Multiscale Image Segmentation using Wavelet-Domain Hidden Markov Models

Hyeokho Choi and Richard G. Baraniuk *

Department of Electrical and Computer Engineering, Rice University

6100 South Main Street, Houston, TX 77005–1892

Email: {choi,richb}@ece.rice.edu, Web: www.dsp.rice.edu, Fax: 713.737.6196

Submitted to *IEEE Transactions on Image Processing*, October 1999

EDICS Numbers: 2-SEGM (Segmentation), 2-WAVP (Wavelets and Multiresolution Processing)

Abstract— We introduce a new image texture segmentation algorithm, HMTseg, based on wavelets and the hidden Markov tree (HMT) model. The HMT is a tree-structured probabilistic graph that captures the statistical properties of the coefficients of the wavelet transform. Since the HMT is particularly well suited to images containing singularities (edges and ridges), it provides a good classifier for distinguishing between textures. Utilizing the inherent tree structure of the wavelet HMT and its fast training and likelihood computation algorithms, we perform multiscale texture classification at a range of different scales. We then fuse these multiscale classifications using a Bayesian probabilistic graph to obtain reliable final segmentations. Since HMTseg works on the wavelet transform of the image, it can directly segment wavelet-compressed images without the need for decompression into the space domain. We demonstrate the performance of HMTseg with synthetic, aerial photo, and document image segmentations.

*This work was supported by NSF grants MIP–9457438 and CCR–9973188, DARPA/AFOSR grant F49620–97–1–0513, ONR grant N00014–99–1–0813, and the Texas Instruments Leadership University Program.

1 Introduction

1.1 Image segmentation

An image segmentation algorithm aims to assign a *class label* to each pixel of an image based on the properties of the pixel and its relationship with its neighbors. A "good" segmentation separates an image into simple regions with homogeneous properties, each with a different "texture" [1].

Recently, many authors have applied Bayesian statistical techniques to jointly estimate the region shapes and determine their classes [2–4].[1] Bayesian techniques regard a sampled image $\boldsymbol{x}$ as a realization of a random field $\boldsymbol{X}$ with distinct and consistent stochastic behaviour in different regions. In an image subregion $\boldsymbol{X}_r \subset \boldsymbol{X}$ of class c, the pixels are assumed distributed with joint probability density function (pdf) $f(\boldsymbol{x}_r|c)$. In these terms, the image segmentation problem can be rephrased as: given an image $\boldsymbol{x}$, estimate for each pixel a class label $c \in \{1, 2, \ldots, N_c\}$. The *labeling field* $\boldsymbol{C}$ records the class label of each pixel. Maximum likelihood (ML) segmentation partitions the image into subregions $\boldsymbol{x}_r$ that maximize the value of the likelihood $f(\boldsymbol{x}_r|c)$ over the regions. Maximum a posteriori (MAP) segmentation in addition weights the likelihoods by the prior probabilities of each c.

The two key ingredients to any segmentation scheme are: (1) a description of the possible image regions $\boldsymbol{x}_r$, and (2) a set of joint pixel pdfs $\{f(\boldsymbol{x}_r|c) : c = 1, 2, \ldots, N_c\}$.

The primary difficulty in image segmentation arises because there are simply too many possible region shapes, and it is intractable to specify the joint pixel pdf for each possibility. Moreover, even if the joint density could be specified for each possible region shape, the cost of computing the optimal ML or MAP segmentation would be prohibitive. In practice, we must impose structures on both the possible image regions and on the pixel pdfs.

1.2 Multiscale image segmentation

Many segmentation algorithms employ a *classification window* of some size in the hope that all pixels in the window will belong to the same class. A typical segmentation then consists of classifying each window of pixels followed by some post-processing.

[1] We denote deterministic quantities using small letters, random variables using capital letters, and vectors using boldface letters.

Clearly, the size of the classification window is crucial. A large window usually enhances the classification reliability (because many pixels provide rich statistical information) but simultaneously risks having pixels of different classes inside the window. Thus, a large window produces accurate segmentations in large, homogeneous regions but poor segmentations along the boundaries between regions. A small window reduces the possibility of having multiple classes in the window, but sacrifices classification reliability due to the paucity of statistical information. Thus, a small window is more appropriate near the boundaries between regions.

To capture the properties of each image region to be segmented, both the large and small scale behaviours should be utilized to properly segment both large, homogeneous regions and detailed boundary regions. In *multiscale segmentation*, we combine the results of many classification windows of different sizes.

In this paper, we will employ the *dyadic squares* (or blocks) to implement classification windows of different sizes. Given an initial $2^J \times 2^J$ square image $\boldsymbol{x}$ of $n := 2^{2J}$ pixels, the dyadic squares are obtained simply by recursively dividing the image into four square subimages of equal size (see Fig. 1(a)). Since the four "child" squares nest inside their "parent" square at the next coarser scale, the dyadic squares have a convenient quad-tree structure; each node in the quad tree in Fig. 1(b) corresponds to a dyadic square. Denote a dyadic square at scale j by $\boldsymbol{d}_i^j$ (with i an abstract index enumerating the squares at this scale). At the two extremes, $\boldsymbol{d}_0^0$ (root of the tree) is the entire image $\boldsymbol{x}$, and each $\boldsymbol{d}_i^J$ (leaf of the tree) is an individual pixel. Given a random field image $\boldsymbol{X}$, the dyadic squares are also random fields, denoted $\boldsymbol{D}_i^j$. In the sequel, when we speak of a generic square, we will often drop the j.

With this structure for representing regions, we will segment images by estimating the class label c for each dyadic square $\boldsymbol{d}_i$. This estimation requires a pixel pdf model for each class that is suited to the dyadic squares. Help is close at hand with the dyadic wavelet decomposition and wavelet-based statistical models.

1.3 Multiscale statistical models and wavelets

Models of different image textures play a fundamental rôle in image classification and segmentation, since the complete joint pixel pdf is typically overly complicated or unavailable in practice.

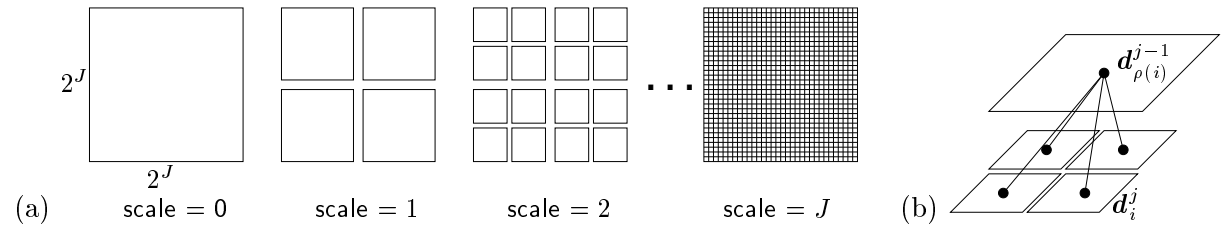


Figure 1: *(a) Image $\boldsymbol{x}$ divided into dyadic squares $\boldsymbol{d}^j_i$ at different scales. Each dyadic square can be associated with a subtree of Haar wavelet coefficients. (b) Quad-tree structure of dyadic squares. The dyadic square $\boldsymbol{d}^{j-1}_{\rho(i)}$ splits into four child squares at scale j.*

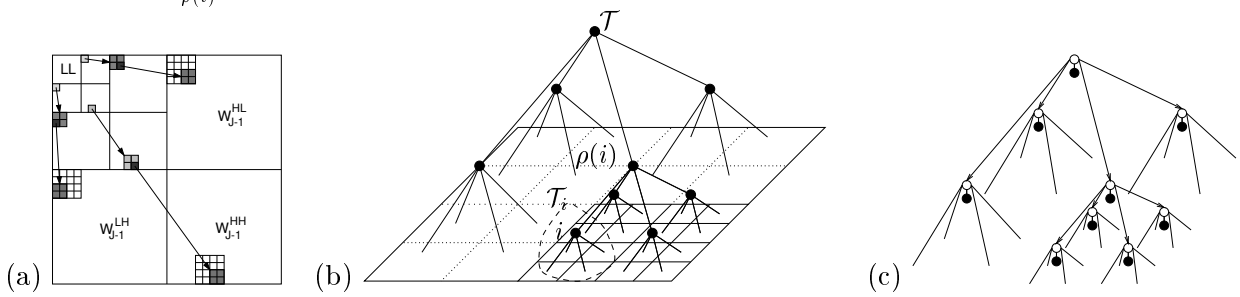


Figure 2: *(a) Parent-child dependencies of the three 2-D wavelet transform subbands: Each arrow points from a parent wavelet coefficient to its four children at the next finer scale. (b) More detailed view of the quad-tree structure in one subband. Each black node corresponds to a wavelet coefficient. The figure also illustrates our tree indexing notation: $\mathcal{T}_i$ is the subtree of coefficients rooted at node i, and $\rho(i)$ is the parent of node i. (c) 2-D wavelet hidden Markov tree (HMT) model. We model each wavelet coefficient (black node) as a Gaussian mixture controlled by a hidden state variable (white node). To capture the persistence across scale property of wavelet transforms (**W6**), we connect the states vertically across scale in Markov-1 chains.*

Transform-domain models are based on the idea that often a linear, invertible transform will "restructure" an image, leaving transform coefficients whose structure is simpler to model. Most real-world images, especially gray-scale texture images, are well characterized by their *singularity* (edge and ridge) structure. The wavelet transform provides a powerful transform domain for modeling singularity-rich images [5].

The wavelet transform can be interpreted as a multiscale edge detector that represents the singularity content of an image at multiple scales and three different orientations. Wavelets overlying a singularity yield large wavelet coefficients; wavelets overlying a smooth region yield small coefficients. Four wavelets at a given scale nest inside one at the next coarser scale, giving rise to a quad-tree structure of wavelet coefficients that mirrors that of the dyadic squares (see Fig. 2(a)). In particular, with the *Haar wavelet transform*, each wavelet coefficient node in the wavelet quad-tree corresponds to a wavelet supported exactly on the corresponding dyadic image square.

In combination, the multiscale singularity detection property and tree structure imply that image

singularities manifest themselves as cascades of large wavelet coefficients through scale along the branches of the quad-tree [5]. Conversely, smooth regions lead to cascades of small coefficients.

This multiscale singularity characterization makes the wavelet domain natural for modeling texture images. Crouse et al. [6] have developed the *hidden Markov tree* (HMT) model, a parametric statistical model for wavelet transforms. The HMT owes its flexibility to two key ingredients. First, it differentiates between "large" and "small" wavelet coefficients by associating with each coefficient a binary state variable that controls its size. Assuming that each coefficient is Gaussian distributed when conditioned on its state models the marginal distribution of each coefficient is a Gaussian mixture. Second, to capture the fact that large and small wavelet coefficients cascade through scale, the states are connected in a Markovian probabilistic quad-tree that mirrors that of the wavelet transform. Each state-to-state link has an underlying state transition matrix that controls (probabilistically) the persistence of large and small states down the tree. Grouping the model parameters into the vector $\mathcal{M}$, the result is a high-dimensional yet highly structured Gaussian mixture model $f(\boldsymbol{w}|\mathcal{M})$ that approximates the overall joint pdf of the wavelet coefficients $\boldsymbol{W}$.

One of the most attractive characteristics of wavelet-based image processing algorithms is that the wavelet transform of an n-pixel image can be computed in just $O(n)$ computations. This efficiency carries over to HMT-based processing. The HMT can be trained to match a set of training data using the iterative expectation-maximization (EM) algorithm at a cost of $O(n)$ computations per iteration. More importantly, given the wavelet transform $\widetilde{\boldsymbol{w}}$ of a test image $\widetilde{\boldsymbol{x}}$ and a set of HMT parameters $\mathcal{M}$, computation of the likelihood $f(\widetilde{\boldsymbol{w}}|\mathcal{M})$ that $\widetilde{\boldsymbol{w}}$ is a realization of the HMT model requires only a simple $O(n)$ calculation. In the likelihood calculation, we place the wavelet transform of the test data on the HMT (fill in the black nodes in Fig. 2(c)) and then sweep up through the tree from leaves to root, performing simple calculations from each scale to the next [6].

The HMT has a nesting structure that matches that of the dyadic squares. Each subtree of the HMT is itself an HMT, with the HMT subtree rooted at node i modeling the statistical behaviour of the wavelet coefficients corresponding to the dyadic square $\boldsymbol{D}_i$. Serendipitously, the partial likelihood calculations obtained at intermediate scales of the HMT tree as part of the leaves-to-root upsweep give the likelihoods $f(\boldsymbol{d}_i|\mathcal{M})$ of *each dyadic subsquare of the image* under the HMT model.

These tools enable a simple multiscale image classification algorithm. Suppose that for each

texture class $c \in \{1, 2, \ldots, N_c\}$ we have specified or trained HMTs with parameters $\mathcal{M}_c$. Now, given the wavelet transform $\widetilde{\boldsymbol{w}}$ of an image $\widetilde{\boldsymbol{x}}$ consisting of a montage of these textures, applying the above multiscale likelihood calculation on each HMT yields the likelihoods $f(\widetilde{\boldsymbol{d}}_i|\mathcal{M}_c)$, $c \in \{1, 2, \ldots, N_c\}$ for each dyadic subimage $\widetilde{\boldsymbol{d}}_i$. Having the multiscale likelihoods at hand, the simplest ML classification

$$\widehat{c}_i^{\mathrm{ML}} := \arg\max_{c \in \{1,2,\ldots,N_c\}} f(\widetilde{\boldsymbol{d}}_i|\mathcal{M}_c) \tag{1}$$

then informs us of the most likely label $\widehat{c}_i^{\mathrm{ML}}$ for each dyadic subimage $\widetilde{\boldsymbol{d}}_i$. This classification process, which we call the *raw ML segmentation*, can be completed in just $O(n)$ computations for an n-pixel image. It yields a set of J different segmentations $\boldsymbol{c}_{\mathrm{ML}}^j$, $j = 0, 1, \ldots, J-1$, one for each different scale j of dyadic square.

Fig. 3 illustrates the process. After training HMT models on the grass and wood textures from Fig. 3(a) and (b), we performed the multiscale classification (1) on the test image (c) to obtain the raw segmentations (d) at various scales.

1.4 Interscale decision fusion

While quick and easy, as Fig. 3(d) attests, the raw ML segmentations suffer from the classical "blockiness vs. robustness" tradeoff that leaves no single $\boldsymbol{c}_{\mathrm{ML}}^j$ desirable. To obtain a high-quality segmentation, clearly we should combine the multiscale results to benefit from both the robustness of large block sizes and the resolution of small block sizes.

Since finer scale dyadic squares nest inside coarser scale squares, the dyadic squares will be statistically dependent across scale for images consisting of fairly large, homogeneous regions. Hence, (reliable) coarse-scale information should be able to help guide (less reliable) finer-scale decisions.

If the dyadic square $\boldsymbol{d}_i^{j-1}$ was classified as class c, then it is quite likely that its four children squares at scale j belong to the same class, especially when j is large (at fine scales). Hence, we will guide the classification decisions for the child squares based on the decision made for their parent square. This will tend to make the class labels of the four children the same unless their likelihood values strongly indicate otherwise, thus reducing the number of misclassifications due to slight perturbations in child likelihood values. In addition to the parent square, we can also use the neighbors of the parent to guide the decision process.

To exploit these parent-child dependencies between the dyadic squares, we will build yet another

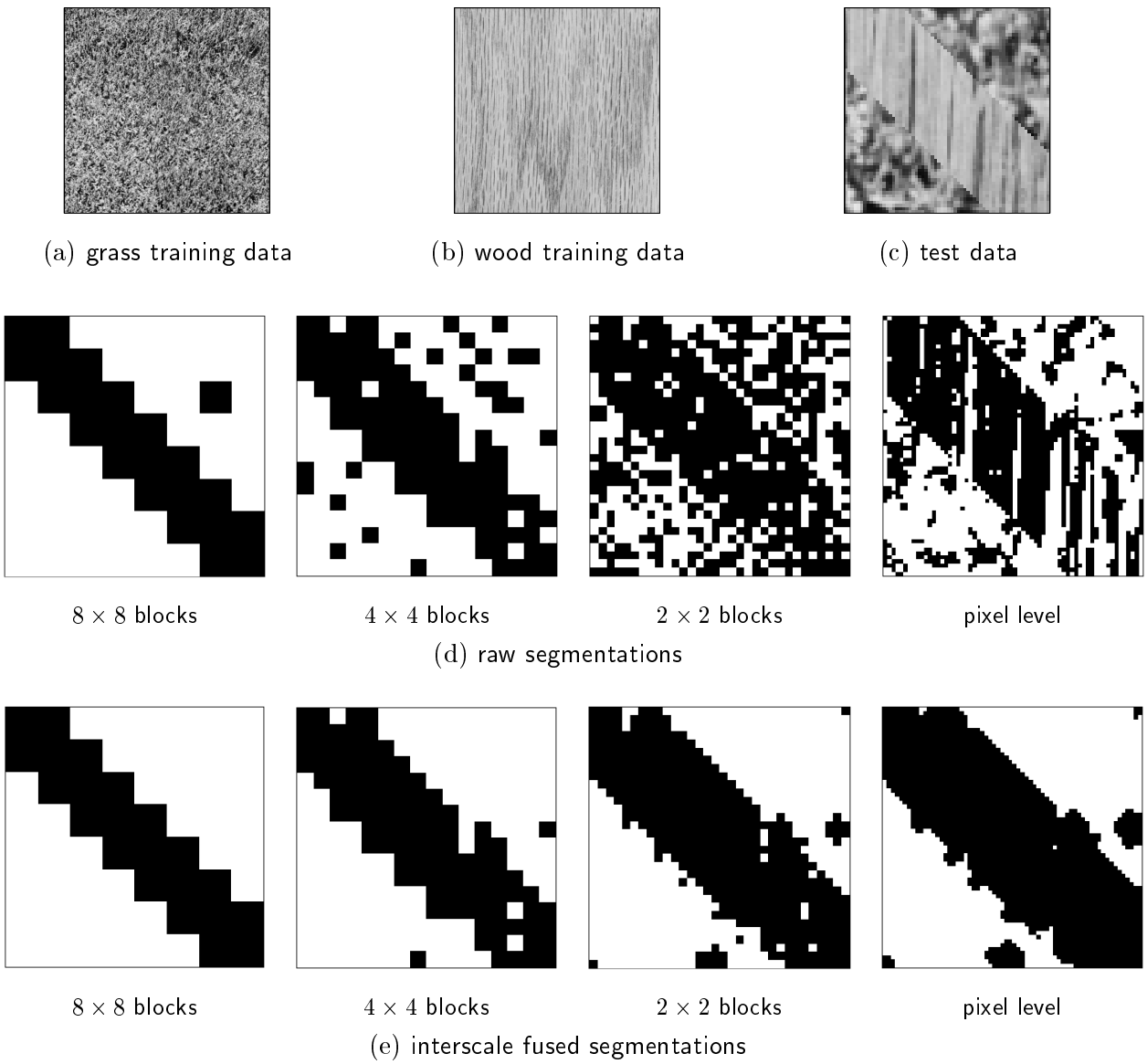


Figure 3: *HMTseg on a synthetic test image. (a)* 512×512 *grass texture image [7], (b)* 512×512 *wood texture image [7], (c)* 64×64 *grass/wood mosaic test image* $\widetilde{\boldsymbol{x}}$ *to be segmented. (d) Raw HMT-based multiscale classifications* $\boldsymbol{c}^j_{\mathrm{ML}}$ *of* $\boldsymbol{x}$ *for* 8×8, 4×4, 2×2, *and pixel-sized dyadic squares. Classification accuracy increases with block size (towards coarser scales) because more statistical information is available for the class label decision. However, this comes at a cost of reduced boundary resolution. (e) Final segmentations* $\boldsymbol{c}^j_{\mathrm{MAP}}$ *using Bayesian context-based interscale fusion.*

tree-structured probability model, the *labeling tree* (more details in Section 4.3 below). Akin to the HMT, the labeling tree models the dependencies between dyadic squares across scale in a Markov fashion, where the dyadic squares at scale j are assumed to depend only on the squares at scale $j-1$. (Dependencies between squares within the same scale are captured through the squares' common ancestors.) While we could use a more general model, for the above-mentioned reasons, there are tremendous "economies of scale" to be gained using tree-based modeling.

Markov modeling leads us to a simple scale-recursive classification of the dyadic squares, where we classify $\boldsymbol{d}_i^j$ based on its likelihood and guidance from the previous scale $j-1$. This Bayesian *interscale decision fusion* computes a MAP estimate of the class label $\widehat{c}_i^{\mathrm{MAP}}$ of each dyadic square $\boldsymbol{d}_i$. Stopping the fusion at scale j, we obtain the MAP segmentation $\boldsymbol{c}_{\mathrm{MAP}}^j$. As we see from Fig. 3(e), multiscale decision fusion greatly improves the robustness and accuracy of the segmentation.

1.5 HMTseg algorithm

Combining the above tools results in a robust and accurate yet simple and efficient segmentation algorithm that we call *HMTseg* [8]. It relies on three separate tree structures: the wavelet transform quad-tree, the HMT, and the labeling tree.

HMTseg Algorithm

1. **Train wavelet-domain HMT models** for each texture using homogeneous training images.

2. **Compute multiscale likelihoods.** Using the likelihood computation algorithm for the HMT model [6], compute the likelihood of each dyadic image square at each different scale. The tournament (1) for each dyadic square yields the ML *raw classifications* $\boldsymbol{c}_{\mathrm{ML}}^j$ for a range of scales j.

3. **Fuse multiscale likelihoods using the labeling tree** to form the multiscale MAP classification. The Bayesian interscale fusion guides fine scale decisions using coarse scale information to obtain the final segmentations $\boldsymbol{c}_{\mathrm{MAP}}^j$ for a range of scales j.

1.6 Related work

HMTseg has several distinct advantages over existing segmentation techniques. Markov random fields (MRF) [9–11] have been extensively applied to model the pixel pdf $f(\boldsymbol{x})$. However, while they

enable spatially local processing, they capture only local interactions and thus have only a limited ability to describe large scale behavior. MRFs can be improved by incorporating more neighboring pixels, but this rapidly increases the complexity of the segmentation algorithm.

As far as we know, HMTseg is the first attempt to use wavelet-domain statistical modeling for *multiscale* image segmentation [8]. (Li and Gray employ wavelet coefficient statistics in [12], but do not compute multiscale segmentations.) Among the many different approaches to multiscale modeling and its application to image segmentation, the multiscale autoregressive (MAR) model of Willsky at al. [13–15] and the multiscale labeling model of Bouman et al. [4, 16, 17] figure prominently.

The MAR models the multiscale statistics of the *scaling* coefficients for segmentation of image in a divide-and-conquer fashion. While the main advantage of multiscale image segmentation is to avoid the ad hoc choice of the classification window size, the MAR segmentation algorithm in [15] (and a similar one in [18]) still requires a proper choice.

The multiscale labeling model does not use an explicit model of the image pixels; rather it indirectly models the pixel pdf using a multiscale model of the class labels only. The technique in [4] is a general systematic method of combining multiscale information. However, because it considers only the behavior of the class labels across scale without actually considering the joint statistics of the image pixels (it assumes that the pixels are independent given the class label), the algorithm is useful only for certain types of images. The algorithms recently proposed in [16, 17] generalize [4] further. However, because these algorithms still do not perform direct modeling and decision of class labels at multiple scales, they require complicated statistical learning methods based on manually prepared training data. Furthermore, they model the wavelet coefficients as independent, which is not accurate for singularity-rich data such as textures because of the strong residual correlations between wavelet coefficients.

HMTseg combines both a direct multiscale likelihood computation using wavelet HMTs and a model of the multiscale behavior of the class labels (labeling tree) using an algorithm similar to that in [16, 17]. Since we obtain the multiscale likelihoods and classifications directly through the HMTs, the multiscale information fusion simplifies considerably. As a result, unlike the algorithms in [16, 17], we are able to extract the labeling tree parameters from the given image to be segmented,

without additional training data.

1.7 Paper organization

In Sections 2 and 3, we study the two basic ingredients of HMTseg: the wavelet transform and the wavelet HMT model. We describe the algorithm in Section 4. Section 5 demonstrates the performance of HMTseg through a number of examples. We conclude in Section 6 by pointing to some remaining issues and suggesting directions for further research.

2 The Wavelet Transform

2.1 Wavelet transform and dyadic squares

The wavelet transform represents the singularity content of an image at multiple scales. There are several different interpretations; we will find the pyramidal multiscale construction for discrete images cleanest for our purposes [19].

We will focus on the simplest wavelet transform, that of Haar. The construction of Haar wavelet coefficients of an image can be explained using four 2-D wavelet filters: the local smoother $h_{\mathrm{LL}} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$, horizontal edge detector $g_{\mathrm{LH}} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix}$, vertical edge detector $g_{\mathrm{HL}} = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix}$, and diagonal edge detector $g_{\mathrm{HH}} = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$.

To compute the wavelet transform of a $2^J \times 2^J$ discrete image $\boldsymbol{x}$, first set $\boldsymbol{u}_J[k,l] := \boldsymbol{x}[k,l]$, $0 \le k,l \le 2^J - 1$. Next, convolve $\boldsymbol{u}_J$ with the filters h_{LL}, g_{LH}, g_{HL}, and g_{HH} and discard every other sample in both the k and l directions. The resulting *subband* images — $\boldsymbol{u}_{J-1}$, $\boldsymbol{w}_{J-1}^{\mathrm{LH}}$, $\boldsymbol{w}_{J-1}^{\mathrm{HL}}$, and $\boldsymbol{w}_{J-1}^{\mathrm{HH}}$, respectively — are each of size $2^{J-1} \times 2^{J-1}$. The 4-pack can be compactly stacked back into a $2^J \times 2^J$ matrix $\begin{bmatrix} \boldsymbol{u}_{J-1} & \boldsymbol{w}_{J-1}^{\mathrm{HL}} \\ \boldsymbol{w}_{J-1}^{\mathrm{LH}} & \boldsymbol{w}_{J-1}^{\mathrm{HH}} \end{bmatrix}$. The filtering and downsampling process can now be continued on the $\boldsymbol{u}_{J-1}$ image and the procedure iterated up to J times (see Fig. 2(a)).

The *scaling coefficient* matrices $\boldsymbol{u}_j$, $0 \le j \le J-1$ are progressively smoothed versions of the original image $\boldsymbol{u}_J$. The *wavelet coefficient* matrices $\boldsymbol{w}_j^{\mathrm{LH}}$, $\boldsymbol{w}_j^{\mathrm{HL}}$, and $\boldsymbol{w}_j^{\mathrm{HH}}$ are high- and band-pass filtered, *edge-detected*, versions of the image that respond strongly to edges in the horizontal, vertical, and diagonal orientations, respectively. For example, the wavelet coefficient $\boldsymbol{w}_{J-1}^{\mathrm{LH}}[k,l]$,

$0 \leq k, l \leq 2^{J-1} - 1$, is large if the 2×2 image block $\begin{bmatrix} \boldsymbol{x}[2k, 2l] & \boldsymbol{x}[2k, 2l+1] \\ \boldsymbol{x}[2k+1, 2l] & \boldsymbol{x}[2k+1, 2l+1] \end{bmatrix}$ contains a horizontal edge and small otherwise.

The iterative computation of each Haar wavelet coefficient from a 2×2 block in a finer-scale image leads naturally to a quad-tree structure on the wavelet coefficients in each subband, as illustrated in Fig. 2(a) and (b) [20]. First assume that we carry out the iterated filtering to scale $j = 0$ and consider only the LH subband. Then the root of the tree lies at $\boldsymbol{w}_0^{\mathrm{LH}}[0, 0]$ and the leaves at $\boldsymbol{w}_{J-1}^{\mathrm{LH}}[k, l]$, $0 \leq k, l \leq 2^J - 1$. As we move down the tree, we move from coarse to fine scale, adding details as we go. More specifically, each *parent* wavelet coefficient $\boldsymbol{w}_j^{\mathrm{LH}}[k, l]$ analyzes the same region in the original image as its four *children* $\boldsymbol{w}_{j+1}^{\mathrm{LH}}[2k, 2l]$, $\boldsymbol{w}_j^{\mathrm{LH}}[2k, 2l+1]$, $\boldsymbol{w}_j^{\mathrm{LH}}[2k+1, 2l]$, and $\boldsymbol{w}_j^{\mathrm{LH}}[2k+1, 2l+1]$. Coefficients on the path to the root are *ancestors*; coefficients on the paths to the leaves are *descendents*. If we terminate the iterated filtering at a scale $j > 0$, then there will be more than one coarsest scale wavelet coefficient in each subband, leading to a forest of quad-trees in each subband [6].

To keep the notation manageable in the sequel, let $\boldsymbol{w}$ denote the collection of all wavelet coefficients and let $\boldsymbol{w}^{\mathrm{LH}}$, $\boldsymbol{w}^{\mathrm{HL}}$, $\boldsymbol{w}^{\mathrm{HH}}$ denote the collections of all coefficients in the respective subbands. Let w_i denote a generic wavelet coefficient, with the subband under consideration determined by context. In our statistical modeling framework, we will regard w_i as a realization of the random variable W_i and $\boldsymbol{w}$ as a realization of the wavelet random field $\boldsymbol{W}$. Define by $J(i)$ the scale of coefficient i in the subband quad-tree. Define $\rho(i)$ as the parent of tree node i. In a given subband, define $\mathcal{T}_i$ as the subtree of wavelet coefficients with root node i; that is, $\mathcal{T}_i$ contains coefficient w_i and all of its descendants (see Fig. 2(b)).

With the 2-D Haar wavelet transform, there is an obvious correspondence between the wavelet coefficients and the dyadic squares (recall Fig. 1(a)), which are obtained by iteratively dividing the image into equal-size quadrants. Recall that $\boldsymbol{d}_i^j$ denotes a dyadic square at scale j, with i an abstract index for the square. (In the sequel, superscripts will always denote scale and subscripts will always denote position within a scale.) Each $\boldsymbol{d}_i^j$ is obtained by dividing a square at scale $j-1$ (the "parent" square, $\boldsymbol{d}_{\rho(i)}^{j-1}$) into four quadrants (the "child" squares). To each dyadic square of pixels $\boldsymbol{d}_i$ there corresponds a unique wavelet coefficient w_i with a special property: all wavelet

coefficients in the subtree $\mathcal{T}_i$ rooted at w_i depend exclusively on the pixel values in $\boldsymbol{d}_i$.

The same procedure of wavelet transform construction procedure can be applied to other wavelet systems besides the Haar. While larger wavelet filters are more appropriate for representing smooth images, the Haar system is more appropriate for our purpose of classifying dyadic squares due to its direct connection with the dyadic squares. We will see that the Haar system is more than adequate for the HMTseg algorithm.

2.2 Wavelet transform properties

Wavelet transforms possess a number of endearing properties that make wavelet-domain statistical image processing attractive [5, 19]:

W1. Locality: Each wavelet coefficient represents the image content localized in spatial location and frequency.

W2. Multiresolution: The wavelet transform analyzes images at a nested set of scales.

W3. Energy Compaction: The wavelet transforms of real-world images tend to be sparse. A wavelet coefficient is large only if edges are present within the support of the corresponding wavelet filter.

W4. Decorrelation: The wavelet coefficients of real-world images tend to be approximately decorrelated (and the only correlations that remain are local, see **W6** below).

The Locality and Multiresolution properties (**W1**,**W2**) enable the wavelet transform to efficiently represent only the edge content of real-world images with large coefficients, resulting in the Compaction property (**W3**), because edges make up only a very small portion of a typical image. The Compaction and Decorrelation properties (**W3**,**W4**) simplify the statistical modeling of images in the wavelet domain as compared with a direct spatial-domain modeling. Because most of the wavelet coefficients tend to be small, we need only model a small number of coefficients accurately.

The Compaction (**W3**) of signal energy in the wavelet domain can be restated statistically in terms of a distinct marginal distribution of the wavelet coefficients:

W5. NonGaussianity: The wavelet coefficients have peaky, heavy-tailed, nonGaussian marginal statistics.

The Decorrelation property (**W4**) inspires simple, spatially localized modeling of the wavelet coefficients. There have been several successful attempts at modeling each wavelet coefficient as independent with a nonGaussian marginal pdf [21, 22].

While independent models are simple and easy to handle, modeling the residual dependencies between wavelet coefficients improves the modeling accuracy considerably [6, 23–25]. The relationship between singularities and the behavior of the wavelet coefficients across scale leads to the following strong dependencies between wavelet coefficients:

W6. Persistence and Clustering: Large/small values of wavelet coefficients tend to propagate across scale in the wavelet quad-tree [26, 27]. If a particular wavelet coefficient is large/small, then adjacent coefficients are very likely to also be large/small [6, 28].

These properties indicate that the wavelet transforms of real-world images have a local dependency structure that should not be ignored. With these facts in mind, we now turn to modeling images in the wavelet domain.

3 Wavelet-domain Hidden Markov Tree Model

The wavelet hidden Markov tree (HMT) models the joint statistics of the wavelet coefficients by capturing both the nonGaussian marginal pdf (**W5**) and the key joint dependencies (**W6**) of the wavelet coefficients [6]. In this Bayesian framework, the image is regarded as a random realization from a distribution or family of images.

3.1 Modeling the nonGaussian marginal distribution (W5)

The Compaction property (**W3**) of the wavelet transform implies that the transform of most real-world images consists of a small number of large coefficients and a large number of small coefficients. We can consider the population of small coefficients as outcomes of a pdf with a small variance. Similarly, the collection of large coefficients can be considered as outcomes of a pdf with a large variance. Hence, the pdf $f(w_i)$ of each wavelet coefficient is well approximated by a

two-density *Gaussian mixture model* [29–31].[2]

To each wavelet coefficient W_i, we associate a discrete hidden state S_i that takes on the values $m = \mathsf{S},\mathsf{L}$, signifying the small and large variance, with probability mass function (pmf) $p_{S_i}(m)$. Conditioned on $S_i = m$, W_i is Gaussian with mean $\mu_{i,m}$ and variance $\sigma^2_{i,m}$. Thus, its overall pdf is given by

$$f(w_i) \;=\; \sum_{m=\mathsf{S},\mathsf{L}} p_{S_i}(m)\, f(w_i|S_i = m), \tag{2}$$

where $f(w_i|S_i = m) \sim N(\mu_{i,m}, \sigma^2_{i,m})$ and $p_{S_i}(\mathsf{S}) + p_{S_i}(\mathsf{L}) = 1$.

3.2 Modeling the key dependencies (W6)

Once we model the marginal density of each wavelet coefficient as a Gaussian mixture, dependencies between the wavelet coefficients can be captured by specifying the joint probability mass function of the hidden states. Thanks to the approximate decorrelation of the wavelet coefficients (**W4**), the most important correlations are the parent-child interactions due to the persistence across scale property (**W6**). The HMT assumes that:

A1. The dependency structure of the wavelet coefficients in each subband has a quad-tree structure.

For now, consider modeling one subband of the wavelet transform. In Fig. 2(c) we picture the wavelet coefficients as black nodes and their associated hidden states as white nodes. To capture **W6**/**A1**, we connect the hidden states in a directed Markov-1 probabilistic graph [6]. For each parent-child pair of hidden states $\{S_{\rho(i)}, S_i\}$, the state transition probabilities $\epsilon^{\rho(i),m}_{i,m'}$ for $m, m' = \mathsf{S},\mathsf{L}$ represent the probability for W_i to be small/large when its parent $W_{\rho(i)}$ is small/large. For each i, we thus have the state transition probability matrix $\begin{bmatrix} \epsilon^{\rho(i),\mathsf{S}}_{i,\mathsf{S}} & \epsilon^{\rho(i),\mathsf{S}}_{i,\mathsf{L}} \\ \epsilon^{\rho(i),\mathsf{L}}_{i,\mathsf{S}} & \epsilon^{\rho(i),\mathsf{L}}_{i,\mathsf{L}} \end{bmatrix} = \begin{bmatrix} \epsilon^{\rho(i),\mathsf{S}}_{i,\mathsf{S}} & 1-\epsilon^{\rho(i),\mathsf{S}}_{i,\mathsf{S}} \\ 1-\epsilon^{\rho(i),\mathsf{L}}_{i,\mathsf{L}} & \epsilon^{\rho(i),\mathsf{L}}_{i,\mathsf{L}} \end{bmatrix}$. For typical gray-scale images, we expect $\epsilon^{\rho(i),\mathsf{S}}_{i,\mathsf{S}}$ and $\epsilon^{\rho(i),\mathsf{L}}_{i,\mathsf{L}}$ to be large due to the persistence property **W6**. The quad-tree dependency structure also indirectly captures some clustering property of the coefficients within scale through the mutual ancestors. This completes the specification of the HMT model for one subband.

[2] We can use more than two mixture densities to provide a fit to the actual $f(w_i)$ with any desired fidelity. In practice, however, we have seen no performance benefit to using more than two.

It is important to emphasize that the Markov structure of the HMT is on the *states* of the wavelet coefficients and not on the coefficients themselves. In the HMT, each wavelet coefficient W_i is conditionally independent of all other random variables given its state S_i. Furthermore, given the parent state $S_{\rho(i)}$, the pair of nodes $\{S_i, W_i\}$ are independent of the entire tree except for S_i's descendants.

A complete 2-D wavelet transform has three subbands with parallel quad-tree structures. In particular, node i in the LH, HL, and HH quad-trees corresponds to the same dyadic square $\boldsymbol{d}_i$ in the image. While the three subbands must therefore be dependent on each other, for tractability reasons, the HMT assumes that:

A2. The three subbands of the 2-D wavelet transform are independent.

As we will see from our results below, assumptions **A1**,**A2** are fairly mild.

In addition to modeling the wavelet coefficients, we can separately model the scaling coefficients, using a mixture density, for example [6]. However, for image segmentation, we intentionally ignore the scaling coefficients. Since the values of the scaling coefficient corresponds to local averages of the pixel values, we thus build into our statistical models independence to the local brightness level. This is a desirable feature for many image segmentation applications, because even in a region having homogeneous statistical properties, the local brightness level often varies in different parts of the region.

3.3 HMT parameters

Each quad-tree HMT has parameters

$$\boldsymbol{\Theta} := \begin{cases} \mu_{i,m},\ \sigma^2_{i,m} & \text{(mixture means and variances)} \\ \epsilon^{\rho(i),m}_{i,m'} & \text{(state transition probabilities from } S_{\rho(i)} \text{ to } S_i) \\ p_{S_0}(m) & \text{(pmf for the root node state).} \end{cases}$$

Given $\boldsymbol{\Theta}$, the HMT models the joint pdf of the subband wavelet coefficients.

The complete wavelet HMT model $\mathcal{M}$ consists of three HMTs (one for each wavelet subband). Denoting the parameter vectors for the three subband HMTs as $\boldsymbol{\Theta}^{\mathrm{HH}}, \boldsymbol{\Theta}^{\mathrm{HL}}$ and $\boldsymbol{\Theta}^{\mathrm{LH}}$, respectively, we have $\mathcal{M} := \{\boldsymbol{\Theta}^{\mathrm{HH}}, \boldsymbol{\Theta}^{\mathrm{HL}}, \boldsymbol{\Theta}^{\mathrm{LH}}\}$. The HMT is thus a parametric model for the joint pdf of the

wavelet coefficients. Using assumption **A2**, we can write

$$f(\boldsymbol{w}|\mathcal{M}) := f(\boldsymbol{w}^{\mathrm{LH}}|\boldsymbol{\Theta}^{\mathrm{LH}})\, f(\boldsymbol{w}^{\mathrm{HL}}|\boldsymbol{\Theta}^{\mathrm{HL}})\, f(\boldsymbol{w}^{\mathrm{HH}}|\boldsymbol{\Theta}^{\mathrm{HH}}). \tag{3}$$

As it stands, the HMT has a large number of parameters (approximately $4n$ for an n-pixel image). This can make model training difficult when only a small amount of training data is available. Fortunately, wavelet coefficients tend to exhibit similar statistical characteristics within the same scale [6, 32], and so we can often use the same parameters for those coefficients. This nodal *tying* reduces the number of parameters considerably, avoiding the risk of overfitting the model [6, 33].

3.4 Training and likelihood computation

We can *train* the wavelet HMT model parameters to match a set of training data. The iterative expectation-maximization (EM) algorithm finds the locally optimal (in the ML sense) set of model parameters $\mathcal{M}$ for given set of training data.[3] In each iteration, the E step defines a likelihood surface based on the current parameters. The M step then updates the parameters to maximize the likelihood that the training data came from the model. Iteration of the two steps is guaranteed to converge to an $\mathcal{M}$ that locally maximizes the likelihood [33]. In the HMT, each EM iteration consists of an up/down sweep through the tree ($O(n)$ cost for n wavelet coefficients). Once trained, the HMT provides a close approximation to the full joint pdf of the wavelet coefficients.

Given a set of 2-D HMT model parameters $\mathcal{M}$ and the wavelet transform $\widetilde{\boldsymbol{w}}$ of a test image, we can also compute the *likelihood* $f(\widetilde{\boldsymbol{w}}|\mathcal{M})$ that the image was generated by the model [6]. Furthermore, thanks to the dyadic multiscale structure of the wavelet transform and the HMT, we can obtain the *likelihoods of all dyadic squares of the image* simultaneously in a single upward sweep through the tree (a fast $O(n)$ algorithm).

Consider first the likelihood calculation for a subtree $\mathcal{T}_i$ of wavelet coefficients rooted at w_i in one of the subbands [6]. Suppose this subband has HMT parameters $\boldsymbol{\Theta}$. Given the conditional likelihood $\beta_i(m) := f(\mathcal{T}_i|S_i = m, \boldsymbol{\Theta})$ obtained by sweeping up the quad-tree from the leaves to node

[3] The EM algorithm derived for 1-D HMT models in [6] applies without modification in 2-D if we interpret the parent-child relations between nodes appropriately for quad-trees. For a general theory of probabilistic graphs and training algorithms, see [34].

i [6], the likelihood of the coefficients in $\mathcal{T}_i$ can be computed as

$$f(\mathcal{T}_i|\mathbf{\Theta}) = \sum_{m=\mathsf{S},\mathsf{L}} \beta_i(m)\, p(S_i = m|\mathbf{\Theta}), \tag{4}$$

with $p(S_i = m|\mathbf{\Theta})$ state probabilities obtained directly from $\mathbf{\Theta}$ (or computed during training).

Now the connection with the dyadic squares. It is easy to see that the wavelet coefficients of the square $\boldsymbol{d}_i$ consist of the triple $\{\mathcal{T}_i^{\mathrm{LH}}, \mathcal{T}_i^{\mathrm{HL}}, \mathcal{T}_i^{\mathrm{HH}}\}$, each a subtree of one of the three wavelet subband quad-trees. Using three upsweeps, we can easily compute the likelihood (4) for each of these subtrees. Then, using simplification **A2**, we have

$$f(\boldsymbol{d}_i|\mathcal{M}) = f(\mathcal{T}_i^{\mathrm{LH}}|\mathbf{\Theta}^{\mathrm{LH}})\, f(\mathcal{T}_i^{\mathrm{HL}}|\mathbf{\Theta}^{\mathrm{HL}})\, f(\mathcal{T}_i^{\mathrm{HH}}|\mathbf{\Theta}^{\mathrm{HH}}). \tag{5}$$

The HMT and this simple multiscale likelihood computation form the engine that drives the HMTseg algorithm.

4 Multiscale Segmentation using HMT models

We now fill in the sketch of the HMTseg algorithm given in Section 1.5. Since we have dealt with the HMT model and the multiscale likelihood calculation in detail above, we focus on the third, interscale decision fusion step in Section 4.3.

4.1 Training the HMT models

Before we begin the segmentation procedure, we must acquire training data representative of each texture to train the HMT models. We typically obtain these training images either by picking out homogeneous regions of the given image or from completely different images having homogeneous regions representative of the candidate textures. For each class $c \in \{1, \ldots, N_c\}$, we train a 2-D wavelet HMT model $\mathcal{M}_c$. When the number of training images is small, we use intra-scale tying to avoid overfitting the models [6].

4.2 Multiscale likelihood computation

With trained HMT models in hand for each class, the simple one-to-one correspondence between the dyadic squares and the Haar wavelet coefficients enables the HMT-based multiscale likelihood computation (5). The results are the likelihoods of the dyadic squares down to 2×2 block scale.

By the direct block-by-block comparison of the likelihoods (1), we obtain the ML raw segmentations at a range of scales (recall Fig. 3(d)). (We discuss how to carry this down to pixel-sized blocks below in Section 4.4.) We refer to this block-by-block classification as "raw," because we do not exploit any possible relationships between the classifications at different scales. We expect the raw decisions to be more reliable at coarser scales (where we have more image pixels per block) but more finely localized at finer scales (where the blocks are smaller). Unfortunately, this blockiness vs. robustness tradeoff renders the raw ML segmentations undesirable. Clearly it is in our best interest to overcome this tradeoff by folding the coarse-through-fine likelihoods into our final segmentation recipe.

4.3 Context-based interscale fusion

We can improve the raw segmentation considerably by considering the dependencies between the class decisions at different scales. We will do this by modeling the multiscale dependencies between the dyadic blocks.

4.3.1 Bayesian segmentation. In a Bayesian segmentation framework, we treat each class label c_i as a random variable C_i taking a value from $\{1, 2, \ldots, N_c\}$. Given the posterior distribution $p(c_i|\boldsymbol{x})$ of C_i given the image, the MAP classification of dyadic square $\boldsymbol{d}_i$ corresponds to the class label that maximizes the posterior distribution

$$\widehat{c}_i^{\mathrm{MAP}} := \arg\max_{c \in \{1,2,\ldots,N_c\}} p(c_i|\boldsymbol{x}). \tag{6}$$

By Bayes rule, the posterior is given by

$$p(c_i|\boldsymbol{x}) = \frac{f(\boldsymbol{x}|c_i)\, p(c_i)}{f(\boldsymbol{x})}. \tag{7}$$

Let $\boldsymbol{d} := \{\boldsymbol{d}_i\}$ denote the collection of all dyadic squares (at all scales) and note that $\boldsymbol{d}$ contains complete information on the image $\boldsymbol{x}$ (many times over). A posterior equivalent to (7) is thus

$$p(c_i|\boldsymbol{d}) = \frac{f(\boldsymbol{d}|c_i)\, p(c_i)}{f(\boldsymbol{d})}. \tag{8}$$

Since computation and maximization of (8) is intractable in practice, we will perform a succession of manipulations and simplifications to arrive at a practical MAP classifier. Just as the HMT models the pdfs $f(\boldsymbol{w})$ and $f(\boldsymbol{x})$ by echoing the structure of the wavelet coefficient quad-tree, we

will construct a probabilistic tree to model the posterior (8) based on the dyadic square quad-tree of Fig. 1(b). The resulting *labeling tree* model will capture the interscale dependencies between dyadic blocks and their class labels and enable a multiscale Bayesian decision fusion. There are many ways to capture these multiscale dependencies; here we outline one possible approach that balances accuracy with tractability.

4.3.2 Hidden feature variables. Rather than modeling the joint statistics of the dyadic squares $\boldsymbol{D}_i$'s directly, we will model the statistics of a set of associated *hidden feature variables.* To each $\boldsymbol{D}_i^j$, we assign the hidden feature variable $\boldsymbol{H}_i^j$ that we assume controls the textural properties of the square. That is, each $\boldsymbol{D}_i$ is generated based on the distribution $f(\boldsymbol{d}_i|\boldsymbol{H}_i = \boldsymbol{h}_i)$ independently of all other $\boldsymbol{H}_k$ and $\boldsymbol{D}_k$, $k \neq i$. Let $\boldsymbol{H} := \{\boldsymbol{H}_i\}$ denote the collection of all feature variables. Then, given $\boldsymbol{H} = \boldsymbol{h}$, all $\boldsymbol{D}_i$ are independent

$$f(\boldsymbol{d}|\boldsymbol{h}) = \prod_i f(\boldsymbol{d}_i|\boldsymbol{h}_i). \tag{9}$$

The hidden feature variables play a rôle analogous to the hidden states in the HMT: given the values of the states, all wavelet coefficients are independent.

Furthermore, assume that there exists a function T such that $C_i = T(\boldsymbol{H}_i)$, so that the distribution of C_i follows from the distribution of $\boldsymbol{H}_i$. Under these assumptions, our MAP classification problem transforms to maximizing the posterior $f(\boldsymbol{h}_i|\boldsymbol{d})$ (recall (8)), the marginal of

$$f(\boldsymbol{h}|\boldsymbol{d}) = \frac{f(\boldsymbol{d}|\boldsymbol{h})\, f(\boldsymbol{h})}{f(\boldsymbol{d})} = \frac{f(\boldsymbol{h})}{f(\boldsymbol{d})} \prod_i f(\boldsymbol{d}_i|\boldsymbol{h}_i). \tag{10}$$

Here we have used (9). Unfortunately, marginalizing this expression for the MAP decision statistic is difficult in general.

4.3.3 Contexts. To simplify the determination and marginalization of the joint posterior density in (10), we employ the concept of *context* [35]. To each dyadic square $\boldsymbol{D}_i^j$ with hidden feature variable $\boldsymbol{H}_i^j$, we assign the (deterministic) context vector $\boldsymbol{v}_i^j$, which is formed from information about other dyadic squares and hidden feature variables.

The triple $\boldsymbol{v}_i \to \boldsymbol{H}_i \to \boldsymbol{D}_i$ forms a Markov-1 chain (see Fig. 4(a)). That is, $\boldsymbol{v}_i$ encodes sufficient information that, given its value, we can treat $\boldsymbol{H}_i$ and $\boldsymbol{D}_i$ as independent of all other $\boldsymbol{H}_k$ and $\boldsymbol{D}_k$. If $\boldsymbol{v}_i$ is chosen as a discrete vector taking values from a finite set, then it simplifies the modeling considerably. Let $\boldsymbol{v}$ be the collection of all contexts and $\boldsymbol{v}^j$ all contexts at scale j.

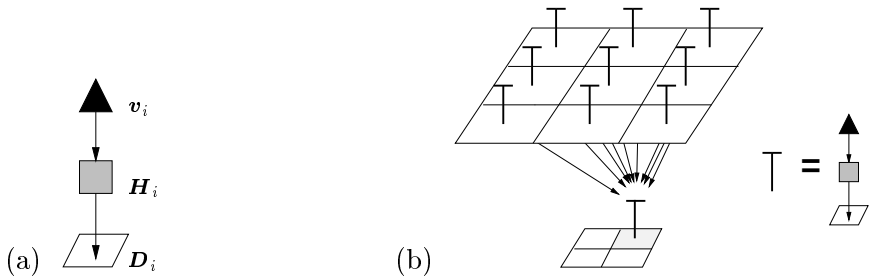


Figure 4: *(a) The context, hidden feature vector and dyadic square forms a Markov-1 chain:* $\boldsymbol{v}_i \to \boldsymbol{H}_i \to \boldsymbol{D}_i$. *(b) Contextual labeling tree. The context of the child square is determined by the decision results of the parent plus its eight neighbors.*

The choice of a good context model is crucial to the performance of the HMTseg. We have a trade-off between the complexity of the context and the accuracy of the model. Among many candidate contexts, we can determine the effective contexts based on known training data. In some sense, the decision-tree based algorithm in [4] is a general form of the context-based fusion algorithm applicable when sufficient training data is available for reliable estimation of the decision parameters.

Contexts allows us to write

$$f(\boldsymbol{h}|\boldsymbol{v}) = \prod_i f(\boldsymbol{h}_i|\boldsymbol{v}_i). \tag{11}$$

Thus, conditioning on the context decouples the joint distribution for the feature variables, which will trivialize the marginalization of (10). Since $\boldsymbol{D}$ is independent of $\boldsymbol{v}$ given $\boldsymbol{H}$ (by the Markov-1 property), conditioning (10) on the contexts yields

$$f(\boldsymbol{h}|\boldsymbol{d},\boldsymbol{v}) = \frac{f(\boldsymbol{d}|\boldsymbol{h})\, f(\boldsymbol{h}|\boldsymbol{v})}{f(\boldsymbol{d}|\boldsymbol{v})} = \frac{1}{f(\boldsymbol{d}|\boldsymbol{v})} \prod_i \left[f(\boldsymbol{d}_i|\boldsymbol{h}_i)\, f(\boldsymbol{h}_i|\boldsymbol{v}_i)\right] \tag{12}$$

and the marginalized, context-based posterior

$$f(\boldsymbol{h}_i|\boldsymbol{d}_i,\boldsymbol{v}_i) \propto f(\boldsymbol{d}_i|\boldsymbol{h}_i)\, f(\boldsymbol{h}_i|\boldsymbol{v}_i). \tag{13}$$

This is a greatly simplified version of the MAP posterior (7) for use in the MAP equation (6). Here, the $f(\boldsymbol{d}_i|\boldsymbol{h}_i)$ are the likelihoods of the dyadic square $\boldsymbol{d}_i$ given different values for its feature variable $\boldsymbol{h}_i$ (or equivalently the classes C_i), which are computed using a HMT likelihood sweep up each texture model. The prior $f(\boldsymbol{h}_i|\boldsymbol{v}_i)$ supplies information on $\boldsymbol{H}_i$ provided by the other $\boldsymbol{H}_k$'s through $\boldsymbol{v}_i$.

4.3.4 Contextual labeling tree. The contexts will model the dependencies between the various hidden feature variables and dyadic squares. But which information should enter into the contexts? Rather than modeling the dependencies at each single scale (with, say, a MRF), we will employ interscale dependency modeling. This approach is both simple and effective.

While each $\boldsymbol{v}_i$ is a potentially a function of all $\boldsymbol{H}_k$, $k \neq i$, here we will employ a tree organization: each $\boldsymbol{v}_i^j$ at scale j will receive information from nine scale $j-1$ feature variables, the parent feature variable $\boldsymbol{H}_{\rho(i)}$ and the parent's eight nearest neighboring $\boldsymbol{H}_k$ (see Fig. 4(b)). We term this organization the *contextual labeling tree.* The limit of coarser scale information to just nine blocks is easily justified by noting that $\boldsymbol{v}_i^j$ will receive information from a region of pixels centered around and 36 times larger than its square $\boldsymbol{d}_i^j$.

Two final simplifications. First, set $\boldsymbol{H}_i = C_i$, so that the feature variable controlling the texture of each square is merely the texture label itself. Second, inspired by the success of hybrid tree model in [4], we use a simple context structure. Each context vector $\boldsymbol{v}_i$ contains two entries: the value of the class label $C_{\rho(i)}$ of the parent square (which will be a MAP estimate in practice) and the majority vote of the class labels of the parent plus its eight neighbors. If there are N_c different textures, then the context can take on N_c^2 different values. Let the number of different values $\boldsymbol{v}_i$ can take be N_v ($= N_c^2$ in the algorithm); thus $\boldsymbol{v}_i \in \{\overline{\boldsymbol{v}}_1, \ldots, \overline{\boldsymbol{v}}_{N_v}\}$.

The simplification $C_i = \boldsymbol{H}_i$ transforms (13) to

$$p(c_i|\boldsymbol{d}_i, \boldsymbol{v}_i) \propto f(\boldsymbol{d}_i|c_i)\, p(c_i|\boldsymbol{v}_i), \tag{14}$$

our final, simplified posterior distribution. Since the $p(c_i|\boldsymbol{v}_i)$ depend on the C_k's from scale $j-1$, we will evaluate and maximize (14) in a multiscale, coarse-to-fine manner to fuse the HMT likelihoods $f(\boldsymbol{d}_i|c_i)$ (precomputed as in Section 4.2) using the labeling tree prior $p(c_i|\boldsymbol{v}_i)$. Our fusion will pass the MAP decisions down through scale to aid the segmentation of fine scale dyadic squares. The result is simple, yet effective.

4.3.5 Interscale fusion EM algorithm. The fusion proceeds as follows. Start at a coarse enough scale $j-1$ such that the ML raw segmentations $\boldsymbol{c}_{\mathrm{ML}}^j$ are statistically reliable. Use these and all coarser ML decisions as the MAP decisions $\boldsymbol{c}_{\mathrm{MAP}}^j$. This is entirely reasonable; at coarse scales (large dyadic squares), the next coarser scale (very large dyadic squares) provides little prior

information for segmentation.

Now move down to the next finer level j. Fix the context values $\boldsymbol{v}_i$ from the $\boldsymbol{c}^j_{\text{MAP}}$ at scale $j-1$ (from the parent feature variable and its eight nearest neighbors). We are given the likelihood $f(\boldsymbol{d}_i|c_i)$ in (14) from the HMT likelihood computation step. Hence, after computing $p(c_i|\boldsymbol{v}_i)$, we can choose the label for $\boldsymbol{c}^j_{\text{MAP}}$ that maximizes the product (14).

To compute $p(c_i|\boldsymbol{v}_i)$, we use an ML estimate averaged over the collection of *all* dyadic squares $\boldsymbol{d}_k$ at scale j. Since this collection is precisely the image $\boldsymbol{x}$, we can write (by the chain rule of conditioning)

$$f(\boldsymbol{x}|\boldsymbol{v}^j) = \prod_{J(i)=j} \sum_{l=1}^{N_c} f(\boldsymbol{d}^j_i|c_i = l)\, p(c_i = l|\boldsymbol{v}_i). \tag{15}$$

Here we sum over the N_c candidate textures and use the fact that all blocks at the same scale j are independent given the contexts $\boldsymbol{v}^j$. The ML estimate of $p(c_i|\boldsymbol{v}_i)$ is that which maximizes the likelihood of the image given the $\boldsymbol{v}_i$'s (given in (15)).

The EM algorithm comes to our rescue; in fact, we can use it to compute and maximize the posterior (14) directly. We do not specify $p(c_i|\boldsymbol{v}_i)$ directly, but rather specify $p(\boldsymbol{v}_i|c_i)$ and apply Bayes rule

$$p(c_i|\boldsymbol{v}_i) = \frac{p(\boldsymbol{v}_i|c_i)\, p(c_i)}{p(\boldsymbol{v}_i)}. \tag{16}$$

Assuming these probabilities to be constant at each scale, set

$$e_{j,m} := p_{c_i}(m), \qquad \alpha_{j,\overline{\boldsymbol{v}}_k,m} := p(\boldsymbol{v}_i = \overline{\boldsymbol{v}}_k|c_i = m) \tag{17}$$

for all i in scale j and $m \in \{1,\ldots,N_c\}$, $k \in \{1,\ldots,N_v\}$. The set of probabilities $\boldsymbol{P} := \{e_{j,m}, \alpha_{j,\overline{\boldsymbol{v}}_k,m}\}$ is computed using the EM algorithm on the contextual labeling tree (see the Appendix for details). Then, the context-based Bayes classification is performed by finding the class label that maximizes the contextual posterior distribution $p(c_i|\boldsymbol{d}_i, \boldsymbol{v}_i)$ from (14) (see (18) in the Appendix).

4.4 Pixel-level segmentation

Since the Haar wavelet HMT characterizes the joint statistics of dyadic image squares only down to 2×2 blocks, we do not directly obtain pixel-level segmentations.[4] Pixel-level segmentation

[4]While the collection of all wavelet and scaling coefficients completely characterizes the original image, the HMT subband independence assumption (**A2**) and the fact that we ignore the scaling coefficients limits our reach to 2×2

requires a model for the pixel brightness of each texture class. However, obtaining an appropriate model can be difficult, since in many images the local brightness varies considerably due to shading, etc. For such images, the 2×2 block segmentations will be far more robust, since they rely on inter-pixel dependencies and not local brightness.

Pixel brightness corresponds to the pdf of a single pixel. For our purposes, we fit a Gaussian mixture to the pixel values for each training texture. We can then compute the likelihood of each pixel and extend the above interscale scale fusion algorithm to the pixel level.

4.5 Implementation issues

As described above, the interscale fusion algorithm starts at the root node of the labeling tree and descends to the finest scale to combine all possible coarse scale information. However, at very coarse scales, the likelihoods of the dyadic squares do not contain significant information, since the squares are large and hence likely to contain several differently textured regions. When fusing multiscale classification results, we can therefore ignore the information at very coarse scales.

Ignoring the coarsest scales has several side benefits. If we start fusing at scale $j_0 > 0$, then we only need the wavelet coefficients, HMT models, and likelihoods at scales $j \geq j_0$. With the Haar transform, starting at scale $j_0 > 0$ is equivalent to dividing the image up into the dyadic squares $D_i^{j_0}$ and then performing HMTseg independently on each of these squares. This saves a considerable amount of computation and reduces the size of the required homogeneous training images to $2^{J-j_0} \times 2^{J-j_0}$. In practice, we set the starting scale j_0 such that the coarsest raw segmentations are reliable enough.

As we proceed to fine scales in the interscale fusion algorithm, the estimation of the context probabilities $p(c_i|\boldsymbol{v}_i)$ may become unstable due to the inevitable inaccuracy of the raw classifications. Since the $p(c_i|\boldsymbol{v}_i)$ tend to change little from scale to scale at fine scales, we cease estimating them beyond a certain scale. This is particularly desirable for pixel-level segmentation. When the pixel brightness models give inaccurate classifications of individual pixels, we can reuse the $p(c_i|\boldsymbol{v}_i)$ estimated at the 2×2-block scale in the interscale fusion. This technique was employed in the document segmentation example of Section 5.2.

blocks.

5 Examples

Figure 3 demonstrated the HMTseg process on a synthetic data example. Here we illustrate two real-world image segmentation problems.

5.1 Aerial photo segmentation

We trained wavelet HMTs for "sea" and "ground" textures using hand-segmented blocks from the 1024×1024 aerial photo [7] in Fig. 5(a). Choosing $j_0 = 4$ for the starting scale (corresponding to 6-scale quad-trees on 64×64 image blocks), we segmented the 256×256 test image in Fig. 5(b).

Fig. 5(c) shows the raw classification results. Pixel-level raw segmentation was obtained using 2-density Gaussian mixture models for pixel brightness of ground and sea textures. Fig. 5(d) illustrates the segmentation resulting from coarse-to-fine interscale fusion. Except for some segmentation errors in the upper middle part of the image (caused by the ground there having a texture more like sea), we observe excellent segmentation results at all scales.

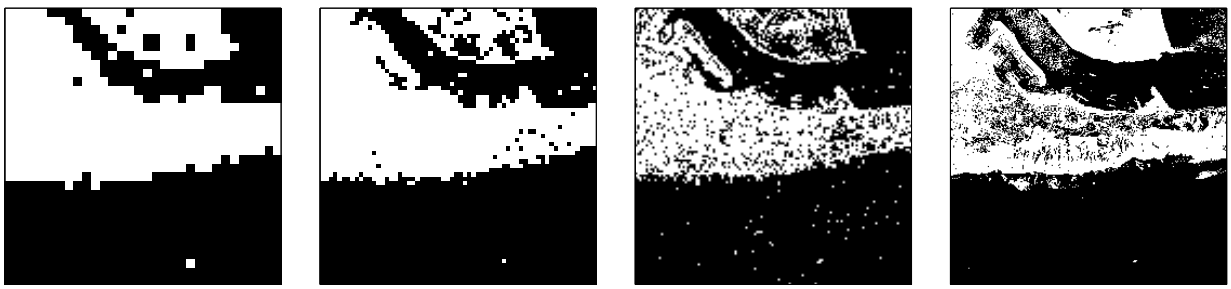
5.2 Document segmentation

We trained HMT and pixel brightness models for "text," "image," and "background" textures using hand-segmented blocks from the 512×512 document in Fig. 6(a). Choosing $j_0 = 3$ for the starting scale (corresponding to 6-scale quad-trees on 64×64 image blocks), we segmented the 512×512 test image in Fig. 6(b).

Fig. 6(c) shows the raw classification results and Fig. 6(d) illustrates the segmentation resulting from coarse-to-fine interscale fusion. Text, image, and background regions are displayed as black, gray, and white, respectively. All text regions were segmented well, including the text surrounded by images on the books. At the bottom, we observe that the large-font title text was segmented as image. This is because the homogeneous texture inside each large letter had properties more similar to images than (small-font) text. The background regions were correctly segmented, even though the brightness of the background varies in different areas and is corrupted by a noise-like feature caused by text on the reverse side of the page.

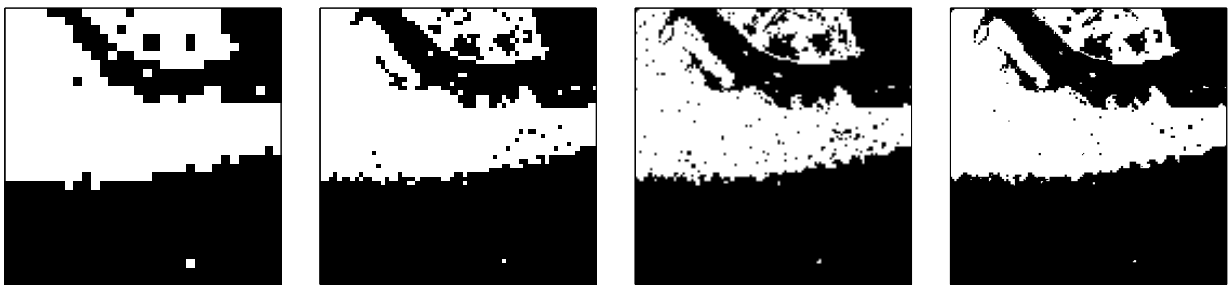
(a) training data

(b) test data



8×8 blocks 4×4 blocks 2×2 blocks pixel level

(c) raw segmentations



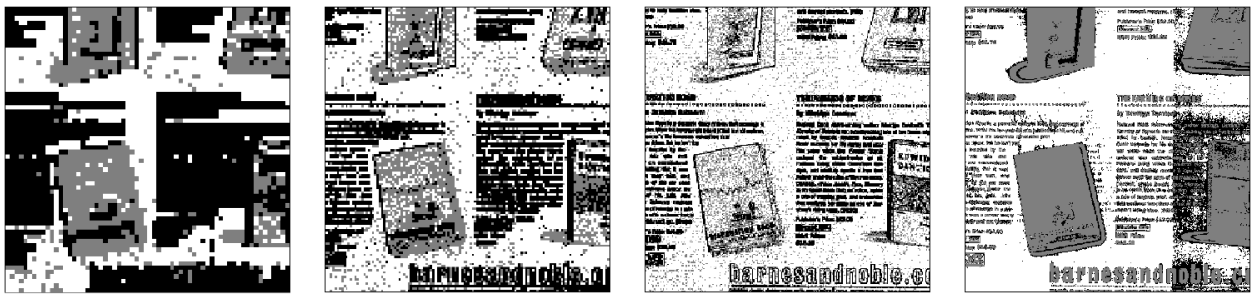
8×8 blocks 4×4 blocks 2×2 blocks pixel level

(d) interscale fused segmentations

Figure 5: *Aerial photo segmentation using HMTseg. (a)* 1024×1024 *aerial photo [7] and (b)* 256×256 *test subimage* $\boldsymbol{x}$*. The homogeneous ground/sea regions outside the region (b) were used to train two HMTs. (c) Raw HMT-based multiscale classifications* $\boldsymbol{c}^j_{\mathrm{ML}}$ *of* $\boldsymbol{x}$ *for* 8×8*,* 4×4*,* 2×2*, and pixel-sized dyadic squares. (d) Final segmentations* $\boldsymbol{c}^j_{\mathrm{MAP}}$ *using Bayesian context-based interscale fusion. The erroneous segmentation of the ground regions in the upper middle portion of the image is due to the large expanses of concrete (runways), whose texture is closer to that of sea than ground in this case.*

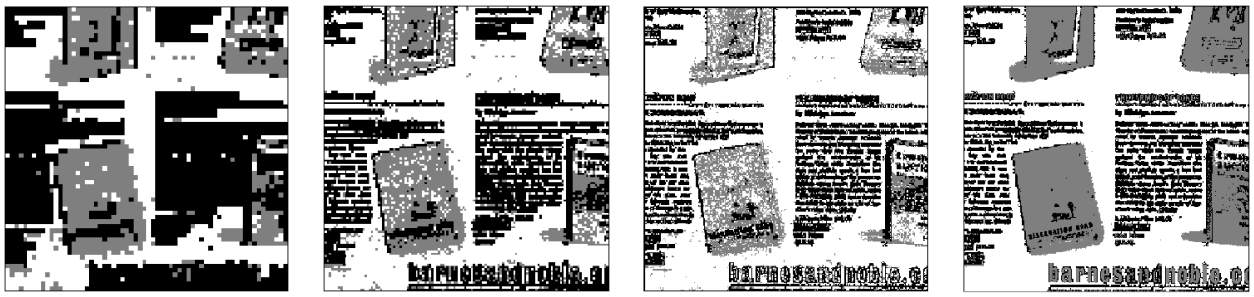
(a) training image

(b) test image



8×8 blocks $\quad$ 4×4 blocks $\quad$ 2×2 blocks $\quad$ pixel level

(c) raw segmentations



8×8 blocks $\quad$ 4×4 blocks $\quad$ 2×2 blocks $\quad$ pixel level

(d) interscale fused segmentations

Figure 6: *Document segmentation using HMTseg. (a)* 512×512 *training image was hand-segmented, and homogeneous regions were used to train HMTs for text, image, and background textures. (b)* 512×512 *test image* $\boldsymbol{x}$*. (c) Raw HMT-based multiscale classifications* $\boldsymbol{c}^j_{\mathrm{ML}}$ *of* $\boldsymbol{x}$ *for* 8×8, 4×4, 2×2, *and pixel-sized dyadic squares. Black, gray, and white represent text, image, and background, respectively. Classification accuracy clearly decreases at fine scales. (d) Final segmentations* $\boldsymbol{c}^j_{\mathrm{MAP}}$ *using Bayesian context-based interscale fusion correctly classify even the angled text on the books. Adding a fourth class (large text) would allow us to correctly classify the text at the bottom of* $\boldsymbol{x}$*.*

6 Conclusions

In this paper, we have developed a new framework for Bayesian image segmentation based on wavelet-domain HMT models. By concisely modeling and fusing the statistical behavior of textures at multiple scales, the HMTseg algorithm produces a robust and accurate segmentation of texture images. HMTseg yields not one final segmentation but a range at different scales.

While we have illustrated with photograph and document images, HMTseg can be applied to many different image types, including radar/sonar images and medical images. Furthermore, because the HMT modeling framework extends trivially to higher-dimensional data, we can employ HMTseg to segment multidimensional data such as geophysical surveys. 1-D signals, such as speech and well-logs, are also within HMTseg's purview.

As an added bonus, HMTseg has the potential to segment wavelet-compressed data directly without re-expanding to the space domain. HMTseg thus provides a natural vehicle for developing joint segmentation/compression algorithms.

Promising avenues for future HMTseg research include the investigation of wavelet basis representation different from Haar, simplified universal HMT modeling [32], more accurate (but complicated) interscale fusion algorithms, and the analysis of multiscale classification errors [36].

A Appendix: EM Algorithm for Context Labeling Tree

Our goal is to find $p(c_i|v_i)$ maximizing $f(\boldsymbol{x}|\boldsymbol{v}^j)$ in (15). We precompute the conditional likelihoods $f(\boldsymbol{d}_i^j|c_i)$ for all $c_i \in \{1, \ldots, N_c\}$ using (5) by sweeping up the HMTs from the leaves to node i [6]. Recall the definitions of $e_{j,m}$, $\alpha_{j,\overline{\boldsymbol{v}}_k,m}$, and $\boldsymbol{P}$ from (17). The EM algorithm runs as follows:

Initialize: Set $I = 0$ and choose $\boldsymbol{P}^0$.

(A natural choice for $\boldsymbol{P}^0$ is the set of parameters obtained in the previous, next coarser scale.)

Expectation (E): Given $\boldsymbol{P}^I$, calculate (using Bayes rule)

$$p(c_i = m|\boldsymbol{d}_i^j, \boldsymbol{v}_i^j) = \frac{e_{i,m}\, \alpha_{j,\boldsymbol{v}_{i,m}}\, f(\boldsymbol{d}_i^j|c_i = m)}{\sum_{l=1}^{N_c} e_{i,l} \alpha_{j,\boldsymbol{v}_{i,l}}\, f(\boldsymbol{d}_i^j|c_i = l)}. \tag{18}$$

Maximization (M): Update the elements of $\boldsymbol{P}^{I+1}$

$$e_{j,m} = \frac{1}{2^j} \sum_i p(c_i = m|\boldsymbol{v}_i^j, \boldsymbol{d}_i^j), \tag{19}$$

$$\alpha_{j,\overline{\boldsymbol{v}}_k,m} = \frac{1}{e_{j,m}} \sum_{i \text{ with } \boldsymbol{v}_i^j = \overline{\boldsymbol{v}}_k} p(c_i = m | \boldsymbol{v}_i^j, \boldsymbol{d}_i^j) \qquad \text{for each } \overline{\boldsymbol{v}}_k,\ k \in \{1, \ldots, N_v\}. \tag{20}$$

Iterate: Increment $I \to I + 1$ and apply E and M until converged.

References

[1] R. Haralick and L. Shapiro, "Image segmentation techniques," *Comput. Vision Graphics Image Processing*, vol. 29, pp. 100–132, 1985.

[2] C. Therrien, "An estimation-theoretic approach to terrain image segmentation," *Comput. Vision Graphics Image Processing*, vol. 22, pp. 313–326, 1983.

[3] H. Derin and H. Elliot, "Modeling and segmentation of noisy and textured images using Gibbs random fields," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-9, no. 1, pp. 39–55, Jan. 1987.

[4] C. Bouman and M. Shapiro, "A multiscale random field model for Bayesian image segmentation," *IEEE Trans. Image Proc.*, vol. 3, no. 2, pp. 162–177, March 1994.

[5] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, San Diego, 1998.

[6] M. S. Crouse, R. D. Nowak, and R. G. Baraniuk, "Wavelet-based statistical signal processing using hidden Markov models," *IEEE Trans. Signal Proc.*, vol. 46, no. 4, pp. 886–902, April 1998.

[7] The USC-SIPI Image Database, "`sipi.usc.edu/services.html`," .

[8] H. Choi and R. G. Baraniuk, "Image segmentation using wavelet-domain classification," in *Proceedings of SPIE technical conference on Mathematical Modeling, Bayesian Estimation, and Inverse Problems*, Denver, CO, July 1999, vol. 3816, pp. 306–320.

[9] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images," *IEEE Trans. on Pattern Anal. Machine Intell.*, vol. 6, pp. 721–741, 1984.

[10] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *J. Royal Statistical Society B*, vol. 36, pp. 192–225, 1974.

[11] R. Challappa and S. Chatterjee, "Classification of textures using Gaussian Markov random fields," *IEEE Trans. Acous. Speech. Signal Proc.*, vol. 33, pp. 959–963, 1985.

[12] J. Li and R. M. Gray, "Text and picture segmentation by the distribution analysis of wavelet coefficients," in *Proc. of ICIP'98*, Chicago, IL, Oct. 1998.

[13] M. Basseville, A. Benveniste, K. C. Chou, S. A. Golden, R. Nikoukhah, and A. S. Willsky, "Modeling and estimation of multiresolution stochastic processes," *IEEE Trans. on Info. Theory*, vol. 38, no. 2, pp. 766–784, Mar. 1992.

[14] M. R. Luettgen, W. C. Karl, A. S. Willsky, and R. R. Tenney, "Multiscale representations of Markov random fields," *IEEE Trans. Signal Proc.*, vol. 41, no. 12, pp. 3377–3395, Dec. 1993.

[15] C. Fosgate, H. Krim, W. Irving, W. Karl, and A. Willsky, "Multiscale segmentation and anomaly enhancement of SAR imagery," *IEEE Trans. on Image Proc.*, vol. 6, no. 1, pp. 7–20, Jan. 1997.

[16] H. Cheng, C. A. Bouman, and J. P. Allebach, "Multiscale document segmentation," in *IS&T 50th Annual Conference*, Cambridge, MA, May 18-23 1997, pp. 417–425.

[17] H. Cheng and C. A. Bouman, "Trainable context model for multiscale segmentation," in *IEEE Int. Conf. on Image Proc. — ICIP '98*, Chicago, IL, Oct. 4-7 1998.

[18] A. Kim and H. Krim, "Hierarchical stochastic modeling of SAR imagery for segmentation/compression," *IEEE Trans. Signal Processing*, vol. 47, no. 2, pp. 458–468, Feb. 1999.

[19] M. Vetterli and J. Kovačević, *Wavlets and Subband Coding*, Prentice Hall, Englewood Cliffs:NJ, 1995.

[20] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Proc.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

[21] S. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, July 1989.

[22] P. Moulin and J. Liu, "Analysis of multiresolution image denoising schemes using generalized-Gaussian priors," in *Proc. IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, Pittsburgh, PA, Oct. 6-9 1998, pp. 633–636.

[23] E. P. Simoncelli and E. H. Adelson, "Noise removal via Bayesian wavelet coring," in *IEEE Int. Conf. on Image Proc. — ICIP 1996*, Lausanne, Switzerland, Sept. 1996.

[24] R. W. Buccigrossi and E. P. Simoncelli, "Image compression via joint statistical characterization in the wavelet domain," Tech. Rep. 414, GRASP Laboratory, University of Pennsylvania, May 1997.

[25] E. P. Simoncelli, "Statistical models for images: Compression, restoration and synthesis," in *Proc. 31st Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, November 1997.

[26] S. Mallat and S. Zhong, "Characterization of signals from multiscale edges," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 7, pp. 710–732, July 1992.

[27] S. Mallat and W. Hwang, "Singularity detection and processing with wavelets," *IEEE Trans. on Info. Theory*, vol. 38, no. 2, pp. 617–643, 1992.

[28] M. T. Orchard and K. Ramchandran, "An investigation of wavelet-based image coding using an entropy-constrained quantization framework," in *Data Compression Conference '94*, Snowbird, Utah, 1994, pp. 341–350.

[29] J. Pesquet, H. Krim, and E. Hamman, "Bayesian approach to best basis selection," in *Proceedings of ICIP'96*, Atlanta, GA, 1996, pp. 2634–2637.

[30] H. Chipman, E. Kolaczyk, and R. McCulloch, "Adaptive Bayesian wavelet shrinkage," *Journal of the American Statistical Association*, vol. 92, 1997.

[31] F. Abramovich, T. Sapatinas, and B. W. Silverman, "Wavelet thresholding via a Bayesian approach," *J. Roy Statist. Soc. Ser. B*, vol. 60, pp. 725–749, 1998.

[32] J. K. Romberg, H. Choi, and R. G. Baraniuk, "Bayesian tree-structured image modeling using wavelet-domain hidden Markov models," in *Proceedings of SPIE technical conference on Mathematical Modeling, Bayesian Estimation, and Inverse Problems*, Denver, CO, July 1999, vol. 3816, pp. 31–44, Extended version available at `www.dsp.rice.edu`.

[33] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–285, Feb. 1989.

[34] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, Cambridge, MA, 1998.

[35] M. S. Crouse and R. G. Baraniuk, "Simplified wavelet-domain hidden Markov models using contexts," in *Proc. 31st Asilomar Conf.*, Pacific Grove, CA, Nov. 1997.

[36] B. Hendricks, H. Choi, and R. Baraniuk, "Analysis of wavelet-domain multiscale classification using Kullback-Leibler distances," in *Proc. 33st Asilomar Conference*, Pacific Grove, CA, Oct. 24-27, 1999.