.

(31) **Scale-space primal sketch** (–♡ – ♧)

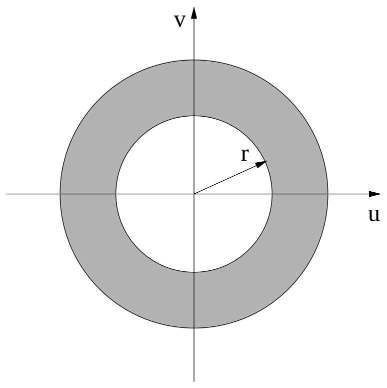

Fig. 13.10. A ring filter for power spectrum analysis.

Table 13.2. Some run length matrix features.

Measurement	Formula
Short runs emphasis	$\frac{\sum_i \sum_j \mathbf{P}_\theta(i,j)/j^2}{\sum_i,\sum_j \mathbf{P}_\theta(i,j)}$
Long runs emphasis	$\frac{\sum_i \sum_j j^2 \mathbf{P}_\theta(i,j)}{\sum_i \sum_j \mathbf{P}_\theta(i,j)}$
Gray level nonuniformity	$\frac{\sum_i \left\{\sum_j \mathbf{P}_\theta(i,j)\right\}^2}{\sum_i \sum_j \mathbf{P}_\theta(i,j)}$
Run length nonuniformity	$\frac{\sum_j \{\sum_i \mathbf{P}_\theta(i,j)\}^2}{\sum_i \sum_j \mathbf{P}_\theta(i,j)}$
Run percentage	$\frac{\sum_i \sum_j \mathbf{P}_\theta(i,j)}{wh}$

In this scale-space analysis, an image is usually successively smoothed using Gaussian kernels so that the original image is represented in multiscale. The hierarchical relationship among image primitives at different scales are then examined. In Refs. 75 and 39, the authors demonstrated that the scale-space primal sketch enables explicit extraction of significant image structures, such as blob-like features, which can be later used to characterise their spatial displacement rules. Also see the "Primal sketch".

(32) **Spectral histogram** $(- - \diamond -)$
The spectral histogram is translation invariant which is often a desirable property in texture analysis and with a sufficient number of filters it can uniquely represent any image up to a translation, as shown in Ref. 76. Essentially, a

spectral histogram is a vector consisting of the marginal distribution of filter responses. It implicitly combines the local structure of an image through examining spatial pixel relationships using filter banks and global statistics by computing marginal distribution. Let $\{F^{(\alpha)}, \alpha = 1, 2, ..., K\}$ denote a bank of filters. The image is convolved with these filters, and each filtering response generates a histogram:

$$\mathbf{H}_{\mathbf{I}}^{(\alpha)}(z) = \frac{1}{|\mathbf{I}|} \sum_{(x,y)} \delta\left(z - \mathbf{I}^{(\alpha)}(x,y)\right), \tag{13.21}$$

where z denotes a bin of the histogram, $\mathbf{I}^{(\alpha)}$ is the filtered image, and $\delta(.)$ is the Dirac delta function. Thus, the spectral histogram for the chosen filter bank is defined as :

$$\mathbf{H}_{\mathbf{I}} = \left(\mathbf{H}_{\mathbf{I}}^{(1)}, \mathbf{H}_{\mathbf{I}}^{(2)}, ..., \mathbf{H}_{\mathbf{I}}^{(K)}\right). \tag{13.22}$$

An example of using spectral histograms for texture analysis can be found in Ref. 76.

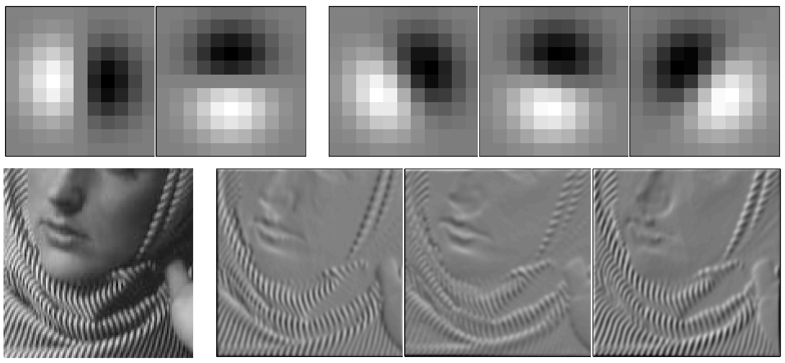

Fig. 13.11. A simple example of steerable filters - from left: The first row shows two basis functions, G_x and G_y, and three derived filters using basis functions at θ = 30, 80, and 140; The next row shows the original image and the three filter responses.

(33) **Steerable filters** $(- - \diamond -)$

The concept of steerable filters was first developed by Freeman and Adelson.[41] The steerable filter are a bank of filters with arbitrary orientations, each of which is generated using a linear combination of a set of basis functions. For example, we can use Gaussian derivative filters to generate steerable filters. For more general cases, please see Ref. 41. Let G_x and G_y denote the first x derivative and the first y derivative of a Gaussian function, respectively. Notably, G_y is merely a rotation of G_x. Then, a first derivative filter for any direction θ can now be easily synthesised via a linear combination of G_x and G_y:

$$D_\theta = G_x \cos\theta + G_y \sin\theta, \tag{13.23}$$

where $\cos\theta$ and $\sin\theta$ are known as the interpolation functions of the basis functions G_x and G_y. Figure 13.11 illustrates our Gaussian derivative based steerable filters. The first two images in the top row show the basis functions, G_x and G_y. The next three are "steered" filters at $\theta = 30$, 80, and 140 respectively. The bottom row shows the original image and the corresponding responses of the three filters. As expected these oriented filters exhibit selective responses at edges which is very useful for texture analysis. A recent application of steerable filters to texture classification can be found in Ref. 77.

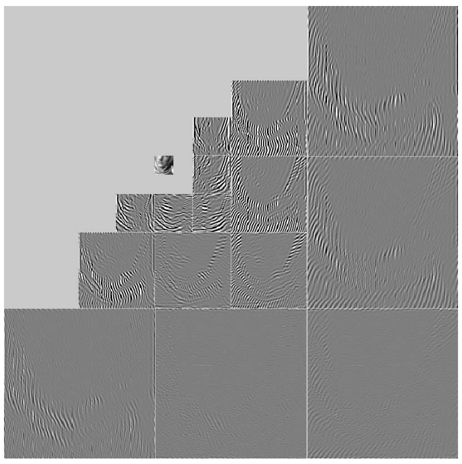
Fig. 13.12. A steerable pyramid representation of the image shown in Fig. 13.11. The original image is decomposed into 4 scales with the last scale as an excessively low pass filtered version. At each scale, the image is further decomposed to 5 orientations (the images are generated using the software provided by the authors in Ref. 78).

(34) **Steerable pyramid** $(- - \diamond -)$
A steerable pyramid is another way of analysing texture in multiple scales and different orientations. This pyramid representation is a combination of multiscale decomposition and differential measurements.[78] Its differential measurement is usually based on directional steerable basis filters. The basis filters are rotational copies of each other, and any directional copy can be generated using a linear combination of these basis functions. The pyramid

can have any number of orientation bands. As a result it does not suffer from aliasing, however, the pyramid is substantially over-complete which degrades its computational efficiency. Figure 13.12 gives an example steerable pyramid representation of the image shown in Fig. 13.11. Also see the "Steerable filter".

(35) **Texems** (−♡ − ♧)
In Ref. 79, Xie and Mirmehdi present a two layer generative model, called texems (short for texture exemplars), to represent texture images. Each texem, characterised by a mean and a covariance matrix, represents a class of image patches extracted from the original images. The original image is then described by a family of these texems, each of which is an implicit representation of a texture primitive. An example is given in Fig. 13.13 where four 7×7 texems are learnt from the given image. The notable difference between the texem and the texton is that the texem model relies directly on raw pixel values in stead of composition of base functions and it does not explicitly describe texture primitives as in the texton model, i.e. multiple or only partial primitives may be encapsulated in each texem. In Ref. 79, two different mixture models were investigated to derive texems for both gray level and colour images. An application to novelty detection in random colour textures was also presented.

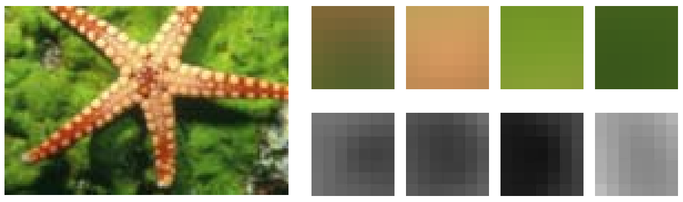

Fig. 13.13. Extracting texems from a colour image - from left: The original colour image and its four 7×7 texems, represented by mean and covariance matrices.

(36) **Textons** (−♡ − −)
Textons were first presented by Julesz[16] as fundamental image structures and were considered as atoms of pre-attentive human visual perception. Leung and Malik[42] adopted a *discriminative* model to describe textons. Each texture image was analysed using a filter bank composed of 48 Gaussian filters with different orientations, scales and phases. Thus, a high dimensional feature vector was extracted at each pixel position. K-means was used to cluster those filter response vectors into a few mean vectors which were referred to as textons.

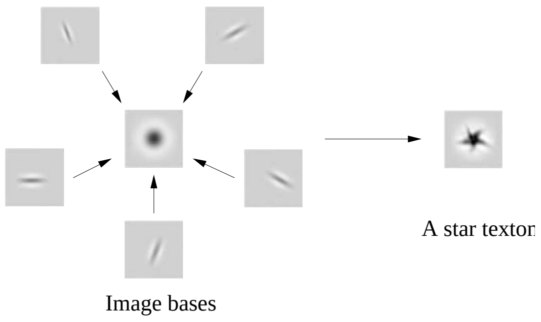

Fig. 13.14. A star texton configuration (*image adapted from*[80]).

More recently, Zhu *et al.*[80] argued that textons could be defined in the context of a *generative* model of images. In their three-level generative model, an image I was considered as a superposition of a number of image base functions that were selected from an over-complete dictionary $\mathbf{\Psi}$. These image bases, such as Gabor and Laplacian of Gaussian functions at various scales, orientations, and locations, were generated by a smaller number of texton elements which were in turn selected from a texton dictionary $\mathbf{\Pi}$. An image I is generated by a base map $\mathbf{B}$ which is in turn generated from a texton map $\mathbf{T}$, i.e:

$$\mathbf{T} \xrightarrow{\mathbf{\Pi}} \mathbf{B} \xrightarrow{\mathbf{\Psi}} \mathbf{I}, \tag{13.24}$$

where $\mathbf{\Pi} = \{\pi_i, i = 1, 2, ...\}$ and $\mathbf{\Psi} = \{\psi_i, i = 1, 2, ...\}$. Each texton, an instance in the texton map $\mathbf{T}$, is considered as a combination of a certain number of base functions with deformable geometric configurations, e.g. star, bird, snowflake. This configuration can be illustrated using a texton of a star shape as shown in Fig. 13.14. By fitting this generative model to observed images, the texton dictionary then is learnt as parameters of the generative model. Example applications of the texton model can be found in Refs. 42,81,82 and 83.

(37) **Texture spectrum** ($\spadesuit\heartsuit - -$)
Similar to the texton approach, the texture spectrum method[84] considers a texture image a composition of texture units and uses the global distribution of these units to characterise textures. Each texture unit comprises a small local neighbourhood, e.g. 3×3, and the pixels within are thresholded according to the central pixel intensity in a very similar approach to LBP's approach. Pixels brighter or darker than the central pixel are set to 0 or 2 respectively, and the rest of the pixels are set to 1. These values are then vectorised to form a

feature vector for the central pixel, the frequency of which is computed across the image to form the texture unit spectrum. Various characteristics from this spectrum are extracted to perform texture analysis, such as symmetricity and orientation.

(38) **Trace transform** (♤ – ––)
The trace transform[85] is a 2D representation of an image in polar coordinates with the origin in the centre of the image. Similar to the Radon transform, it traces lines from all possible directions originating from the centre but instead of computing the integral as in the Radon transform, it evaluates several other functionals along each trace line. Thus, it is considered as a generalisation of the Radon transform. In practice, different functionals are used to produce different trace transforms from the same image. Features can then be extracted from transformed images using diametrical and circus functionals. Figure 13.15 gives an example of the trace transform.

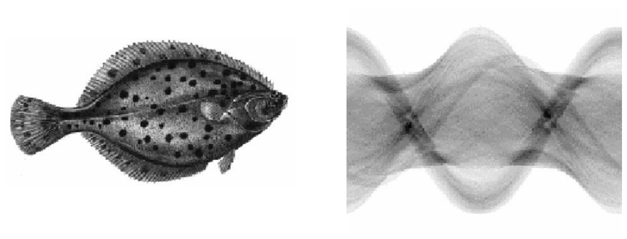

Fig. 13.15. An example of the trace transform (images reproduced with permission from Ref. 85).

(39) **Voronoi tessellation** (–♡ – –)
Voronoi tessellation, introduced by Ahuja,[86] divides a domain into a number of polygonal regions based on a set of given points in this domain. Each polygon contains one given point only and any points that are closer to this given point than any others. The shape of the polygonal regions, or Voronoi pologons, reflect the local spatial point distributions. Figure 13.16 shows an example of Voronoi tessellation. In Ref. 59, Tuceryan and Jain first extracted texture tokens, such as local extrema, line segmentations, and terminations, and then used Voronoi tessellation to divide the image plane. Features from this tessellation, such as area of the pologal regions, its shape and orientation, and relative position to the tokens, were used for texture segmentation.

(40) **Wavelets** (– – ♢–)
Wavelet based texture analysis uses a class of functions that are localised in both spatial and spatial-frequency domain to decompose texture images.

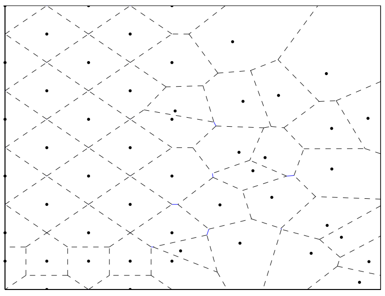

Fig. 13.16. An example of Voronoi tessellation - The dots are feature points, and the tessellation is shown in dashed lines. The points on the left hand are regularly distributed and those on the right randomly placed. These are reflected in the shape and distribution of the polygonal regions.

Wavelet functions belonging to the same family can be constructed from a basis function, known as "mother wavelet" or "basic wavelet", by means of dilation and translation. The input image is considered as the weighted sum of overlapping wavelet functions, scaled and shifted. Let $g(x)$ be a wavelet (in $1D$ form, for simplicity). The wavelet transform of a $1D$ signal $f(x)$ is defined as

$$W_f(\alpha, \tau) = \int_{-\infty}^{\infty} f(x) g^*(\alpha(x - \tau)) dx, \tag{13.25}$$

where $g(\alpha(x - \tau))$ is computed from the mother wavelet $g(x)$, and τ and α denote the translation and scale respectively. The discrete equivalent can be obtained by sampling the parameters α and τ. Typically, the sampling constraints require the transform to be a non-redundant complete orthogonal decomposition. Every transformed signal contains information of a specific scale and orientation. Popular wavelet transform techniques that have been applied to texture analysis include dyadic transform, pyramidal wavelet transform, and wavelet packet decomposition, e.g. Ref. 56.

(41) **Wedge filter** $(- - \diamond -)$

Along with the ring filter, the wedge filter is used to analyse energy distribution in the frequency domain. The image is transformed into the power spectrum, usually using the fast Fourier transform, and wedge filters are applied to examine the directionality of its texture. A wedge filter in polar coordinates can be defined as:

$$P(\theta) = \sum_{r=0}^{\infty} P(r, \theta), \tag{13.26}$$

where r denotes the radius and θ the angle. Figure 13.17 illustrates a wedge filter in a polar coordinates. Also see the "Ring filter".

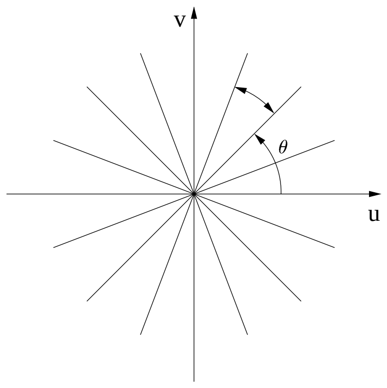

Fig. 13.17. A wedge filter for power spectrum analysis.

(42) **Wigner distribution** $(- - \diamond -)$

The Wigner distribution also gives a joint representation in the spatial and spatial-frequency domain. It is sometimes described as a local spatial frequency representation. Considering a 1D case, let $f(x)$ denote a continuous, integrable and complex function. The Wigner distribution can be defined as:

$$WD(x,\omega) = \int_{-\infty}^{\infty} f(x + \frac{x'}{2}) f^*(x - \frac{x'}{2}) e^{-i\omega x'} dx', \qquad (13.27)$$

where ω is the spatial frequency and $f^*(.)$ is the complex conjugate of $f(.)$. The Wigner distribution directly encodes the phase information and unlike the short time Fourier transform it is a real valued function. Example applications of Wigner distribution to feature extraction and image analysis can be found in Ref. 87. In Ref. 88, the authors demonstrated detecting cracks in random textures based on Wigner distrbution. Also see the "Wavelets".

13.3. Texture Feature Comparison

There have been many studies comparing various subsets of texture features. As a pointer, here we briefly mention some of these studies. In general, the results in most of these works much depend on the data set used, the application domain, and the set of parameters used for the methods examined.

In Ref. 89, Ohanian and Dubes compared the fractal model, co-occurrence matrices, the MRF model, and Gabor filtering for texture classification. The co-occurrence features generally outperformed other features in terms of classification rate. However, as pointed out in Ref. 6, they used raw Gabor filtered images instead of using empirical nonlinear transformations to obtain texture features.

Reed and Wechsler[90] performed a comparative study on various spatial/spatial-frequency representations and concluded that the Wigner distribution had the best joint resolution. In another related work, Pichler *et al.*[91] reported superior results using Gabor filtering over other wavelet transforms.

In Ref. 92, Chang *et al.* evaluated co-occurrence matrices, Laws texture energy, and Gabor filters for segmentation in natural and synthetic images. Gabor filtering again achieved best performance. Later, Randen and Husøy[56] performed an extensive evaluation of various filtering approaches for texture segmentation. The methods included Laws filters, ring and wedge filters, various Gabor filters, and wavelet transforms. No single approach was found to be consistently superior to the others on their twelve texture collages.

Singh and Singh[93] compared seven spatial texture analysis techniques, including autocorrelation, co-occurrence matrices, Laws filters, run lengths, and statistical geometrical (SG) features.[94] The SG features performed best in classifying VisTex and MeasTex[95] textures. In the SG method, the image was segmented into a binary stack depending on the number of graylevels in the image. Then geometrical measurements of the connected regions in each stack were taken as texture features.

Recently, Varma and Zisserman[96] compared two statistical approaches to classify material images from the Columbia-Utrecht (CUReT)[81] texture database. Both approaches applied a filter bank consisting isotropic Gaussian, Laplacian of Gaussian, and orientated edge filters at various scales and orientations. However, the first method, following the work of Konishi and Yuille,[97] directly estimated the distribution of filtering responses and classified the texture images based on the class conditional probability using the Bayesian theorem. The second approach, adopted in Refs. 42,98 and 99, clustered the filtering responses to generate texton representations and used texton frequency to classify textures based on the χ^2 distance measure. The results showed close performance of these two approaches. However, the Bayesian approach degraded quicker when less information in estimating the underlying distribution was available.

In Ref. 100, Drimbarean and Whelan presented a comparative study on colour texture classification. The local linear filter based on discrete Cosine transform (DCT), Gabor filters, and co-occurrence matrices were studied along with different colour spaces, such as RGB and $L^*a^*b^*$. The results showed that colour information was important in characterising textures. The DCT features were found the best of the three when classifying selected colour images from the VisTex dataset.[101]

References

1. R. Haralick, Statistical and structural approaches to texture, *Proceedings of the IEEE*. **67**(5), 786–804, (1979).
2. H. Wechsler, Texture analysis - a survey, *Signal Processing*. **2**, 271–282, (1980).
3. L. Van Gool, P. Dewaele, and A. Oosterlinck, Texture analysis, *Computer Vision, Graphics and Image Processing*. **29**, 336–357, (1985).
4. F. Vilnrotter, R. Nevatia, and K. Price, Structural analysis of natural textures, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **8**, 76–89, (1986).
5. T. Reed and J. Buf, A review of recent texture segmentation and feature extraction techniques, *Computer Vision, Image Processing and Graphics*. **57**(3), 359–372, (1993).
6. M. Tuceryan and A. Jain. Texture analysis. In *Handbook of Pattern Recognition and Computer Vision*, chapter 2, pp. 235–276. World Scientific, (1998).
7. L. Latif-Amet, A. Ertuzun, and A. Ercil, An efficient method for texture defect detection: Subband domain co-occurrence matrices, *Image and Vision Computing*. **18** (6-7), 543–553, (2000).
8. R. Haralick, K. Shanmugan, and I. Dinstein, Textural features for image classification, *IEEE Transactions on Systems, Man, and Cybernetics*. **3**(6), 610–621, (1973).
9. M. Tsatsanis and G. Giannakis, Object and texture classification using higher order statistics, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **14**(7), 733–750, (1992).
10. Y. Huang and K. Chan, Texture decomposition by harmonics extraction from higher order statistics, *IEEE Transactions on Image Processing*. **13**(1), 1–14, (2004).
11. R. Galloway, Texture analysis using gray level Run lengths, *Computer Graphics and Image Processing*. **4**, 172–179, (1974).
12. T. Ojala, M. Pietikäinen, and D. Harwood, A comparative study of texture measures with classification based on feature distribution, *Pattern Recognition*. **29**(1), 51–59, (1996).
13. S. Zucker, Toward a model of texture, *Computer Graphics and Image Processing*. **5**, 190–202, (1976).
14. K. Fu, *Syntactic Pattern Recognition and Applications*. (Prentice-Hall, 1982).
15. D. Marr, Early processing of visual information, *Philosophical Transactions of the Royal Society of London*. **B-275**, 483–524, (1976).
16. B. Julesz, Textons, the element of texture perception and their interactions, *Nature*. **290**, 91–97, (1981).
17. A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pp. 1033–1038, (1999).
18. J. Malik and P. Perona, Preattentive texture discrimination with early vision mechanisms, *Journal of the Optical Society of America, Series A*. **7**, 923–932, (1990).
19. F. Ade, Characterization of texture by 'eigenfilter', *Signal Processing*. **5**(5), 451–457, (1983).
20. I. Jolliffe, *Principal Component Analysis*. (Springer-Verlag, 1986).
21. J. Coggins and A. Jain, A spatial filtering approach to texture analysis, *Pattern Recognition Letters*. **3**, 195–203, (1985).
22. F. D'Astous and M. Jernigan. Texture discrimination based on detailed measures of

the power spectrum. In *International Conference on Pattern Recognition*, pp. 83–86, (1984).

23. M. Turner, Texture discrimination by Gabor functions, *Biological Cybernetics*. **55**, 71–82, (1986).
24. M. Clark and A. Bovik, Texture segmentation using Gabor modulation/demodulation, *Pattern Recognition Letters*. **6**, 261–267, (1987).
25. H. Sari-Sarraf and J. Goddard, Vision systems for on-loom fabric inspection, *IEEE Transactions on Industry Applications*. **35**, 1252–1259, (1999).
26. J. Scharcanski, Stochastic texture analysis for monitoring stochastic processes in industry, *Pattern Recognition Letters*. **26**, 1701–1709, (2005).
27. X. Yang, G. Pang, and N. Yung, Robust fabric defect detection and classification using multiple adaptive wavelets, *IEE Proceedings Vision, Image Processing*. **152** (6), 715–723, (2005).
28. R. Coifman, Y. Meyer, and V. Wickerhauser. Size properties of wavelet packets. In eds. M. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, *Wavelets and Their Applications*, pp. 453–470. Jones and Bartlett, (1992).
29. B. Mandelbrot, *The Fractal Geometry of Nature*. (W.H. Freeman, 1983).
30. J. Mao and A. Jain, Texture classification and segmentation using multiresolution simultaneous autoregressive models, *Pattern Recognition*. **25**(2), 173–188, (1992).
31. M. Comer and E. Delp, Segmentation of textured images using a multiresolution Gaussian autoregressive model, *IEEE Transactions on Image Processing*. **8**(3), 408–420, (1999).
32. S. Li, *Markov Random Filed Modeling in Image Analysis*. (Springer, 2001).
33. N. Jojic, B. Frey, and A. Kannan. Epitomic analysis of appearance and shape. In *IEEE International Conference on Computer Vision*, pp. 34–42, (2003).
34. C. Coroyer, D. Declercq, and P. Duvaut. Texture classification using third order correlation tools. In *IEEE Signal Processing Workshop on High-Order Statistics*, pp. 171–175, (1997).
35. A. Khotanzad and R. Kashyap, Feature selection for texture recognition based on image synthesis, *IEEE Transactions on Systems, Man, and Cybernetics*. **17**(6), 1087–1095, (1987).
36. L. Siew, R. Hodgson, and E. Wood, Texture measures for carpet wear assessment, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **10**, 92–105, (1988).
37. D. Clausi, An analysis of co-occurrence texture statistics as a function of grey level quantization, *Canadian Journal of Remote Sensing*. **28**(1), 45–62, (2002).
38. A. Monadjemi. *Towards Efficient Texture Classification and Abnormality Detection*. PhD thesis, University of Bristol, UK, (2004).
39. T. Lindeberg, Detecting salient blob-like image structures and scales with a scale-space primal sketch: A method for focus-of-attention, *International Journal of Computer Vision*. **11**(3), 283–318, (1993).
40. D. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*. **60**(2), 91–110, (2004).
41. W. Freeman and E. Adelson, The design and use of steerable filters, *IEEE Transac-*

tions on Pattern Analysis and Machine Intelligence. **13**(9), 891–906, (1991).
42. T. Leung and J. Malik, Representing and recognizing the visual appearance of materials using three-dimensional textons, *International Journal of Computer Vision*. **43** (1), 29–44, (2001).
43. A. Monadjemi, M. Mirmehdi, and B. Thomas. Restructured eigenfilter matching for novelty detection in random textures. In *British Machine Vision Conference*, pp. 637–646, (2004).
44. C. Fredembach, M. Schröder, and S. Süsstrunk, Eigenregions for image classification, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **26**(12), 1645–1649, (2004).
45. L. Chang and C. Cheng. Multispectral image compression using eigenregion based segmentation. In *IEEE International Geoscience and Remote Sensing Symposium*, vol. 4, pp. 1844–1846, (2001).
46. C. Stauffer. Learning a probabilistic similarity function for segmentation. In *IEEE Workshop on Perceptual Organization in Computer Vision*, pp. 50–58, (2004).
47. V. Cheung, B. Frey, and N. Jojic. Video epitome. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 42–49, (2005).
48. A. Pentland, Fractal-based description of nature scenes, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **9**, 661–674, (1984).
49. J. Gangepain and C. Roques-Carmes, Fractal approach to two dimensional and three dimensional surface roughness, *Wear*. **109**, 119–126, (1986).
50. R. Voss. Random fractals: Characterization and measurement. In eds. R. Pynn and A. Skjeltorp, *Scaling Phenomena in Disordered Systems*. Plenum, (1986).
51. J. Keller, S. Chen, and R. Crownover, Texture description and segmentation through fractal geometry, *Computer Vision, Graphics, and Image Processing*. **45**, 150–166, (1989).
52. B. Super and A. Bovik, Localizing measurement of image fractal dimension using Gabor filters, *Journal of Visual Communication and Image Representation*. **2**, 114–128, (1991).
53. C. Allain and M. Cloitre, Characterizing the lacunarity of random and deterministic fractal sets, *Physical Review*. **A-44**(6), 3552–3558, (1991).
54. A. Jain and F. Farrokhnia, Unsupervised texture segmentation using Gabor filters, *Pattern Recognition*. **24**, 1167–1186, (1991).
55. A. Kumar and G. Pang, Defect detection in textured materials using Gabor filters, *IEEE Transactions on Industry Applications*. **38**(2), 425–440, (2002).
56. T. Randen and J. Husøy, Filtering for texture classification: a comparative study, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **21**(4), 291–310, (1999).
57. R. Conners and C. Harlow, A theoretical comparison of texture algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **2**(3), 204–222, (1980).
58. M. Swain and D. Ballard, Indexing via color histograms, *International Journal of Computer Vision*. **7**(1), 11–32, (1990).
59. M. Tuceryan and A. Jain, Texture segmentation using voronoi polygons, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **12**, 211–216, (1990).
60. P. Burt and A. Adelson, The laplacian pyramid as a compact image code, *IEEE Transactions on Communications*. **31**, 532–540, (1983).

61. K. Laws. *Textured Image Segmentation*. PhD thesis, University of Southern California, USA, (1980).
62. T. Mäenpää and M. Pietikäinen. Texture analysis with local binary patterns. In eds. C. Chen and P. Wang, *Handbook of Pattern Recognition and Computer Vision*, pp. 197–216. World Scientific, 3 edition, (2005).
63. E. Simoncelli, W. Freeman, E. Adelson, and D. Heeger, Shiftable multi-scale transforms, *IEEE Transactions on Information Theory*. **38**(2), 587–607, (1992).
64. R. Gonzalez and R. Woods, *Digital Image Processing*. (Addison Wesley, 1992).
65. D. Marr, *Vision*. (W. H. Freeman and Company, 1982).
66. F. Tomita and S. Tsuji, *Computer Analysis of Visual Textures*. (Kluwer Academic Publisher, 1990).
67. C. Guo, S. Zhu, and Y. Wu. Towards a mathematical theory of primal sketch and sketchability. In *IEEE International Conference on Computer Vision*, vol. 2, pp. 1228–1235, (2003).
68. K. Jafari-Khouzani and H. Soltanian-Zadeh, Radon transform orientation estimation for rotation invariant texture analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **27**(6), 1004–1008, (2005).
69. G. Cross and A. Jain, Markov random field texture models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **5**, 25–39, (1983).
70. R. Chellappa. Two-dimensional discrete Gaussian Markov random field models for image processing. In eds. L. Kanak and A. Rosenfeld, *Progress in Pattern Recognition 2*. Elsevier, (1985).
71. S. Zhu, Y. Wu, and D. Mumford, FRAME: Filters, random field and maximum entropy - towards a unified theory for texture modeling, *International Journal of Computer Vision*. **27**(2), 1–20, (1997).
72. H. Wechsler and M. Kidode, A random walk procedure for texture discrimination, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **1**(3), 272–280, (1979).
73. L. Grady, Random walks for image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **9**(11), 1768–1783, (2006).
74. O. Mitchell, C. Myers, and W. Boyne, A min-max measure for image texture analysis, *IEEE Transactions on Computers*. **C-26**, 408–414, (1977).
75. T. Lindeberg and J. Eklundh, Scale-space primal sketch: Construction and experiments, *Image and Vision Computing*. **10**(1), 3–18, (1992).
76. X. Liu and D. Wang, Texture classification using spectral histograms, *IEEE Transactions on Image Processing*. **12**(6), 661–670, (2003).
77. Y. Wu, K. Chan, and Y. Huang. Image texture classification based on finite gaussian mixture models. In *International Workshop on Texture Analysis and Synthesis*, pp. 107–112, (2003).
78. E. Simoncelli and W. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *IEEE International Conference on Image Processing*, pp. 444–447, (1995).
79. X. Xie and M. Mirmehdi, TEXEM: Texture exemplars for defect detection on random textured surfaces, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. (2007). to appear.
80. S. Zhu, C. Guo, Y. Wang, and Z. Xu, What are textons?, *International Journal of*

Computer Vision. **62**(1-2), 121–143, (2005).

81. K. Dana, B. Ginneken, S. Nayar, and J. Koenderink, Reflectance and texture of real-world surfaces, *ACM Transactions on Graphics*. **18**(1), 1–34, (1999).
82. C. Schmid. Constructing models for content-based image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 39–45, (2001).
83. M. Varma and A. Zisserman, A statistical approach to texture classification from single images, *International Journal of Computer Vision*. **61**(1/2), 61–81, (2005).
84. D. He and L. Wang, Texture features based on texture spectrum, *Pattern Recognition*. **24**(5), 391–399, (1991).
85. A. Kadyrov and M. Petrou, The trace transform and its applications, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **23**(8), 811–828, (2001).
86. N. Ahuja, Dot pattern processing using voronoi neighbourhoods, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **4**, 336–343, (1982).
87. G. Cristobal, C. Gonzalo, and J. Bescos. Image filtering and analysis through the wigner distribution. In ed. P. Hawkes, *Advances in Electronics and Electron Physics Series*, vol. 80, pp. 309–397. Academic Press, (1991).
88. C. Boukouvalas, J. Kittler, R. Marik, M. Mirmehdi, and M. Petrou. Ceramic tile inspection for colour and structural defects. In *Advances in Materials and Processing Technologies*, pp. 390–399, (1995).
89. P. Ohanian and R. Dubes, Performance evaluation for four classes of textural features, *Pattern Recognition*. **25**(8), 819–833, (1992).
90. T. Reed and H. Wechsler, Segmentation of textured images and gestalt organization using spatial/spatial-frequency representations, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **12**, 1–12, (1990).
91. O. Pichler, A. Teuner, and B. Hosticka, A comparison of texture feature extraction using adaptive Gabor filter, pyramidal and tree structured wavelet transforms, *Pattern Recognition*. **29**(5), 733–742, (1996).
92. K. Chang, K. Bowyer, and M. Sivagurunath. Evaluation of texture segmentation algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 294–299, (1999).
93. M. Singh and S. Singh. Spatial texture analysis: A comparative study. In *International Conference on Pattern Recognition*, vol. 1, pp. 676–679, (2002).
94. Y. Chen, M. Nixon, and D. Thomas, Statistical geometrical features for texture classification, *Pattern Recognition*. **28**(4), 537–552, (1995).
95. G. Smith and I. Burns, Measuring texture classification algorithm, *Pattern Recognition Letters*. **18**, 1495–1501, (1997).
96. M. Varma and A. Zisserman, Unifying statistical texture classification frameworks, *Image and Vision Computing*. **14**(1), 1175–1183, (2004).
97. S. Konishi and A. Yuille. Statistical cues for domain specific image segmentation with performance analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 125–132, (2000).
98. O. Cula and K. Dana. Compact representation of bidirectional texture functions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1041–1047, (2001).
99. M. Varma and A. Zisserman. Classifying images of materials from images: Achieving viewpoint and illumination independence. In *European Conference on Computer*

Vision, pp. 255–271, (2002).

100. A. Drimbarean and P. Whelan, Experiments in colour texture analysis, *Pattern Recognition Letters*. **22**, 1161–1167, (2001).
101. MIT Media Lab. VisTex texture database, (1995). URL `http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html`.

Subject Index