Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition 39 (2006) 695 – 706
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

A study of Gaussian mixture models of color and texture features for image classification and segmentation

Haim Permuter[a,*], Joseph Francos[b], Ian Jermyn[c]

[a]Department of Electrical Engineering, Stanford University, CA 94305, USA
[b]Electrical and Computer Engineering Department, Ben-Gurion University, Beer Sheva 84105, Israel
[c]Ariana (Joint INRIA/I3S Research Group), INRIA, B.P. 93, 06902 Sophia Antipolis, France

Received 11 February 2005; received in revised form 19 October 2005; accepted 19 October 2005

Abstract

The aims of this paper are two-fold: to define Gaussian mixture models (GMMs) of colored texture on several feature spaces and to compare the performance of these models in various classification tasks, both with each other and with other models popular in the literature. We construct GMMs over a variety of different color and texture feature spaces, with a view to the retrieval of textured color images from databases. We compare supervised classification results for different choices of color and texture features using the Vistex database, and explore the best set of features and the best GMM configuration for this task. In addition we introduce several methods for combining the 'color' and 'structure' information in order to improve the classification performances. We then apply the resulting models to the classification of texture databases and to the classification of man-made and natural areas in aerial images. We compare the GMM model with other models in the literature, and show an overall improvement in performance.
© 2005 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.

Keywords: Image classification; Image segmentation; Texture; Color; Gaussian mixture models; Expectation maximization; *k*-means; Background model; Decision fusion; Aerial images

1. Introduction

In many domains of image processing, there is a strong correspondence between entities in the scene and textures[1] in the image. This implies that the ability to recognize these textures can furnish important semantic information about the scene. Consequently, the problems of texture description and classification, and the closely related problem of segmentation, have received considerable attention, with numerous approaches being proposed (Refs. [1,2] and references therein). In particular, in the field of content-based image retrieval, the ability to answer the question: "Is there a significant amount of such-and-such texture in this image?", can be the basis for many types of query.

Two variations on the problem exist: supervised and unsupervised segmentation. In the former, models of the texture associated with different entities in the scene are assumed known, and are then applied to the image in the hope of segmenting it into regions corresponding to those entities. Clearly this requires a training stage in which human beings group texture exemplars into classes, corresponding to the entities involved, from which the corresponding model parameters are then learnt. In the unsupervised case, no models are known a priori. Instead, the aim is to discover similarities in the data that betray the existence of one or more distinct classes into which the data can be divided. This may or may not involve explicitly learning the model parameters. When the entities in the scene into which the image should be segmented are not decided upon beforehand, as they often are not, unsupervised segmentation is methodologically

* Corresponding author.
E-mail addresses: haim1@stanford.edu (H. Permuter), francos@ee.bgu.ac.il (J. Francos), ian.jermyn@inria.fr (I. Jermyn).

[1] By the word 'texture', we denote both what we will later call 'structure' information, and color information.

ill-defined, since no specification of the ideal result is given. In supervised segmentation on the other hand, texture classes necessarily correspond to distinct entities in the scene, and the success or failure of the segmentation can be decided on this basis. In this paper, we consider the supervised texture segmentation problem.

1.1. Literature

Many kinds of statistical models have been applied to texture classification. These include Bayes classifiers assuming multivariate Gaussian distributions for the features [3–6]; Fisher transformation [7,8]; nonparametric nearest-neighbor classification [9–12]; classification trees [8]; learning vector quantization [13,14]; feed-forward neural networks [15]; and recently support vector machine [16,17] and multiple histogram combined with self-organized map [18] . In some earlier cases, the statistical modelling after the feature extraction is just thresholding [19–22]; or simple extremum picking [23–25]. Markov random fields, and especially Gaussian Markov random fields have been extensively used for texture modelling and segmentation since the early work in Ref. [26]. For a good review, see the paper by Geman and Graffigne [27]. Li and Gray [28] proposed a 2D hidden Markov model (HMM) for image classification, while a somewhat different model is the noncausal HMM described in Ref. [29].

Another recent class of models uses hidden Markov trees (HMTs) to model the joint statistics of wavelet coefficients. Tree models sacrifice some descriptive power (usually only inter- rather than intra-scale dependencies) to ease of implementation (many algorithms that work in the case of linear graphs, also work on trees, but not on more complicated models). HMT models were first introduced in Ref. [30], and were applied to texture analysis in Refs. [31,32]. They are typically used, even in texture applications, with binary-valued hidden state variables that switch between high and low variance Gaussian distributions for the wavelet coefficients. This behavior is intended to capture the difference between edges and noisy but otherwise smooth regions in images, an important distinction for ‘edge-preserving denoising’. Indeed, for denoising, HMTs result in state of the art algorithms. It is not clear however, that they remain appropriate for single textures, whose statistics may differ markedly from those for natural images considered as a whole. In particular, the division into ‘edges’ and ‘noise’ seems strange in this context. HMTs are used in Ref. [33], where texture and color are combined in an HMT model. Texture is described using HMTs of grayscale wavelet coefficient magnitudes, while color is described using independent Gaussian distributions at each scale for the colored scaling coefficients.

In recent work [34], we proposed the use of Gaussian mixture models (GMMs) for texture classification, demonstrating improved performance over other, computationally more expensive methods. This paper is an extension of the work presented there. In related but differently directed work, Gray et al. [35] also used GMMs for image classification.

2. Classification, GMMs, and feature spaces

In this section, we describe in top-down fashion the models we will use. We begin with our general approach to the classification problem, and continue by describing the place of GMMs within that framework. Finally, we describe the various feature spaces on which the GMMs are defined. We assume throughout that we are dealing with N classes, labelled by $n \in N$.[2]

2.1. Classification

Any classification model is defined on the space $\mathscr{N}$ of maps from the image domain to the set N of classes (each class n corresponds to an entity of interest in the scene), the possible ‘classifications’. Thus each classification $v \in \mathscr{N}$ assigns a class $n = v(p) \in N$ to each pixel p giving the class of that pixel. By defining a posterior probability distribution on $\mathscr{N}$, and using a suitable loss function, an optimal classification can be chosen. The loss function is more often than not taken to be the negative of a delta function, the resulting estimate then being a maximum a posteriori (MAP) estimate. The posterior distribution is expressed as the (normalized) product of a likelihood, such as the GMM models that we will discuss in this paper, which gives the distribution of images corresponding to a given class, and a prior probability distribution on the classifications.

Prior models for the classification v usually have a minimum sub-image that can be analyzed. Typically such a model assumes that the regions corresponding to classes are larger than the minimum sub-image. A similar but different restriction is that the neighboring pixels of a given pixel will be with a higher probability from the same class than from an other class. A standard choice for the prior is thus the Potts model, which penalizes a classification by the total length of class boundary it contains. Unfortunately, the use of such a model renders the MAP estimation problem hard to solve, at least rapidly. In order to avoid this problem of computational complexity while producing a similar effect, we use two heuristics: we assume that v is constant on $S \times S$ subimages, called ‘blocks’, but that its values on different blocks are independent and equiprobable; and we use a loss function/classification rule that incorporates a local ‘averaging’ of the class over block neighborhoods called ‘patches’. The set of blocks in an image will be denoted B, and individual blocks by b. The neighborhood patch $P(b)$ of a block b is the set of blocks in a larger $T \times T$ subimage with b at its center.

[2] Throughout we use an integer N to represent both the number itself and the set $\{1, \ldots, N\}$.

For the likelihood, we assume that while the pixels within a block may be strongly dependent, the data in two different blocks are independent given their classes. Thus the likelihood of an image I given the classification v is given by

$$\Pr(I|v) = \prod_{b\in B} \Pr(I_b|v_b), \tag{1}$$

where the subscript b indicates a function restricted to the block b. Given our assumptions about the prior probability of v, the posterior probability of v given an image is then

$$\begin{aligned}\Pr(v|I) &= \frac{1}{\Pr(I)} \Pr(I|v)\Pr(v) \\ &= \prod_{b\in B} \frac{1}{\Pr(I_b)} \Pr(I_b|v_b)\frac{1}{N} \\ &= \prod_{b\in B} \Pr(v_b|I_b).\end{aligned} \tag{2}$$

In order to derive estimates of v, we introduce the following loss function:

$$L(v^*, v) = -\sum_{b\in B} \prod_{b'\in P(b)} \delta(v^*_b, v_{b'}), \tag{3}$$

where v^* is the proposed classification, and v is the true classification for which we know only the posterior probability $\Pr(v|I)$. The expected value of this loss function is

$$\begin{aligned}\langle L\rangle(v^*) &= \sum_v L(v^*, v)\Pr(v|I) \\ &= -\sum_v \sum_{b\in B} \left[\prod_{b'\in P(b)} \delta(v^*_b, v_{b'})\right] \\ &\quad\times \left[\prod_{b''\in B} \Pr(v_{b''}|I_{b''})\right] \\ &= -\sum_{b\in B} \sum_{v_{P(b)}} \left[\prod_{b'\in P(b)} \delta(v^*_b, v_{b'})\Pr(v_{b'}|I_{b'})\right] \\ &= -\sum_{b\in B} \left[\prod_{b'\in P(b)} \Pr(v_{b'} = v^*_b|I_{b'})\right].\end{aligned} \tag{4}$$

Minimization of the mean loss and use of Eq. (2) then leads to the classification rule

$$\hat{v}_b = \arg\max_{n\in N} \left[\prod_{b'\in P(b)} \Pr(I_{b'}|v_{b'} = n)\right]. \tag{5}$$

This says: "Assign to a block b that class n which, if all the blocks in the patch $P(b)$ had class n, would maximize the probability of the data in $P(b)$". The effect of this classification rule is similar to that of a Potts prior, in that it encourages spatial homogeneity of the classification and smooth boundaries, because neighboring blocks have overlapping patches. Its great advantage is that it is not necessary to consider the unknown classes of neighboring blocks in making a classification decision. This reduces computation time considerably. The nature of the classification rule is elucidated by considering that if $P(b) = \{b\}$, the loss function reduces to the negative of the number of correctly classified pixels, i.e. up to an additive constant, the number of misclassified pixels, which is the loss function used for maximum posterior marginal (MPM) classification.

Note what happens to the loss function when the two classifications v^* and v are identical, i.e. when the proposed classification is correct. The loss function then computes the negative of the number of blocks that are surrounded by blocks of the same class, which is the same up to an additive constant as the number of blocks that have at least one neighbor with a different class. Although this is not the Potts prior, it is clearly similar in spirit. Note however, the strange occurrence that even the correct classification is penalized according to its nature, which renders the loss function unusual.

2.2. Gaussian mixture models

The only thing missing above is the probability of a block b given its class, $\Pr(I_b|v_b)$. Here we assume that this is given by

$$\Pr(I_b|v_b) = A(I)\Pr(F(I_b)|v_b), \tag{6}$$

where F is a feature map from the set of possible block subimages to some feature space $\mathscr{F}$, and $A(I)$ is class-independent. This assumption is equivalent to assuming that

$$\Pr(v_b|I_b) = C(I)\Pr(v_b|F(I_b)), \tag{7}$$

where $C(I)$ is also class-independent, and thus that the features are sufficient for MAP estimation. Henceforth we drop the class-independent factors.

Given a feature map, F, the likelihood of the data in a block given its class will be described by a GMM. Denoting the feature vector associated to a block by $\vec{x}$, the probability of $\vec{x}$ is given by

$$\begin{aligned}\Pr(\vec{x}|\{p_i, \vec{\mu}_i, \Sigma_i\}) &= \sum_{i=1}^{M} p_i G(\vec{x}, \vec{\mu}_i, \Sigma_i) &(8)\\ &= \sum_{i=1}^{M} p_i b_i(\vec{x}), &(9)\end{aligned}$$

where $b_i(\vec{x}) = G(\vec{x}, \vec{\mu}_i, \Sigma_i)$ and $G(\vec{x}, \vec{\mu}, \Sigma)$ is a Gaussian of mean $\vec{\mu}$ and covariance Σ. The parameters for a given class are thus $\{p_i, \vec{\mu}_i, \Sigma_i | i \in M\}$. We have not indicated the class explicitly.

In order to calculate the value of $\Pr(\vec{x}|v_b)$, we should integrate over the parameters of this model using their prior probability, which will be given by the posterior probability of the parameters given training data. If we assume that this probability is very sharply peaked about its maximum, we can simply MAP estimate the parameters from the training

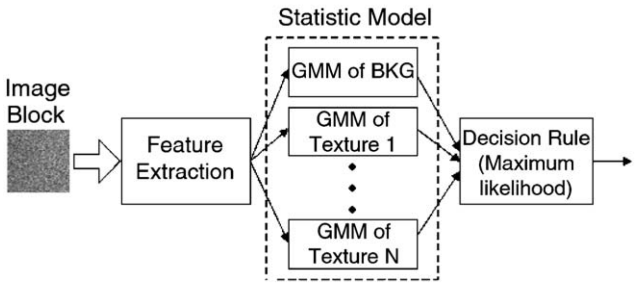

Fig. 1. Block diagram of texture classification algorithm using GMM.

data and substitute into the above likelihood to obtain the integral.

Fig. 1 shows the block diagram of the classification that is applied to an image block by using the GMMs for modelling the classes. First, a feature extraction is applied on the textured block. Then, each texture model emits the likelihood that the features of the texture block were generated by the GMM that models the texture class. Finally, those likelihoods are used for the final decision.

A GMM is a natural model to use if a class contains a number of distinct subclasses, as is often the case (for example, forest textures of different types of trees in an aerial image). An alternative to GMMs would then seem to be to treat the components of the GMMs for each block as hidden states (effectively, 'subclasses'), and to couple them to each other spatially, thus producing a 2D HMM. While this might seem like a good idea, the parameter estimation problem becomes much harder, both algorithmically and because there are more parameters to be estimated from the same limited amount of data. In addition, since the semantics of the hidden states is a priori unknown, it is not clear that the dependencies introduced by the HMM are present. We choose to adopt the simpler approach, to be confirmed a posteriori by comparison with the HMM alternative as used, for example, by Li and Gray [28].

To apply the above classification procedure, we must learn the parameters of the GMM models given a training set consisting of the data from T blocks of a single class, $X = \{\vec{x}_t | t \in T\}$. We will use maximum-likelihood estimation for this purpose. As is well-known, local maxima of the likelihood for GMMs can be found using the EM algorithm [36].

2.2.1. Training procedure and the BKG class

In addition to the classes that we wish to classify, we introduce the background ('BKG') class. Its parameters are learned from the blocks in the union of the training sets of each class. The BKG model has two roles. First, it is used to initialize the training of the individual texture models, thus ensuring that the initialization is the same for all classes and not biased towards any one. Second, the BKG model is used as a 'no decision' class. If the BKG model is more likely than any of the individual classes, then no decision is made.

Although in principle the EM algorithm could be used to train the BKG model, it becomes computationally expensive for the larger amounts of data involved. Experimentally we found that using a k-means algorithm instead reduced this computational expense considerably. Although the likelihood value attained was lower than that achieved with EM, for the amount of data involved in training the BKG model the difference was not too large. Coupled with the nature of the BKG class as a 'no decision' and initialization class, a nature that does not justify a large computational expense, this led us to use the k-means algorithm to train the BKG class.

2.3. Feature spaces

We have not yet defined the feature map F, which takes the space of $S \times S$ images to a feature space $\mathscr{F}$, but first we must choose S and T, the sizes of a block and a patch. For segmentation purposes, there is a trade-off between our ability to discriminate classes and the accuracy of boundary estimation. However, for retrieval purposes, the accuracy of texture boundaries is less critical. We choose a block size of $S = 16$, because the results from our experiences showed that it is large enough to capture a reasonable sample of the largest structures in the textures in the images with which we are dealing. The block size, should ideally be adaptive to the texture pattern and to the image resolution such that it will change for different textures/resolutions of the images, however in this work we fixed the block size. Choosing patch size is equivalent to choosing a degree of smoothing for the classifications: there is a tendency for blocks near the center of a given patch to be assigned the same class, since their corresponding patches have many blocks in common. We choose square patches containing nine blocks.

We divide the feature spaces into two subspaces: structure features and color features. Structure features are designed to capture spatial regularity of the image over the block, and will be extracted from the intensity images alone. Color features will use the full RGB information. Practically speaking, color provides an extremely powerful cue for the distinguishing of different classes in the scene.

2.3.1. Structure features

For the structure features we used three main kind of features: 2D Autoregressive model, energies in different wavelet subbands and energies of DCT regions. The reason for using those kind of features is because they have been proved to be efficient in texture analysis and because they can be applied on small blocks of texture. In general, the only restriction on the feature extraction used as an input to the GMM is that it will extract the information from small blocks of image without the need of the image surround the block.

The Auto-Regressive features are the coefficients estimated by a least-squares estimator of nonsymmetric

Table 1
Different structure features used with the GMM

Name of the feature	Description	Dimension
AR with mean	2D Auto-Regressive coefficients of order 1, 1	$4+1$
AR with mean	2D Auto-Regressive coefficients of order 2, 2	$12+1$
AR with mean with variance	2D Auto-Regressive model of order 1, 1 and its variance	$4+1+1$
AR with mean with variance	2D Auto-Regressive model of order 2, 2 and its variance	$12+1+1$
Wavelet Haar scale 3	Energies in Haar-wavelet subbands of level 3	10
Wavelet Haar scale 4	Energies in Haar-wavelet subbands of level 4	13
Wavelet Daubechies4 scale 4	Energies in Daubechies4-wavelet subbands of level 4	13
Wavelet biorthogonal-1 scale 4	Energies in biorthogonal-1 wavelet subbands of level 4	13
Wavelet biorthogonal-2 scale 4	Energies in biorthogonal-4 wavelet subbands of level 4	13
DCT scale 3	DCT like wavelet of scale 3	10
DCT scale 4	DCT like wavelet of scale 4	13

half-plane directly from the image data. In Table 1 we presents two model orders (1, 1) and (2, 2) which consists 4 and 12 coefficients, respectively. In addition to the AR coefficients we added the mean and the variance of the image block because this information is not represented by the AR coefficients.

The second kind of features are based on the energies in different wavelet subbands. We evaluated four diffident wavelet decomposition: Haar, Daubechies-4, and two kind of biorthogonal spline wavelets. More details about those wavelet decompositions can be found in Ref. [37]. For each wavelet we evaluated several levels of decomposition which define the number of subbands.

The last kind of features is the energies of DCT coefficients in regions of frequency space corresponding to a wavelet decomposition. This set of features (which are similar to the features used in Refs. [28,38]) are defined as follows, for a $2^s \times 2^s$ block: $f_{00} = D_{0,0}$ and, for $m \in [1, \ldots, s]$, by

$$f_{m0} = \sum_{i=2^{m-1}}^{2^m-1} \sum_{j=0}^{2^{m-1}-1} |D_{i,j}|, \tag{10a}$$

$$f_{0m} = \sum_{i=0}^{2^{m-1}-1} \sum_{j=2^{m-1}}^{2^m-1} |D_{i,j}|, \tag{10b}$$

$$f_{mm} = \sum_{i=2^{m-1}}^{2^m-1} \sum_{j=2^{m-1}}^{2^m-1} |D_{i,j}|, \tag{10c}$$

where $\{D_{i,j} : i, j \in \{0, \ldots, 2^s\}\}$ are the DCT coefficients of the block and s is the scale of the feature extraction. This is illustrated in Fig. 2.

2.3.2. Color features

As color features, we used the data mean and covariance over a block of the RGB and LAB values which is a nonlinear transformation of RGB. Since the covariance matrix is symmetric, only half of it, including the diagonal, is included in the feature vector. The color feature vector is thus a 9-dimensional vector, 3 coming from the mean and 6 from the covariance (Tables 2 and (3)).

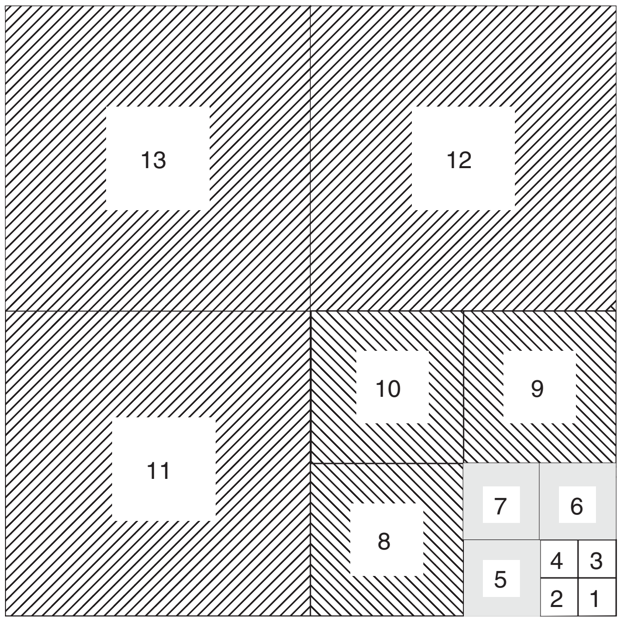

Fig. 2. 'Wavelet-like' DCT features for a 16 × 16 block. Each feature component is the sum of the square magnitudes of the DCT coefficients in the appropriate sub-block. The sub-block are of 4 different scales, 1 × 1, 2 × 2, 3 × 3, 4 × 4, hence we call it DCT scale 4.

2.3.3. Combining color and structure

Since we have both color and structure features available, we have to combine this information. We experimented with five different ways of doing this: three ways are decision-level fusion based on defining a confidence value for the classifications and the other two ways are at the feature-based level by combining the structure feature vector and the color feature vector into one feature vector.

The first method works as follows. For each block, and for both structure and color features, we compute the right-hand side of Eq. (5) for all classes. Let $n_1(b)$ and $n_2(b)$ be

Table 2
Different color features used with the GMM

Name of the feature	Description	Dimension
RGB mean	Mean of the R, G, B over block 16×16 pixels	3
RGB mean and cov.	RGB mean and cov. matrix over block	9
LAB mean	Mean of the L, A, B over block of 16×16 pixels	3
LAB mean and cov.	LAB mean and cov. matrix over block	9

Table 3
Classification accuracies (percentage) of the full and diagonal covariance GMMs using color (RGB mean and cov.) and structure features (DCT of scale 4)

	Color	Structure
Diagonal cov. GMM	71.2	90.9
Full cov. GMM	90.6	91.2

the maximizing and next-to-maximizing classes for block b. We define the 'confidence' of the classification decision by

$$C_1(b) = \Pr(I_b|v_b = n_1(b))/\Pr(I_b|v_b = n_2(b)). \tag{11}$$

If the classifications resulting from using the structure features and the color features conflict, we choose the decision with the highest confidence.

The second method is similar to the first one just that the confidence value is the ratio between the maximum likelihood and the likelihood of the Background class, as given by

$$C_2(b) = \Pr(I_b|v_b = n_1(b))/\Pr(I_b|v_b = bkg). \tag{12}$$

The confidence measure used above can be replaced by the negative entropy of the posterior distribution $\Pr(v(b)=n|I_b)$. This is the third method.

The fourth and fifth methods are different from the previous methods because the fusion is done already at the feature extraction stage. The idea of the forth method is to create one feature vector by concatenating the structure feature to the color feature. The fifth method idea is to apply the structure extraction to each component of color. The advantage of those methods is that the statistical model take into account the dependencies between the structure feature and the color feature. However, it suffers from "course-of-dimensionality" because the dimension of feature vector is larger then the dimension of the color and structure feature. Therefore, the number of unknown parameters of the model is much larger and there is the need for much more data in order to estimate the parameters.

As discovered, at least for the experiments reported here the fusion based on the confidence value that is the difference between the maximum-likelihood and the next to maximum-likelihood results with the best performances.

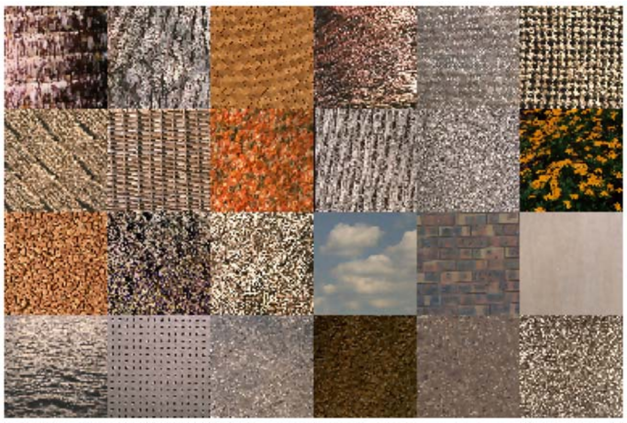
Fig. 3. Textures from the Vistex database: Bark0; Bark12; Fabric0; Fabric4; Fabric7; Fabric8; Fabric11; Fabric13; Fabric15; Fabric17; Fabric19; flowers0; Food0; Leaves12; Grass1; Clouds0; Brick0; Wood2; water0; Tile7; Stone4; Sand0; Misc2; Metal0.

3. Experimental results

We used several databases in order to evaluate the GMM classification performances, to study more about the GMM properties and to compare the GMM performances with other statistical models. The experiments were conducted on the MIT Vision Texture (VisTex) database, on mosaic images created from the VisTex database, and on the aerial images of the San Francisco Bay area that were used in Refs. [28,38–40].

3.1. Vistex texture database

We chose randomly 24, 512×512 textured color images from the Vistex database. The textures are displayed in Fig. 3. The feature extraction was done on blocks of size 16, with overlap of 8 pixels. The patch size for classification was 3. Each image was divided into subimages of size 32×32 pixels, corresponding to one patch. All blocks extracted from the first 96 subimages of each texture were used for training, while the remaining 160 subimages were used for testing.

For each class we trained a GMM with five components using 30 iterations of the EM algorithm. We chose five components because increasing the number of components did not improve the results significantly. We used 30 iterations for a similar reason: the EM algorithm appeared to have

Table 4
Classification results (in percents) on 24 textures from the VisTex database

Index	Texture	Structure	Color	Both
1	Bark0	99	96	100
2	Bark12	66	91	94
3	Fabric0	100	100	100
4	Fabric4	69	86	89
5	Fabric7	99	93	99
6	Fabric8	96	99	99
7	Fabric11	83	98	96
8	Fabric13	93	93	99
9	Fabric15	98	100	100
10	Fabric17	98	88	97
11	Fabric19	99	94	99
12	Flowers0	91	98	98
13	Food0	98	100	100
14	Leaves12	94	28	93
15	Grass1	94	95	98
16	Clouds0	100	99	100
17	Brick0	93	96	96
18	Wood2	100	100	100
19	Water0	41	88	84
20	Tile7	99	65	88
21	Stone4	98	71	91
22	Sand0	99	100	100
23	Misc2	95	100	100
24	Metal0	97	98	99
		Avg.structure	Avg.color	Avg.both
		91.2	90.6	96.6

converged after this number of iterations. We used the same initialization for each texture class: the BKG model. The results of the classification using the color features (RGB mean with covariance), the structure features (DCT scale 4) and the combined decision by using confidence value based on second to maximum are shown in Table 4.

3.2. Choosing the feature extraction

Even though the main focus in this paper is the use of GMM for texture classification, an important question is which feature extraction is the most appropriate to be used as the input for the GMM. As mentioned in Section 2.3.1 we have evaluated the performances of several kind of feature extraction for the structure information including the energies in different wavelet subbands, coefficient of 2D Autoregressive , and the DCT coefficients in regions of frequency space corresponding to a wavelet decomposition. The performance of the classification by using difference feature extraction on the Vistex database are given in Table 5. As can be seen from the table the DCT-like wavelet perform a lightly better classification than the other feature extraction.

For the 'color' feature we evaluate the mean and the covariance of the three components color in the RGB and LAB color space. As can be seen from Table 6 the classification performance of LAB and RGB space is very close to each other. However, as shown in the next sub-section, we found that the fusion of classifications based on the features RGB- and DCT-like wavelet gave the best results.

Table 5
Texture classification performance using structure features on Vistex database

Name of the feature	Performance
2D AR of order 1, 1 with mean	80.9
2D AR of order 2, 2 with mean	86.0
2D AR of order 1, 1 with mean with variance	89.6
2D AR of order 2, 2 with mean with variance	90.4
Wavelet Haar scale 3	87.2
Wavelet Haar scale 4	90.1
Wavelet Daubechies scale 4	89.5
Wavelet biorthogonal-1 scale 4	88.6
Wavelet biorthogonal-2 scale 4	86.0
DCT scale 3	89.8
DCT scale 4	**91.2**

The bold numbers mark the best performance.

Table 6
Texture classification performance using color features on Vistex database

Name of the feature	Performance
RGB mean	74.7
RGB mean and cov.	**90.6**
LAB mean	77.0
LAB mean and cov.	**90.8**

The bold numbers mark the best performance.

3.3. Fusing the 'structure' and 'color' information

As explained in Section 2.3.3 we applied five methods for fusing the 'structure' and 'color' information in order to improve the performance of classification. Three of the methods are based on defining a confidence value for the classifications based on 'structure' and 'color' features. Table 7 presents the main results of fusing the 'structure' and 'color' decision on Vistex database by using confidence values methods. As can be seen from Table 7, fusing the decisions improve the performances. However, the fusion method that is based on the difference between the maximum likelihood and the 2nd maximum likelihood had the best performances especially when using the RGB mean and covariance as the color features and the DCT of 4th scale as the structure features.

Another interested experience is combining the 'structure' and 'color' features into one feature vector and using it as the input feature for the GMM. We used two ways in order to combine the 'structure' and the 'color' features. The first way was concatenating the two vectors in one vector and the second way was applying the 'structure' feature extraction on each color component R G B. The results of this experience are shown in Table 8. Comparing those results to the results achieved by fusing the 'color' and the 'structure' classification by the confidence value, we can notice that

Table 7
Texture classification performance after fusing color and structure features at the decision level by using confidence value that is based on 2nd maximum likelihood, on the BKG class likelihood, and on the entropy of the posterior distribution

Color	Structure					
	Second maximum likelihood		BKG likelihood		Entropy of the posterior	
	AR	DCT scale 4	AR	DCT scale 4	AR	DCT scale 4
RGB mean and cov.	90.8	**96.6**	90.9	92.0	90.4	92.3
LAB mean and cov.	94.5	91.9	92.9	91.9	91.5	92.2

The bold numbers mark the best performance.

Table 8
Texture classification performances of combined colored and structured featured

The feature	Performances
Concatenating the DCT-like wavelet scale 3 and the mean of RGB	86.9
Concatenating the DCT-like wavelet scale 4 and the mean of RGB	87.2
DCT scale 3 on each component color R, G and B	90.8
DCT scale 4 on each component color R, G and B	79.6

fusing by confidence value is preferable. The main reason for that is that high dimension of the feature vector cause to ‘course of dimensions’. As the feature vector has a higher dimension the GMM has more parameters that needs to be trained and hence there is a need for much more training data. The ‘course of dimension’ is the most obvious in the case that the feature extraction is DCT-like wavelet of scale 4 and is applied to each component color R, G and B. In such a case the dimension of the feature vector is $13 \times 3 = 39$ and therefore each Gaussian covariance matrix consists 780 unknown parameters which is an enormous number of unknown parameters.

3.4. Full or diagonal covariance?

In the experiments described above, we used an unrestricted covariance for the GMMs because the different features could be arbitrarily correlated. It is also possible to restrict the GMM to have diagonal covariance. The use of a diagonal covariance has two practical advantages:

(1) There are less parameters to be estimated.
(2) Calculation of the inverse matrix is trivial.

Because of these advantages, there are applications, for example speaker verification, in which a diagonal covariance is used [41]. The obvious disadvantage of assuming diagonality is that the different features may in practice be strongly correlated, in which case the model will be incapable of representing the feature distribution accurately.

Fig. 4 illustrates the point. We chose randomly two features from the structure features ‘DCT of scale 4’, and two features from the color features ‘RGB cov. and mean’. The first row of the figure shows scatter plots of these two pairs of features for a sample of the ‘Bark0’ texture. The second row shows typical isoprobability contours from each of the components of the full covariance GMMs trained on these two feature pairs, while the third row shows the same contours for the corresponding diagonal covariance GMMs. Each GMM has five components. For the diagonal covariance GMM, the axes of the isoprobability contours, which are ellipses, are constrained to be parallel to the feature axes, which is not the case for the full covariance models.

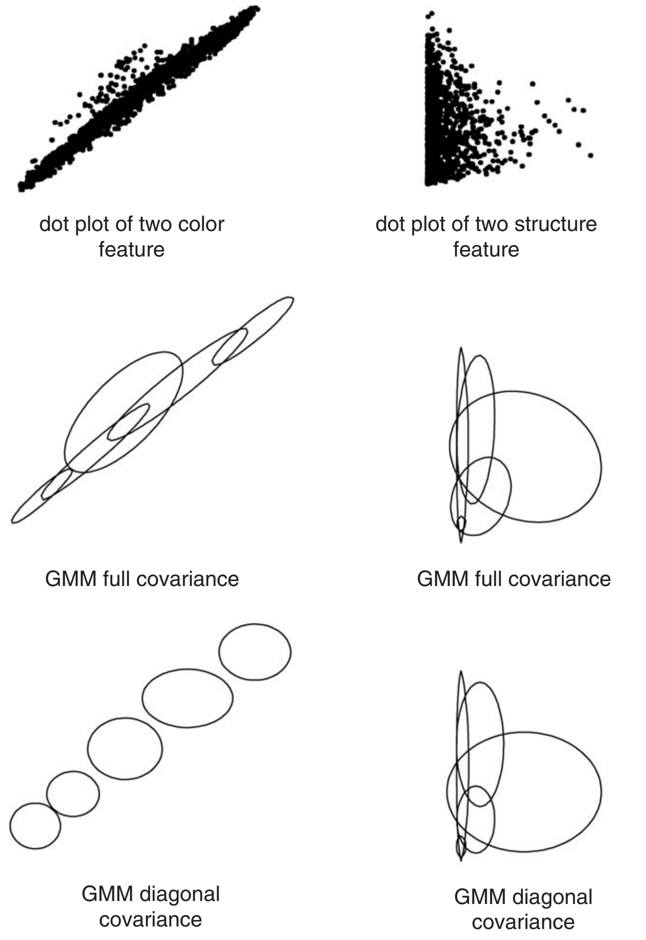

Fig. 4. Modelling the distribution of two structure features and two color features using a full covariance GMM (second row) and a diagonal covariance GMM (third row).

Table 9
Estimated probability of correct classification as a function of the selected order of the GMM, using structure features (DCT of scale 4) and color features (RGB mean and cov.) on the Vistex database

GMM order	1	2	3	4	**5**	6	7	8	9	10
'structure' P_{correct}	0.563	0.889	0.895	0.903	**0.912**	0.906	0.904	0.902	0.898	0.896
'color' P_{correct}	0.702	0.882	0.902	0.904	**0.906**	0.902	0.898	0.897	0.898	0.895

The bold numbers mark the best performance.

In Fig. 4, we see that the color features we selected are strongly dependent, and that therefore the GMM must use a full covariance matrix in order to model the color feature distribution correctly. For the structure features, the difference is far less clear, and it seems that a diagonal covariance matrix is sufficient. An evaluation of the two types of model in the Vistex experiments is presented in Table 3. There is a large decrease in the classification accuracy when diagonality is enforced for the color features, while for the structure features the decrease is small.

3.5. Choosing the number of GMM components

An important issue in mixture modelling is the selection of the number of components. The usual tradeoff in model order selection arises: with too many components, the mixture may over-fit the data, while a mixture with too few components may not be flexible enough to approximate the true underlying model. There are many algorithms that discuss how to select the model order, for example Refs. [42,43]. Selecting the number of components in the GMM is crucial for unsupervised segmentation tasks in which each component represents a separate class. In our case however, when each GMM models a single texture class and segmentation is supervised, selecting the number of components becomes less important. It can be seen from Table 9 that the results of the classification are very similar for all model orders greater than one (Selecting the GMM order to be one means that we are using a multivariate Gaussian.) and for both the color and structure features the GMM with five Gaussians has the best classification performances.

3.6. Classification at the boundary of two different textures

By using the Vistex database we have created several mosaic images in order to illustrate the results of the classification at the boundary of the textures. Fig. 5 shows the results of classifying a mosaic image with straight lines boundaries while Fig. 6 shows the results of classifying a mosaic with a round boundary. In both figures we can clearly see that the error rate is much higher at the boundary of the textures then at the center of the textures. However, the general shape of the boundary can be recognized even after the classification. The reason of the higher misclassification at the boundary is due to the fact the GMM algorithm is based on the assumption that there is a patch of 3×3 blocks that contain

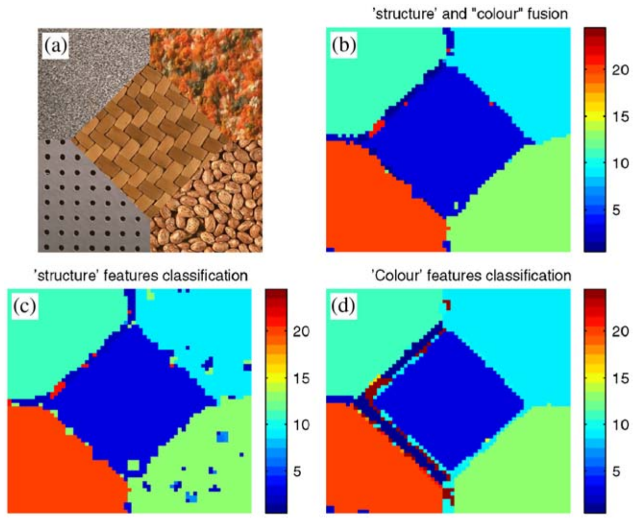

Fig. 5. Mosaic image and its classification by using color and structure features. Subplot (a) is the Mosaic image created from the Vistex database. Subplot (c) and (d) are the classification results by using the 'structure' and 'color' feature, respectively. Subplot (b) is the classification results after the fusion process. The color bar on the right of subplots (b–d) represents the index of the 24 textures that exists in our database.

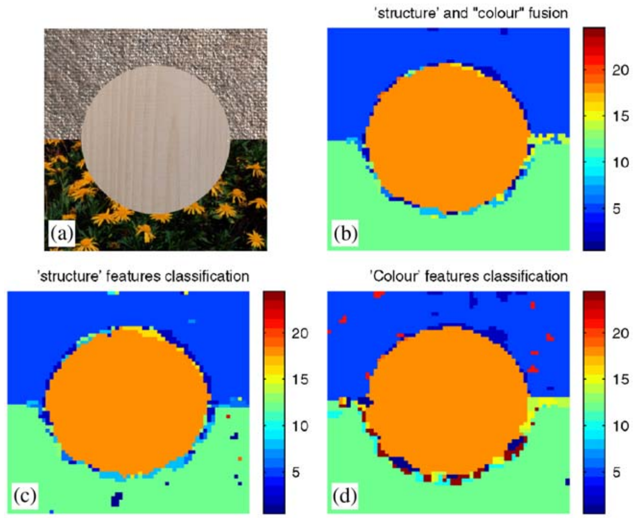

Fig. 6. Mosaic image and its classification by using color and structure features. Subplot (a) is the Mosaic image created from the Vistex database. Subplot (c) and (d) are the classification results by using the 'structure' and 'color' feature, respectively. Subplot (b) is the classification results after the fusion process. The color bar on the right of subplots (b–d) represents the index of the 24 textures that exists in our database.

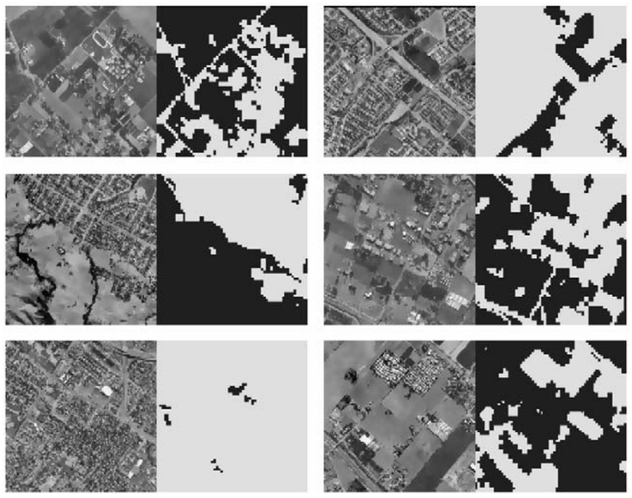

Fig. 7. Aerial images. On the left of each pair, the original images. On the right, the manual segmentations. The dark areas are natural, the lighter areas, man-made.

the same texture. This assumption is false at the boundary between two texture, however if most of the blocks in the patch are from the same texture there is a high chance that the algorithm will still work. The error at the boundary was our main reason for choosing a small patch even though choosing a bigger patch would increase the performance in the center of the texture.

Part (c) and (d) in Figs. 5 and 6 show the classification results when using the 'structure' and 'color' feature, respectively. Part (b) shows the classification results after the fusion by using the confidence value based on the difference of the ML and the second ML as described in Section 2.3.3. It can be seen from both figures that some of the textures are better classified by the 'structure' features while others by the 'color' features. For instance, the textures on the right side of the image in Fig. 5 (food and fabric13) have dominant colors components and therefore is better classified by the 'color' features. On the other hand, the texture on the left-button side (tile7) in the same image has a very unique structure and therefore is better classified by the 'structure' features. From both figures it can be seen very clearly that the fusion of the classifications improves the classification both at the edges and at the center of the textures.

3.7. Aerial images

This database includes six 512×512 gray-scale images with manual segmentations of the images into man-made and natural areas. We used the manual segmentations as ground truth for evaluating the algorithm. The images are displayed in Fig. 7.

We used this database for evaluation exactly as it was used in Refs. [38,28]. For each iteration, one image was used as test data, and the other five were used as training data. Overall performances are evaluated by averaging over all iterations. Each class ('man-made' and 'natural') was modelled by a five-component GMM of the structure features only, since clearly there is no color information. For initialization we used the BKG model.

The results from the GMM algorithm were compared to the results from other statistical models reported in Refs. [28,38]: the two-dimensional hidden Markov model (2D HMM) [28]; the two-dimensional multi-resolution hidden Markov model (2D MHMM) [38]; CART (a decision tree algorithm) [44]; and LVQ1 (version 1 of Kohonen's learning vector quantization) [45]. The classification performance for each test image in the six-fold cross-validation and the average performance rates are listed in Table 10. The results in Table 10 indicate that for this specific experiment the GMM model perform slightly better then the MHMM and HMM and superior to the CART and LVQ algorithm. This result is significant because the complexity of the GMM is much smaller then the MHNN and the HMM model.

By comparing the average classification of the aerial images with the average classification of the Vistex database (Table 4) we notice that the performances of the Vistex database are better then those in the aerial photos. The main reason for the difference is due to the fact that the aerial images contain much more boundaries between textures then the Vistex database. As we saw from the experiments of the mosaic images in Section 3.6, the error rate is much higher at the boundary of two textures then in the areas that there is only one kind of texture.

4. Conclusion

We have described Gaussian mixture models (GMMs) of texture and color features. We have evaluated the

Table 10
Classification performances (correct rates) by algorithm

Iteration	CART	LVQ1	HMM	MHMM	GMM
1	77.4	78.4	81.0	82.7	83.6
2	92.0	80.9	82.4	83.7	86.0
3	71.1	71.6	79.7	72.2	80.4
4	74.8	75.1	76.0	79.5	80.9
5	85.8	81.3	81.7	87.5	95.8
6	79.8	81.9	86.7	88.5	84.6
Ave.	78.5	78.2	81.2	84.0	85.2

classificationperformances variety of 'color' and 'structure' features and found which are the most appropriate one. In addition we suggested several methods for combining the 'color' and 'structure' information and analyzed the influence of the model selection of the GMM and the influence of using a diagonal covariance versus the full covariance of the Gaussian. The advantage of using the GMMs presented here, is that they provide improved performance over other existing methods, while requiring only modest computational requirements.

5. Summary

This work defines Gaussian mixture models (GMM) of colored texture on several feature spaces and compares the performance of these models in various classification tasks, both with each other and with other models popular in the literature. The work evaluates the classification performances on variety of 'color' and 'structure' features and finds which are the most appropriate one. In addition the work suggests several methods for combining the 'color' and 'structure' information and analyzes the influence of the model selection of the GMM and the influence of using a diagonal covariance versus the full covariance of the Gaussian. The advantage of using the Gaussian mixture models presented in the work, is that they provide improved performance over other existing methods, while requiring only modest computational requirements.

Acknowledgments

This work was partially supported by EU project MOUMIR (HP-99-108, http://www.moumir.org). The Authors would like to thank Prof. R.M. Gray and Prof. J. Li for providing the ariel images database that is used in the experiments section. They also wish to thank the reviewers for their helpful comments and suggestions.

References

[1] T. Reed, J.M.H. du Buf, A recent review of texture segmentation and feature extraction techniques, CVGIP Image Understanding 57 (1993) 359–372.
[2] M. Tuceryan, A. Jain, Handbook of Pattern Recognition and Computer Vision, World Scientific, Singapore, 1993.
[3] M. Unser, Texture classification and segmentation using wavelet frame, IEEE Trans. Image Process. 4 (11) (1995) 1549–1560.
[4] T.P. Weldon, W.E. Higgins, Design of multiple gabor filter for texture segmentation, in: Proceedings of the International Conference on Acoustics, Speech and Signal Processes, 1996, pp. 2243–2246.
[5] T.P. Weldon, W.E. Higgins, Integrated approach to texture segmentation using multiple gabor filters, in: Proceedings of the International Conference on Image Processing, 1996, pp. 955–958.
[6] M. Unser, Local linear transforms for texture measurements, Signal Processing 11 (1986) 61–79.
[7] J.S. Weszka, C.R. Dyer, A. Rosenfeld, A comparative study of texture measures for terrain classification, IEEE Trans. Systems Man Cybernet. 6 (1976) 376–380.
[8] N. Saito, R.R. Coifman, Local discriminator bases and their application, J. Math. Image Vis. 5 (4) (1995) 337–358.
[9] J. Strand, T. Taxt, Local frequency features for texture classification, Pattern Recognition 27 (10) (1994) 1397–1406.
[10] T. Ojala, M. Pietikäinen, D. Harwood, A comparative study of texture measures with classification based on feature distributions, Pattern Recognition 29 (1) (1996) 51–59.
[11] P.P. Ohanian, R.C. Dubes, Performance evaluation of four classes of textural features, Pattern Recognition 25 (8) (1992) 819–833.
[12] S. Singh, M. Markou, J. Haddon, Nearest neighbour classifiers in natural scene analysis, Pattern Recognition 34 (8) (2001) 1601–1612.
[13] T. Randen, J.H. Husoy, Multichannel filtering for image texture segmentation, Opt. Eng. 33 (1994) 2617–2625.
[14] T. Randen, J.H. Husoy, Filtering for texture classification: a comparative study, IEEE Trans. Pattern Anal. Mach. Intell. 21 (4) (1999) 291–310.
[15] F. Farrokhnia, Multi-channel filtering techniques for texture segmentation and surface quality inspection, Ph.D. Thesis, Michigan State University, 1990.
[16] S. Li, J.T. Kwok, H. Zhu, Y. Wang, Texture Classification Using the Support Vector Machines, 2003.
[17] K. Kim, K. Jung, S. Park, H. Kim, Support vector machines for texture classification, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 1542–1550.
[18] M. Pietikäinen, T. Nurmela, T. Mäenpää, M. Turtinen, View-based recognition of real-world textures, Pattern Recognition 37 (2) (2004) 313–323.
[19] M.M. Pietikäinen, A. Rosenfeld, L.S. Davis, Experiments with texture classification using averages local pattern matches, IEEE Trans. Systems Man Cybernet. 13 (3) (1983) 421–426.
[20] A. Teuner, O. Pichler, B.J. Hosticka, Unsupervised texture segmentation of images using tuned matched gabor filters, IEEE Trans. Image Process. 4 (1995) 863–870.
[21] T. Randen, J.H. Husoy, Texture segmentation using filters with optimized energy separation, IEEE Trans. Image Process. 8(4).
[22] A. Mahalanobis, H. Singh, Application of correlation filters for texture recognition, Appl. Opt. 33 (11) (1994) 2173–2179.
[23] A.C. Bovik, M. Clark, W.S. Geisler, Multichannel texture analysis using localized spatial filters, IEEE Trans. Pattern Anal. Mach. Intell. 12 (1990) 55–73.
[24] T. Randen, J.H. Husoy, Texture segmentation with optimal linear prediction error filters, Piksel'n 11 (1994) 25–28.
[25] T. Randen, Filter and filter bank design for image texture recognition, Ph.D. Thesis, Norwegian University of Science and Technology, 1997.
[26] G.R. Cross, A.K. Jain, Markov random field texture models, IEEE Trans. Pattern Anal. Mach. Intell. 5 (1983) 25–39.
[27] S. Geman, C. Graffigne, Markov random field image models and their applications to computer vision, in: Proceedings of the International Congress of Mathematicians 1986, American Mathematical Society, Providence, RI, 1987, pp. 1496–1517.
[28] J. Li, A. Najmi, R.M. Gray, Image classification by a two-dimensional hidden Markov model, IEEE Trans. Signal Process. 48 (2) (2000) 517–533.
[29] B.R. Povlow, S.M. Dunn, Texture classification using noncausal hidden Markov models, IEEE Trans. Pattern Anal. Mach. Intell. 17 (10) (1995) 1010–1014.
[30] M. Crouse, R. Nowak, R. Baraniuk, Wavelet-based statistical signal processing using hidden Markov models, IEEE Trans. Signal Process. 46 (4) (1998) 886–902.
[31] H. Choi, R.G. Baraniuk, Image segmentation using wavelet-domain classification, in: Proceedings of the SPIE Technical Conference on Mathematical Modeling, Bayesian Estimation, and Inverse Problems, 1999, pp. 306–320.

[32] G. Fan, X.-G. Xia, Maximum likelihood texture analysis and classification using wavelet-domain hidden Markov models, in: Proceedings of the 34th Asilomar Conference on Signals, Systems and Computers, 2000.

[33] C.W. Shaffrey, N.G. Kingsbury, I.H. Jermyn, Unsupervised image segmentation via Markov trees and complex wavelets, in: Proceedings of the IEEE International Conference on Image Processing, Rochester, USA, 2002.

[34] H. Permuter, J. Francos, I.H. Jermyn, Gaussian mixture models of texture and colour for image database retrieval, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Hong Kong, China, 2003.

[35] S.H. Yoon, C.S. Won, K. Pyun, R.M. Gray, Image classification using GMM with context information and reducing dimension for singular covariance, in: Proceedings of the IEEE Data Compression Conference (DCC), Snowbird, UT, 2003, p. 457.

[36] A. Dempster, N. Larid, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. Roy. Statist. Soc. 39 (1977) 1–38.

[37] I. Daubechies, Ten lectures on wavelets, in: CBMS-NSF Regional Conferences Series in Applied Mathematics, Society for Industrial and Applied Mathematics, 1992.

[38] J. Li, R. Gray, R. Olshen, Multiresolution image classification by hierarchical modeling with two dimensional hidden Markov models, IEEE Trans. Inform. Theory 46 (5) (2000) 1826–1841.

[39] K.L. Oehler, R. Gray, Combining image compression and classification using vector quantization, IEEE Trans. Pattern Anal. Mach. Intell. 17 (5) (1995) 461–473.

[40] K. Oehler, Image compression and classification using vector quantization, Ph.D. Thesis, Stanford University, 1993.

[41] D.A. Reynolds, T.F. Quatieri, R.B. Dunn, Speaker verification using adapted Gaussian mixture models, Digital Signal Process. 10 (2000) 19–41.

[42] M. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 381–396.

[43] S. Richardson, P. Green, On Bayesian analysis of mixture with unknown number of components (with discussion), J. Roy. Statist. Soc. 59 (1997) 731–792.

[44] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees, Chapman & Hall, London, 1984.

[45] T. Kohonen, Self-Organization and Associative Memory, Springer, Berlin, 1989.

About the Author—HAIM PERMUTER received his B.Sc. (summa cum laude) and M.Sc. (summa cum laude) degree in Electrical and Computer Engineering from the Ben-Gurion University, Israel, in 1997 and 2003, respectively. Between 1997 and 2004, he was an officer at a research and development unit of the Israeli Defense Forces. He is currently pursuing his Ph.D. degree in Electrical Engineering at the Stanford University, CA. He is a recipient of both the Fullbright Fellowship and the Stanford Graduate Fellowship (SGF).

About the Author—JOSEPH FRANCOS received his B.Sc. degree in Computer Engineering in 1982, and his D.Sc. degree in Electrical Engineering in 1991, both from the Technion-Israel Institute of Technology. In 1993 he joined the ECE Department, Ben-Gurion University, where he is now an Associate Professor. His current research interests are in parametric modelling and estimation of 2-D random fields, random fields theory, image registration, and texture analysis and synthesis. Dr. Francos served as an Associate Editor for the IEEE Transactions on Signal Processing from 1999 to 2001.

About the Author—IAN JERMYN received his B.A. (Physics) from the Oxford University (1986), his Ph.D. (Theoretical Physics) from the Manchester University (1991), and his Ph.D. (Computer Science) from the Courant Institute (2000). He is currently a Senior Research Scientist in the Ariana group at INRIA. His research interests include shape and texture modelling.