# Multiscale Bayesian Segmentation Using a Trainable Context Model *†


Hui Cheng and Charles A. Bouman
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285
{hui,bouman}@ecn.purdue.edu



EDICS Number: 2-WAVP Wavelets and Multiresolution Processing
EDICS Number: 2-SEGM Segmentation



Corresponding Author : Charles A. Bouman
Address : School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285
Phone : +1 765 494-0340
FAX : +1 765 494-3358
Email : bouman@ecn.purdue.edu



**Abstract**

In recent years, multiscale Bayesian approaches have attracted increasing attention for use in image segmentation. Generally, these methods tend to offer improved segmentation accuracy with reduced computational burden. Existing Bayesian segmentation methods use simple models of context designed to encourage large uniformly classified regions. Consequently, these context models have a limited ability to capture the complex contextual dependencies that are important in applications such as document segmentation.

In this paper, we propose a multiscale Bayesian segmentation algorithm which can effectively model complex aspects of both local and global contextual behavior. The model uses a Markov chain in scale to model the class labels that form the segmentation, but augments this Markov chain structure by incorporating tree based classifiers to model the transition probabilities between adjacent scales. The tree based classifier models complex transition rules with only a moderate number of parameters.

One advantage to our segmentation algorithm is that it can be trained for specific segmentation applications by simply providing examples of images with their corresponding accurate segmentations. This makes the method flexible by allowing both the context and the image models to be adapted without modification of the basic algorithm. We illustrate the value of our approach with examples from document segmentation in which text, picture and background classes must be separated.



*This project was supported by the Xerox Corporation.

†To appear in the IEEE Trans. on Image Processing.

# 1 Introduction

Image segmentation is an important first step for many image processing applications. For example, in document processing it is usually necessary to segment out text, pictorial and graphic regions before scanned documents can be effectively analyzed, compressed or rendered [1, 2]. Segmentation has also been shown useful for image and video compression [3, 4]. For each of these cases, the objective is to separate images into regions with distinct homogeneous behavior.

In recent years, Bayesian approaches to segmentation have become popular because they form a natural framework for integrating both statistical models of image behavior and prior knowledge about the contextual structure of accurate segmentations. An accurate model of contextual structure can be very important for segmentation. For example, it may be known that segmented regions must have smooth boundaries or that certain classes can not be adjacent to one another.

In a Bayesian framework, contextual structure is often modeled by a Markov random field (MRF) [5, 6, 7]. Usually, the MRF contains the discrete class of each pixel in the image. The objective then becomes to estimate the unknown MRF from the available data. In practice, the MRF model typically encourages the formation of large uniformly classified regions. Generally, this smoothing of the segmentation increases segmentation accuracy, but it can also smear important details of a segmentation and distort segmentation boundaries. Approaches based on MRF's also tend to suffer from high computational complexity. The non-causal dependence structure of MRF's usually results in iterative segmentation algorithms, and can make parameter estimation difficult [8, 9]. Moreover, since the true segmentation is not available, parameter estimation must be done using an incomplete data method such as the EM algorithm [10, 11, 12].

Another long term trend has been the incorporation of multiscale techniques in segmentation algorithms. Methods such as pixel linking [13], boundary refinement [14, 15], and decision integration [16] through pyramid structures have been used to enforce contextual information in the segmentation process. In addition, pyramid [17] or wavelet decompositions [18, 19] yield powerful multiscale features that can capture both local and global image characteristics.

Not surprisingly, there has been considerable interest in combining both Bayesian and multiscale techniques into a single framework. Initial attempts to merge these view points focused on using multiscale algorithms to compute segmentations but retained the underlying fixed scale MRF context model [20, 21, 22]. These researchers found that multiscale algorithms could substantially reduce computation and improve robustness, but the simple MRF context model limited the quality of segmentations.

In [23, 24], Bouman and Shapiro introduced a multiscale context model in which the segmentation was modeled using a Markov chain in scale. By using a Markov chain, this approach avoided many of the difficulties associated with noncausal MRF structures and resulted in a non-iterative segmentation algorithm similar in concept to the forward-backward algorithm used with hidden Markov models (HMM). Laferte, Heitz, Perez and Fabre used a similar approach, but incorporated a multiscale feature model using a pyramid image decomposition [25]. In related work, Crouse, Nowak, and Baraniuk have proposed the use of multiscale HMM's to model wavelet coefficients for applications such as image de-noising and signal detection [26].

In another approach, Kato, Berthod, and Zerubia first used a 3-D MRF as a context model for segmentation [27]. In this model, each class label depends on class labels at both the same scale and the adjacent finer and coarser scales. Comer and Delp used a similar context model but incorporated a 3-D autoregressive feature model [28].

In this paper, we propose an image segmentation method based on the multiscale Bayesian framework. Our approach uses multiscale models for both the data and the context. Once a complete model is formulated, the sequential maximum *a posterior* (SMAP) estimator [24] is used to segment images.

An important contribution of our approach is that we introduce a multiscale context model which can capture complex aspects of both local and global contextual behavior. The method is based on the use of tree based classifiers [29, 30] to model the transition probabilities between adjacent scales in the multiscale structure. This multiscale structure is similar to previously proposed segmentation models [24, 31, 32], with the segmentations at each resolution forming a Markov chain in scale. However, the tree based classifier allows for much more complex transition rules, with only a moderate number of parameters. Moreover, we propose an efficient parameter estimation algorithm for training which is not iterative and only needs one coarse-to-fine recursion through resolutions.

Our multiscale image model uses local texture features extracted via a wavelet decomposition. The wavelet transform produces a pyramid of feature vectors with each three dimensional feature vector representing the texture at a specific location and scale. While wavelet decompositions tend to decorrelate data, significant correlation can remain among wavelet coefficients at similar locations but different scales. In fact, this dependency is often exploited in image coding techniques such as zerotrees [33]. We account for these dependencies by modeling the wavelet feature vectors as a class dependent multiscale autoregressive process [34]. This approach more accurately models some textures without adding significant additional computation.

A unique feature of our segmentation method is that it can be trained for any segmentation application by simply providing examples of images with their corresponding accurate segmentations. We believe that this makes the method flexible by allowing it to be adapted for different segmentation applications without modification of the basic algorithm. The training procedure uses the example images together with their segmentations to estimate all parameters of both the image and context models in a fully automatic manner. [1] Once the model parameters are estimated, segmentation is computationally efficient requiring a single fine-to-coarse-to-fine iteration through the pyramid.

Although our segmentation method is based on the multiscale Bayesian framework introduced in [24], it has several distinctions from the previous approach. First, we employ a more comprehensive context model, and the parameters of the context model are estimated from training images instead of from the image being segmented. Second, we use a multiscale image data model and the Haar basis wavelet coefficients as image data features. In addition, the correlation among wavelet coefficients across adjacent scales is modeled as a class dependent multiscale autoregressive process. This is different from [24] which uses a fixed scale image data model and pixel values of the original

[1] Software implementation of this algorithm is available from http://www.ece.purdue.edu/~bouman.

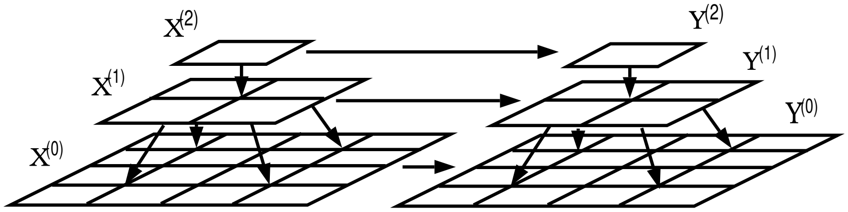



Figure 1: The multiscale segmentation model proposed in this paper. $Y^{(n)}$ contains the image feature vectors extracted at scale $n$ while $X^{(n)}$ contains the corresponding class of each pixel at scale $n$. Notice that both image features, $Y$, and the context model, $X$, use multiscale pyramid structures.

image as image features.

In order to test the performance of our algorithm, we apply it to the problem of document segmentation. This application is interesting because of both its practical significance and the great contextual complexity inherent to modern documents [2]. For example, most documents conform to complex rules regarding the spatial placement of regions such as picture, text, graphics and background. While specifying these rules explicitly would be difficult and error prone, we show that these rules can be effectively learned from a limited number of training examples.

In section 2, we introduce a multiscale image segmentation model and a general form of the SMAP estimate derived from our model. The detailed algorithm for computing the SMAP estimate is discussed in section 3. Section 4 presents the parameter estimation algorithms developed for our model. Experimental results are discussed in section 5, and section 6 concludes this paper.

## 2 Multiscale Image Segmentation

In this paper, we will adopt a Bayesian segmentation approach, but our method differs from many in that we use a multiscale model for both the data and the context. Figure 1 illustrates the basic structure of our multiscale segmentation model [32]. At each scale $n$, there is a random field of image feature vectors, $Y^{(n)}$, and a random field of class labels, $X^{(n)}$.[2] For our application, the image features $Y^{(n)}$ will correspond to Haar basis wavelet coefficients at scale $n$. Intuitively, $Y^{(n)}$ contains image texture and edge information at scale $n$, while $X^{(n)}$ contains the corresponding class labels. The behavior of $Y^{(n)}$ is therefore assumed dependent on its class labels $X^{(n)}$ and coarse

[2]We will use upper case letters to denote random quantities while lower case variables will denote their realizations.

scale image features $Y^{(n+1)}$ as is indicated by the arrows in Figure 1.

Notice that each random field $X^{(n)}$ is assumed dependent only on the previous coarser scale field $X^{(n+1)}$. This dependence gives $X^{(n)}$ a Markov chain structure in the scale variable $n$. We will see that this structure is desirable because it can capture complex spatial dependencies in the segmentation, but it allows for efficient computational processing. The multiscale structure can also account for both large and small scale characteristics that may be desirable in a good segmentation.

For convenience, we define $X^{(\leq n)} = \{X^{(i)}\}_{i=0}^{n}$ to be the set of class labels at scales $n$ or finer, and $X^{(>n)} = \{X^{(i)}\}_{i=n+1}^{L}$ where $L$ is the coarsest scale. We also define $Y^{(\leq n)}$ and $Y^{(>n)}$ similarly. Using this notation, the Markov chain structure may be formally expressed in terms of the probability mass functions

$$p(x^{(n)}|x^{(>n)}) = p(x^{(n)}|x^{(n+1)}) \ . \tag{1}$$

So the probability of $x$ is given by

$$p(x) = \prod_{n=0}^{L} p(x^{(n)}|x^{(n+1)}) \tag{2}$$

where throughout this paper, the term $p(x^{(L)}|x^{(L+1)})$ is assumed to mean $p(x^{(L)})$, since $L$ is the coarsest scale.

The image features $y^{(n)}$ are assumed conditionally independent given the class labels $x^{(n)}$ and image features $y^{(n+1)}$ at the coarser scale. Therefore, the conditional density of $y$ given $x$ may be expressed as

$$p(y|x) = \prod_{n=0}^{L} p(y^{(n)}|x^{(n)}, y^{(n+1)}) \tag{3}$$

Combining equations (2) and (3) results in the joint density

$$\begin{aligned} p(y, x) &= p(y|x)\, p(x) \\ &= \prod_{n=0}^{L} p(y^{(n)}|x^{(n)}, y^{(n+1)}) p(x^{(n)}|x^{(n+1)}) \ . \end{aligned}$$

In order to segment the image, we must estimate the class labels $X$ from the image feature data $Y$. Perhaps the MAP estimator is the most common method for doing this. However, the MAP estimate is not well suited for multiscale segmentation because it results from minimization of a cost functional which equally weights both fine and coarse scale misclassifications. In practice, coarse scale misclassifications are much more important since they affect many more pixels. For example, a misclassification at scale $n$ may affect $2^{n^2}$ pixels at the finest resolution. Because of this, we will adopt the sequential MAP (SMAP) cost function proposed in [24]. Let $C(X, x)$ be the SMAP cost of choosing segmentation $x$ when the true segmentation is $X$. Then, $C(X, x)$ is chosen to be

$$
\begin{aligned}
C(X, x) &= \frac{1}{2} + \sum_{n=0}^{L} 2^{n-1} C_n(X, x) \\
C_n(X, x) &= 1 - \prod_{i=n}^{L} \delta(X^{(i)} - x^{(i)})
\end{aligned}
$$

where $\delta(X^{(i)} - x^{(i)}) = 1$, if $X^{(i)} = x^{(i)}$ and $\delta(X^{(i)} - x^{(i)}) = 0$, if $X^{(i)} \neq x^{(i)}$. Intuitively, this SMAP cost functional assigns more weight to misclassifications at coarser scales, and is therefore more appropriate for application in discrete multiscale estimation problems.

In [24], it was shown that the SMAP estimator resulting from the minimization

$$
\hat{x} = \arg\ \min_{x} E[C(X, x) | Y = y] \tag{4}
$$

can be computed using the following recursive coarse-to-fine relationship

$$
\hat{x}^{(n)} \quad = \quad \arg\max_{x^{(n)}} \left\{ \log p_{y^{(\leq n)}|x^{(n)}, y^{(n+1)}}(y^{(\leq n)} | x^{(n)}, y^{(n+1)}) + \log p_{x^{(n)}|\hat{x}^{(n+1)}}(x^{(n)} | \hat{x}^{(n+1)}) \right\} \tag{5}
$$

where the coarsest segmentation $\hat{x}^{(L)}$ is computed using the conventional MAP estimate. While [24] did not assume the same multiscale data model as is used in this paper, the methods of the proof hold without change. The SMAP estimation procedure is a coarse-to-fine recursion which starts by computing $\hat{x}^{(L)}$, the MAP estimate at the coarsest scale $L$. At each scale $n$, equation (5) is then applied to compute the new segmentation while conditioning on the previous coarser scale segmentation $\hat{x}^{(n-1)}$. Each application of (5) is similar to MAP estimate of the segmentation $x^{(n)}$ since it requires maximization of a data term related to $y^{(\leq n)}$ and a context or prior term related to the probability of $x^{(n)}$ conditioned on the previous coarser segmentation $\hat{x}^{(n+1)}$.

# 3 Computing the SMAP Estimate

In the previous section, we derived a SMAP estimator based on the multiscale image segmentation model illustrated in Figure 1. This segmentation model is a general model. It only defines the global interaction among fields of class labels and fields of image features. In this section, we will specify the interaction among class labels and image features at the pixel level. We will also give specific forms for both the data and the context terms in (5), then use these forms to derive a specific algorithm for the SMAP estimator. In other words, what is discussed in the previous section is the abstract of the algorithm, and what is discussed in this section is the embodiment of the implementation.

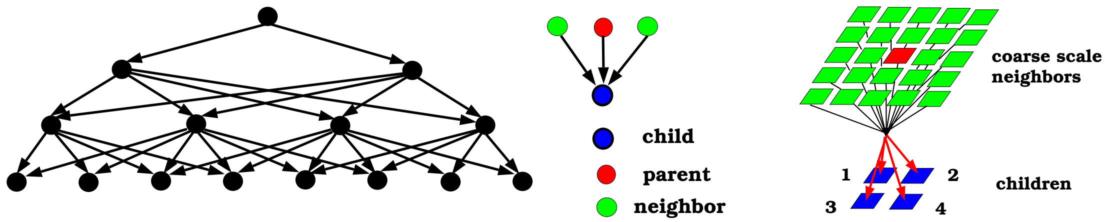
In order to make the computation feasible, some assumptions are made in this section to localize the computation. There are two important assumptions. First, we will assume that the data term of (5) can be expressed as the sum of log likelihood functions at each pixel. We denote individual pixels by $x_s^{(n)}$ and $y_s^{(n)}$, where $s$ is the position in a 2-D lattice $S^{(n)}$. Using this notation, the data term of (5) will have the form

$$\log p(y^{(\leq n)}|x^{(n)}, y^{(n+1)}) = \sum_{s \in S^{(n)}} l_s^{(n)}(x_s^{(n)}) \tag{6}$$

where the functions $l_s^{(n)}(k)$ are appropriately chosen log likelihood functions. Let $|S^{(n)}|$ be the number of pixels at scale $n$. Then, the above assumption can be also interpretated in the statistical sense as follows: we assume that the set of image features $y^{(\leq n)}$ can be partitioned into $|S^{(n)}|$ disjoint subsets which are conditionally independent given the segmentation at scale $n$, $x^{(n)}$ and the image features at scale $n+1$, $y^{(n+1)}$. Section 3.2 will give the details for how to compute these functions $l_s^{(n)}(k)$.

Second, we will assume that the context term of (5) can be expressed as the product of probabilities for each pixel. That is the class labels $x_s^{(n)}$ are assumed conditionally independent given the coarser segmentation $x^{(n+1)}$. Therefore, the context term of (5) will have the form

$$\log p(x^{(n)}|x^{(n+1)} = \hat{x}^{(n+1)}) = \sum_{s \in S^(n)} \log\, p(x_s^{(n)}|\hat{x}^{(n+1)}). \tag{7}$$




(a) (b)

Figure 2: The pyramidal graph model. (a) 1-D analog of the pyramidal graph model, where each pixel has 3 neighbors at the coarser scale. (b) 2-D pyramidal graph model using a $5 \times 5$ neighborhood. This is equivalent to interpolation of a pixel at the previous coarser scale into four pixels at the current scale.

We will discuss how to compute the conditional probabilities $p(x_s^{(n)} = k | x^{(n+1)} = \hat{x}^{(n+1)})$ in section 3.1.

With these two assumptions, the SMAP recursion of (5) can be simplified to a single pass, pixel by pixel update rule

$$\hat{x}_s^{(n)} = \arg \max_{0 \leq k < M} \left\{ l_s^{(n)}(k) + \log p(x_s^{(n)} = k | x^{(n+1)} = \hat{x}^{(n+1)}) \right\} \tag{8}$$

where $M$ is the number of possible class labels.

### 3.1 Computing Context Terms for the SMAP Estimate

Our context model requires that we compute the probability distribution for each pixel $x_s^{(n)}$ given the coarser scale segmentation $x^{(n+1)}$. In order to limit complexity of the model, we will assume that $x_s^{(n)}$ is only dependent on $x_{\partial s}^{(n+1)}$, a set of neighboring pixels at the coarser scale. Here, $\partial s \subset S^{(n+1)}$ denotes a window of pixels at scale $n+1$. We will refer this dependency among class labels as the pyramidal graph model. Figure 2(a) illustrates the pyramidal graph model for the 1-D case where each pixel has 3 neighbors at the coarser scale. Notice that each arrow points from a neighbor in $x_{\partial s}^{(n+1)}$ to a pixel $x_s^{(n)}$.

Intuitively, this context model is also a model for interpolating a pixel $s^{(n+1)}$ into its child pixels. Figure 2(b) illustrates this situation in 2-D when a $5 \times 5$ neighborhood is used at the coarser scale. Notice that in 2-D, each pixel $s^{(n+1)}$ has four child pixels at the next finer resolution. Each of the four child pixels will have the same set of neighbors; however they must be modeled using different distributions, because of their different relative positioning. We denote each of these four

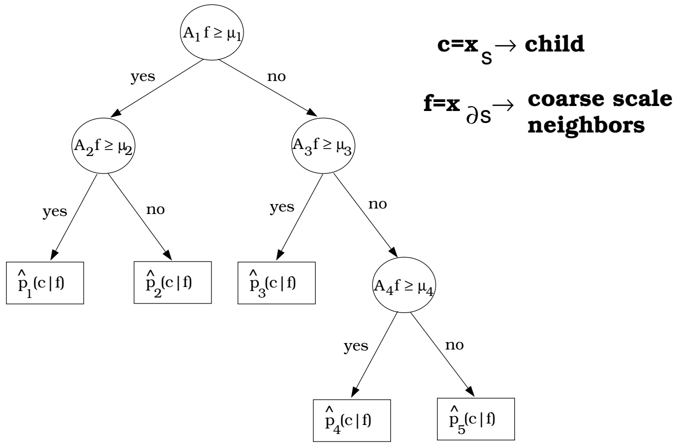



Figure 3: Class probability tree. Circles represent interior nodes, and squares represent leaf nodes. At each interior node, a linear test is performed and the node is split into two child nodes. At each leaf node $\tilde{t}$, the conditional probability mass function $p_i^{(n)}(c|f)$ is approximated by $\hat{p}_{\tilde{t}}(c)$.

distinct probability distributions by $p_i^{(n)}(x_s^{(n)}|x_{\partial s}^{(n+1)})$ for $i = 1, 2, 3, 4$. For simplicity, we will use $c$ to denote $x_s^{(n)}$, and $f$ to denote $x_{\partial s}^{(n+1)}$, so that this probability distribution may be written as $p_i^{(n)}(c|f)$. Later we will see that $c$ and $f$ are actually binary encodings of the information contained in $x_s^{(n)}$ and $x_{\partial s}^{(n+1)}$.

Unfortunately, the transition function $p_i^{(n)}(c|f)$ may be very difficult to estimate if the coarse scale neighborhood is large. For example, if there are four classes and the size of the coarse neighborhood is $5 \times 5$, there are $4^{25} \approx 10^{16}$ possible values of $f$. Hence, it is impractical to compute $p_i^{(n)}(c|f)$ using a look-up table containing all possible values of $f$. For most applications, the distribution of $f$ will be concentrated among a small number of possible values. We can exploit this structure in the distribution of $f$ to dramatically simplify the computation of $p_i^{(n)}(c|f)$.

In order to compute and estimate $p_i^{(n)}(c|f)$ efficiently, we use class probability trees (CPT) [29] to represent $p_i^{(n)}(c|f)$. A CPT is shown in Figure 3. The CPT represents a sequence of decisions or tests that must be made in order to compute the conditional probability of $c$ given $f$. The input to the tree is $f$. At each interior node, a splitting rule is used to determine which of the two child nodes should be taken. In our case, the splitting rule is computed by comparing $A_t f - \mu_t$ to 0, where $A_t$ is a pre-computed vector and $\mu_t$ is a pre-computed scalar. In this way, the tree is traversed moving from the root to a terminal leaf node. Each leaf node $\tilde{t}$ is associated with an empirically computed probability mass function $\hat{p}_{\tilde{t}}(c)$. When $f$ reaches $\tilde{t}$, $p_i^{(n)}(c|f)$ is set to $\hat{p}_{\tilde{t}}(c)$.

If a CPT has $K$ leaf nodes, then the CPT approximates the true transition probability using $K$ probability mass functions. Therefore, by controlling the number of leaf nodes in a CPT, even for a relative large neighborhood, such as a $7 \times 7$ neighborhood, we can still estimate the transition probabilities efficiently and accurately. Since a larger neighborhood usually gives more contextual information, CPT's allow us to work with a larger neighborhood and consequently have a better model of the context, while retaining computational efficiency in our model. In section 4.1, we will give specific methods for building a CPT from training data.

To achieve the best accuracy from the CPT algorithm, we have found that proper encoding of the quantities $x_s^{(n)}$ and $x_{\partial s}^{(n+1)}$ into $c$ and $f$ is important. Specifically, the encoding should not impose any ordering on the $M$ class labels because an ordering imposed on the class labels combined with the matrix operation, $A_t f - \mu_t$, used in the splitting rule could bias the results and consequently degrade the classification accuracy. For example, if we denote text class as 1, picture class as 2 and background class as 3, it would imply that the background class is closer to the picture class that it is to text class. However, if we use $[1,0,0]^t$ to denote text class, $[0,1,0]^t$ to denote picture class, and $[0,0,1]^t$ to denote background class, it will give equal distance among the 3 classes. Therefore, we use binary encoding of class labels. We define $c$ to be a binary vector of length $M$ where the $x_s^{(n)}$-th component of $c$ is 1, and other components are 0. If we denote the $i$-th component of $c$ as $c_j$, then

$$c_j = \begin{cases} 1 & \text{if } x_s^{(n)} = j \\ 0 & \text{otherwise} \end{cases} \quad 0 \leq j < M \ .$$

For example, when $x_s^{(n)} = 2$, and $M = 4$, then $c = (0,0,1,0)$. Similarly, we define $f$ to be a binary vector of length $Mb$, where $b$ is the number of pixels in the coarse neighborhood $\partial s$. The binary vector $f$ is then formed by concatenating the binary encodings of each coarse scale neighbor contained in $x_{\partial s}^{(n+1)}$.

In addition, we assume the prior distributions of class labels at the coarsest resolution to be i.i.d. uniform. In practice, we have always observed that the data term dominates the context term at the coarsest resolution. Therefore, the specific choice of the prior distribution generally has no

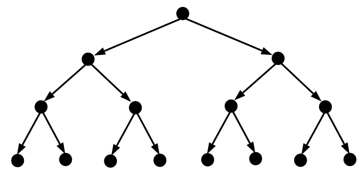


Figure 4: 1-D analog of the quadtree model.

significant effect on the segmentation result.

### 3.2 Computing Log Likelihood Terms for SMAP Estimate

In order to capture the correlation among image features across scales, we assume that each feature $y_s^{(n)}$ depends on both an image feature $y_{\partial s}^{(n+1)}$ at the coarser scale and its class label $x_s^{(n)}$, where $\partial s$ is the parent of $s$. We assume that, for each class $x_s^{(n)}$, $y_s^{(n)}$ can be predicted by a different linear function of $y_{\partial s}^{(n+1)}$ which depends on both the class label and the scale. We denote the prediction error by $\tilde{y}_s^{(n)}$.

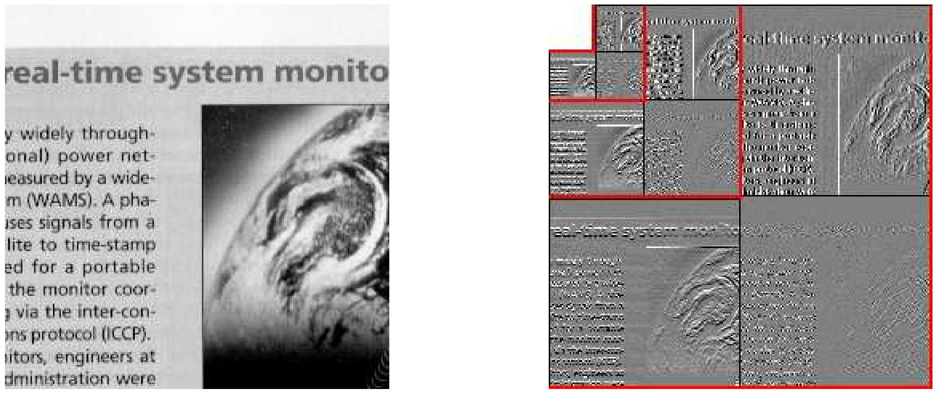
$$\tilde{y}_s^{(n)} = y_s^{(n)} - \left[\alpha_{x_s}^{(n)} y_{\partial s}^{(n+1)} + \beta_{x_s}^{(n)}\right] \tag{9}$$

where $\alpha_{x_s}^{(n)}$ and $\beta_{x_s}^{(n)}$ are prediction coefficients which are functions of both class labels and scales.

To have an efficient algorithm for computing the log likelihood terms $l_s^{(n)}(k)$ defined in equation (6), we assume that the prediction errors $\tilde{y}_s^{(n)}$ are conditionally independent given the class labels $x_s^{(n)}$. That is

$$\begin{aligned} \log\, p(y^{(n)}|x^{(n)}, y^{(n+1)}) &= \log\, p(\tilde{y}^{(n)}|x^{(n)}) \\ &= \sum_{s \in S^{(n)}} \log\, p(\tilde{y}_s^{(n)}|x_s^{(n)}) \ . \end{aligned}$$

To calculate the log likelihood terms, we also need to compute the conditional probability distribution of $x_s^{(n)}$ given $x^{(n+1)}$. But we can not use the pyramidal graph model discussed in section 3.1, because it will result in a form which is not computationally tractable. Therefore, we use a context model which is simpler than the pyramidal graph model. In this model, we assume that $x_s^{(n)}$ depends only on one class label at the previous coarser resolution. Though we still use $x_{\partial s}^{(n+1)}$ to denote the class label which $x_s^{(n)}$ depends on, this time, $\partial s$ is a set containing only one pixel at scale $n+1$. This simple dependency among class labels is often referred to as the quadtree




(a) (b)

Figure 5: Haar basis wavelet decomposition. (a) Original image. (b) Illustration of wavelet coefficients of a three level wavelet decomposition using Haar basis.

model [24, 32], and its 1-D analog is shown in Figure 4. We further reduce the computation by assuming that each of the four children have the same probability distribution. Therefore, we replace the four distinct distributions used in the pyramidal graph model with a single distribution. We will denote the probability mass function for each child by $\theta_{k,m,n} = p(x_s^{(n)} = k | x_{\partial s}^{(n+1)} = m)$ where $0 \leq k, m < M$ and $0 \leq n < L$. Since $\theta_{k,m,n}$ has at most $M^2$ distinct values for each scale $n$, we will use a look up table to represent this probability distribution.

In Appendix A, we use these assumptions to derive the following formula for computing the log likelihood terms

$$l_s^{(0)}(k) = \log p(\tilde{y}_s^{(0)} | x_s^{(0)} = k) \tag{10}$$

$$l_s^{(n)}(k) = \log p(\tilde{y}_s^{(n)} | x_s^{(n)} = k) + \sum_{i=1}^{4} \log \left\{ \sum_{m=0}^{M-1} \exp\left[ l_{s_i}^{(n-1)}(m) \right] \theta_{m,k,n-1} \right\} \tag{11}$$

where $s_i$ $(i = 1, 2, 3, 4)$ are the four children of $s$. Using (10) and (11), the log likelihood terms can be computed using a fine-to-coarse recursion through scales. First, the log likelihood term at the finest scale, $n = 0$, is calculated by applying equation (10). Then the log likelihood at the next coarser scale is computed with (11) for $n = 1$. This process is repeated until the coarsest scale is reached.

In our model, the feature vector at each pixel $y_s$ is formed using the coefficients of a Haar basis wavelet decomposition. While the Haar basis is not very smooth, it is very computationally efficient to implement and does a good job of extracting useful feature vectors. The wavelet transform results in three bands at each resolution, which are often referred to at the low-high, high-low, and high-high bands. Among the three bands, the low-high and the high-low bands are closely related to horizontal and vertical edges, respectively. An example of the Haar basis wavelet decomposition of a document image is shown in Figure 5. Figure 5(a) is a portion of a scanned document image. Figure 5(b) is the image of wavelet coefficients of a three level Haar basis wavelet decomposition, where bright pixels denote large positive coefficients, and dark pixels denote negative coefficients with large amplitudes. Because of the structure of the wavelet transform, each of these bands has half the spatial resolution of the original image. Each feature vector $y_s^{(n)}$ in our pyramid is then a three dimensional vector containing components from each of these three bands extracted at the same position in the image. Using this structure, the finest resolution of the pyramid has only half the resolution of the original image.

The conditional probability distribution of the feature vector's prediction error, $p(\tilde{y}_s^{(n)}|x_s^{(n)} = k)$ can be modeled using a variety of statistical methods. In our approach, we use the multivariate Gaussian mixture model [35]

$$p(\tilde{y}_s^{(n)} = \tilde{y}|x_s^{(n)} = k) = \sum_{j=1}^{J_{k,n}} \gamma_{j,k,n} \frac{1}{(2\pi)^{3/2}|C_{j,k,n}|^{1/2}} \exp\left[-\frac{1}{2}(\tilde{y} - \mu_{j,k,n})^t C_{j,k,n}^{-1}(\tilde{y} - \mu_{j,k,n})\right] \qquad (12)$$

where $J_{k,n}$ is the order of the Gaussian mixture for class $k$ and scale $n$; and $\mu_{j,k,n}$, $C_{j,k,n}$, and $\gamma_{j,k,n}$ are the mean, covariance matrix, and weighting associated with the $j$-th component of the Gaussian mixture for class $k$ and scale $n$. In general, $C_{j,k,n}$ will be positive definite, and $\gamma_{j,k,n} \in [0,1]$ with $\sum_{j=1}^{J_{k,n}} \gamma_{j,k,n} = 1$. For large $J_{k,n}$, the Gaussian mixture density can approximate any probability density.

### 3.3 Algorithm for Computing the SMAP Estimate

Once the model parameters are estimated, the SMAP estimate discussed in section 2 and section 3 can be computed using the following algorithm:

1. Feature extraction: perform $L$ level Haar basis wavelet decomposition of the input image.

2. For the finest resolution, compute $l_s^{(0)}(k)$ according to (10) and (12) for all $s \in S^{(0)}$ and $0 \leq k < M$.

3. Fine-to-coarse recursion to compute $l_s^{(n)}(k)$:

   (a) Set $n = 1$.

   (b) Compute $l_s^{(n)}(k)$ according to (11) and (12) for all $s \in S^{(n)}$ and $0 \leq k < M$.

   (c) If $n < L$, $n = n + 1$, and go to step 3b. Otherwise, go to step 4.

4. For the coarsest resolution, compute $\hat{x}_s^{(L)}$ for all $s \in S^{(L)}$ as follows:

$$\hat{x}_s^{(L)} = \arg \max_{0 \leq k < M} l_s^{(L)}(k) \ .$$

5. Coarse-to-fine recursion to compute $\hat{x}_s^{(n)}$:

   (a) Set $n = L - 1$.

   (b) Compute $\hat{x}_s^{(n)}$ according to (8) for all $s \in S^{(n)}$.

   (c) If $n > 0$, $n = n - 1$, and go to step 5b. Otherwise, stop.

# 4 Parameter Estimation

The SMAP segmentation algorithm described above depends on the selection of a variety of parameters that control the modeling of both data features and the context model. This section will explain how these parameters may be efficiently estimated from training data. The training data consists of a set of images together with their correct segmentations at the finest scale. This training data is then used to model both the texture characteristics and contextual structure of each region. The training process is performed in four steps:

1. Estimate quadtree model parameters $\theta_{m,k,n}$ used in equation (11).

2. Decimate (subsample) the ground truth segmentations to form ground truth at all scales.

3. Estimate the Gaussian mixture model parameters of (12).

4. Estimate the coarse-to-fine transition probabilities $p_i^{(n)}(c|f)$ used in equation (8) by building an optimized class probability tree (CPT).

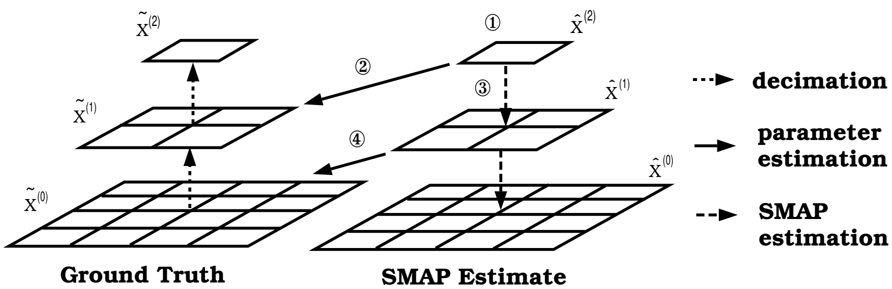



Figure 6: Parameter estimation of the context model. (1) Compute the segmentation at the coarsest resolution, $\hat{x}^{(2)}$. (2) Estimate the transition probabilities $p_i^{(1)}(c|f)$ using the SMAP segmentation $\hat{x}^{(2)}$ and the decimated ground truth segmentation $\tilde{x}^{(1)}$. (3) Compute $\hat{x}^{(1)}$ using $p_i^{(1)}(c|f)$. (4) Estimate $p_i^{(0)}(c|f)$ using $\hat{x}^{(1)}$ and $\tilde{x}^{(0)}$. This procedure is then repeated for all scales.

Perhaps the most important and difficult part of parameter estimation is step 4. This step estimates the parameters of the context model by observing the coarse-to-fine transition rates in the training data. Step 4 is a difficult incomplete data problem because we do not have access to the unknown class labels $X^{(n)}$ at all scales. One simple solution would be to estimate $p_i^{(n)}(c|f)$ from the subsampled ground truth labels computed in step 2. However, training from subsampled ground truth leads to biased estimates $p_i^{(n)}(c|f)$ that will result in excessive noise sensitivity in the SMAP segmentation. Alternatively, we have investigated the use of the EM algorithm together with Monte Carlo Markov chain techniques to compute unbiased estimates of the parameters [31]. While this methodology works, it is very computationally expensive and impractical for use with large sets of training data.

Our solution to step 4 is a novel coarse-to-fine estimation procedure which is computationally efficient and non-iterative, but results in accurate parameter estimates. The details of our method are explained in the following section 4.1.

Estimation of quadtree model parameters is discussed in section 4.2. The resulting quadtree model is then used to decimate the ground truth segmentation, so that ground truth is available at all scales. The resulting ground truth is then used to estimate Gaussian mixture model parameters using a well known clustering approach based on the EM algorithm.

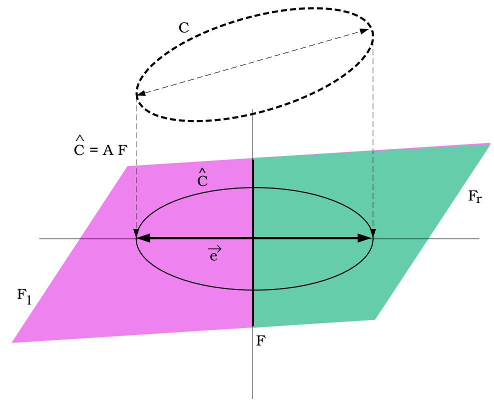



Figure 7: Splitting rule based on the least squares estimation. The dash ellipse represents the covariance matrix of $C$ and the solid ellipse represents the covariance matrix of $\hat{C}$, where $\hat{C}$ is the least squares estimate of $C$. $\vec{e}$ is the principle axis of the covariance matrix of $\hat{C}$. $F$ is split into $F_r$ and $F_l$ according to the axis perpendicular to $\vec{e}$.

### 4.1 Estimation of Context Model Parameters

Our context model is parameterized by the transition probabilities $p_i^{(n)}(c|f)$. Here $f$ is a binary encoding of the coarse scale neighbors $X_{\partial s}^{(n)}$, and $c$ is a binary encoding of the unknown pixel $X_s^{(n)}$. Notice that a different transition distribution is separately estimated for each scale, $n$, and for each of the four children $i$. This is important since it allows the model to be both scale and orientation dependent.

Our procedure for estimating the transition probabilities $p_i^{(n)}(c|f)$ is illustrated in Figure 6. The method works by estimating the transition probabilities from the coarser scale SMAP segmentation $\hat{x}^{(n+1)}$ to the correct ground truth segmentation denoted by $\tilde{x}^{(n)}$. Importantly, $\hat{x}^{(n+1)}$ does not depend on the transition probabilities $p_i^{(n)}(c|f)$. This can be seen from (5), the equation for computing the SMAP segmentation. This is a crucial fact since it allows $\hat{x}^{(n+1)}$ to be computed before $p_i^{(n)}(c|f)$ is estimated. Once $p_i^{(n)}(c|f)$ is estimated, it is then used to compute $\hat{x}^{(n)}$, allowing the estimation of $p_i^{(n-1)}(c|f)$. This process is then recursively repeated until the transition parameters at all scales are estimated.

In our approach, class probability trees are used to represent $p_i^{(n)}(c|f)$, so the ground truth $\tilde{x}^{(n)}$ and segmentation $\hat{x}^{(n+1)}$ will be used to construct and train the tree at each scale $n$ and for each of the four child pixels $i = 1, 2, 3, 4$. We design the tree using the recursive tree construction (RTC) algorithm proposed by Gelfand, Ravishankar, and Delp [30], together with a multivariate splitting rule based on the least squares estimation. We have found that this method is very robust and yields tree depths that produce accurate segmentations. Determining the proper tree depth is very important because a tree that is too deep will over parameterize the model, but a tree that is too shallow will not properly characterize the contextual structure of the training data.

The RTC algorithm works by partitioning the sample set into two halves. Initially, a tree is grown using the first partition, and then the tree is pruned using the second partition. Next the roles of the two partitions are swapped, with the second partition used for growing and the first partition used for pruning. This process is repeated, with partitions alternating roles, until the tree converges. At each iteration, the tree is pruned to minimize the misclassification probability on the data partition not being used for growing the tree.

In order to use the RTC algorithm, we must choose a method for growing the tree. Tree growing is done using a recursive splitting method. This method, illustrated in Figure 7, is based on a multivariate splitting procedure. First, the coarse scale neighbors, $f$, are used to compute $\hat{c}$, the least squares estimate of $c$. Then the values of $\hat{c}$ are split into two sets about the mean and along the direction of the principal eigenvector. The multivariate nature of the splitting procedure is very important because it allows clusters of $f$ to be separated out efficiently.

More specifically, let $t$ be the node being split into two nodes. We will assume that $N$ samples of the training data pass into node $t$, so each sample of training data consists of the desired class label, $c_n$, and the coarse scale neighbors, $f_n$ where $n = 1, \cdots, N$. Both $c_n$ and $f_n$ are binary encoded column vectors. Let $\mu_c$ and $\mu_f$ be the sample means for the two vectors

$$\begin{aligned} \mu_c &= \frac{1}{N} \sum_{n=1}^{N} c_n \\ \mu_f &= \frac{1}{N} \sum_{n=1}^{N} f_n \end{aligned}$$

We may then define the matrices

$$\begin{aligned} C &= [c_1 - \mu_c, c_2 - \mu_c, \ldots, c_N - \mu_c] \\ F &= [f_1 - \mu_f, f_2 - \mu_f, \ldots, f_N - \mu_f] \end{aligned}$$

The least squares estimate of $C$ given $F$ is then

$$\hat{C} = [CF^t(FF^t)^{-1}]F \ .$$

Let $\vec{e}$ be the principal eigenvector of the covariance matrix $R = \hat{C}\,\hat{C}^t$. Then our splitting rule is : if $A_t f - \mu_t \geq 0$, $f$ goes to the left child of $t$; otherwise, $f$ goes to the right child of $t$, where

$$\begin{aligned} A_t &= \vec{e}^{\,t} C F^t (FF^t)^{-1} \\ \mu_t &= A_t\,\mu_f \ . \end{aligned}$$

At each step, we split the node which results in the largest decrease in entropy for the tree. This is done by splitting all the candidate nodes in advance and computing the entropy reduction for each node.

### 4.2 Estimation of Quadtree Parameters

The quadtree model is parameterized by the transition probabilities $p(x_s^{(n)} = k | x_{\partial s}^{(n+1)} = m) = \theta_{k,m,n}$, where $x_s^{(n)} = k$ and $x_{\partial s}^{(n+1)} = m$. As with the context model parameters, estimation of the parameters $\theta_{k,m,n}$ is an incomplete data problem because the true segmentation classes are not known at each scale. However, in this case the EM algorithm [36] can be used to solve this problem in a computationally efficient way.

For our problem, the EM algorithm can be written as the following iterative procedure.

$$\theta^{(j+1)} = \arg\max_{\theta} E\left[\log p(X^{(>0)}|\theta) \mid \tilde{x}^{(0)}, \theta^{(j)}\right] \tag{13}$$

where $\theta^{(j)}$ are the estimated quadtree parameters at iteration $j$, and $\tilde{x}^{(0)}$ is the ground truth segmentation at the finest resolution. Using our model, the maximization in (13) has the following solution.

$$\theta_{k,m,n}^{(j+1)} = \frac{\sigma_{k,m,n}^{(j)}}{\sum_{l=0}^{M-1} \sigma_{l,m,n}^{(j)}} \tag{14}$$

where $\sigma_{k,m,n}^{(j)}$ is defined as the following.

$$\sigma_{k,m,n}^{(j)} = \sum_{s \in S^{(n)}} p(x_s^{(n)} = k, x_{\partial s}^{(n+1)} = m \mid \tilde{x}^{(0)}, \theta^{(j)})$$

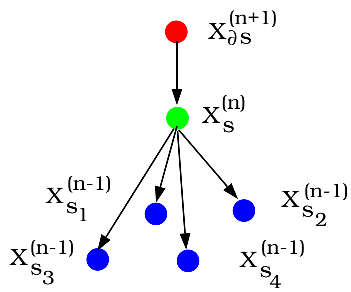


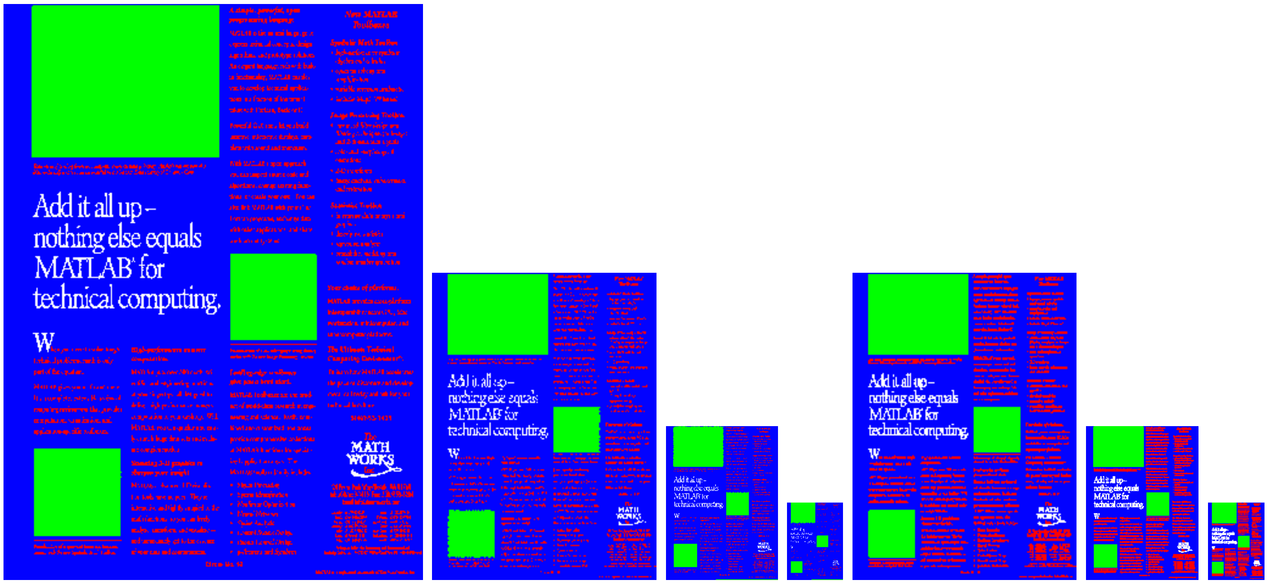

Figure 8: Dependency among class labels in the quadtree model. Given class labels at all pixels except $x_s^{(n)}$, $x_s^{(n)}$ only depends on class labels of its parent, $x_{\partial s}^{(n+1)}$, and four children, $x_{s_i}^{(n-1)}$.

The conditional probabilities $p(x_s^{(n)} = k, x_{\partial s}^{(n+1)} = m \mid \tilde{x}^{(0)}, \theta^{(j)})$ can be computed using either a recursive formula [37, 38] or stochastic sampling techniques. The recursive formulations have the advantage of giving exact update expressions for (13). However, we have found that for this application stochastic sampling methods are easily implemented and work well.

The stochastic sampling approach requires two steps. First, samples of $X^{(>0)}$ are generated using the Gibbs sampler [39]. Then, $\sigma_{k,m,n}^{(j)}$ is estimated using the histogram of the samples. For the quadtree model, the Gibbs sampler can be easily implemented, because the class label of a pixel, $x_s^{(n)}$ only depends on the class label of its parent $x_{\partial s}^{(n+1)}$ and the class labels of its four children $x_{s_i}^{(n-1)}$ (see Figure 8). The detailed algorithm for stochastic sampling is given in Appendix B.

### 4.3 Decimation of Ground Truth Segmentation

After the quadtree models are estimated, we will use them to decimate the fine resolution ground truth to form ground truth segmentations at all resolutions. Importantly, simple decimation algorithms do not give the best results. For example, simple majority voting tends to smear or remove fine details of a segmentation. Figure 9(a) is a ground truth segmentation, and the decimated segmentations using the majority voting are shown in Figure 9(b). Clearly, most of the fine details, such as text lines, and captions are removed by repeated decimation. To address this problem, we will use a decimation algorithm based on the maximum likelihood (ML) estimation. Figure 9(c) shows the results using our ML approach. Notice that the fine details are well preserved in Figure 9(c).




(a) (b) (c)

Figure 9: The ground truth image and decimated ground truth images for n=0,1,2. (a) Ground truth segmentation. (b) Decimated ground truth segmentations using majority voting. (c) Decimated ground truth segmentations using ML estimate.

Our ML estimate of the ground truth at scale $n$ is given by

$$\tilde{x}^{(n)} = \arg\max_{x^{(n)}} p(\tilde{x}^{(0)}|x^{(n)}) \ .$$

This can be easily computed by first computing log likelihood terms in a fine-to-coarse recursion as in equations (10) and (11).

$$\begin{aligned}
\tilde{l}_s^{(1)}(k) &= \sum_{i=1}^{4} \log\, \theta_{\tilde{x}_{s_i}^{(0)},k,0} \\
\tilde{l}_s^{(n)}(k) &= \sum_{i=1}^{4} \log \left\{ \sum_{m=0}^{M-1} \exp\left[ \tilde{l}_{s_i}^{(n-1)}(m) \right] \theta_{m,k,n-1} \right\}
\end{aligned}$$

and then selecting the class label which maximizes the log likelihood at each pixel.

$$\tilde{x}_s^{(n)} = \arg \max_{0 \le k \le M-1} \tilde{l}_s^{(n)}(k)$$

### 4.4 Estimation of Data Model Parameters

In section 3.2, we have used the Gaussian mixture model of equation (12) to approximate the conditional probability distribution $p(\tilde{y}_s^{(n)} = \tilde{y}|x_s^{(n)} = k)$. The EM algorithm is a standard algorithm for estimating parameters of a mixture model [35, 36]. We use the EM algorithm to estimate the means $\mu_{j,k,n}$, the covariance matrices $C_{j,k,n}$, and the weights $\gamma_{j,k,n}$ for each Gaussian mixture density. The model order $J_{k,n}$ is chosen for each class $k$ using the Rissanen criteria [40]. Training data set are generated using the feature vectors $y^{(n)}$ and ground truth segmentation $\tilde{x}^{(n)}$. The prediction coefficients defined in (9) are estimated from training data using the standard least squares estimation.

## 5 Simulation Results

In this section, we apply our segmentation algorithm to the problem of document segmentation. Document segmentation is a interesting test case for the algorithm because documents have complex contextual structure which can be exploited to improve segmentation accuracy. In addition, multiscale features are important for documents since regions such as text, picture, and background can only be accurately distinguished by using texture features at both small and large scales. For a review of document segmentation algorithms, one can refer to [2]. To distinguish our algorithm from the SMAP algorithm proposed in [24], we will call our algorithm the trainable SMAP (TSMAP) algorithm.

The TSMAP algorithm is tested on a database of 50 grayscale document images scanned at 100dpi (dots per inch) on an low cost 32 bits flat-bed scanner. We use the scanned images as they are with no pre-processing. In some cases, the images contain “ghosting” artifacts when pictures and text on the back of a document image can “bleed through” during the scanning process. The database of 50 images was partitioned into 20 training images and 30 testing images. Each of the 20 training images was manually segmented into three classes: text, picture and background. These segmentations were then used as ground truth for parameter estimation. Four training images and their associated ground truth segmentations are shown in Figure 10. The proposed algorithm is coded in C and runs on a 100 MHz Hewlett-Packard model 755 workstation. On the average, it takes around 40 seconds to segment an 850 by 1,100 image (an 8.5in by 11in page scanned at 100dpi).

In our experiments, we allowed a maximum of 8 resolution levels where level 0 is the finest

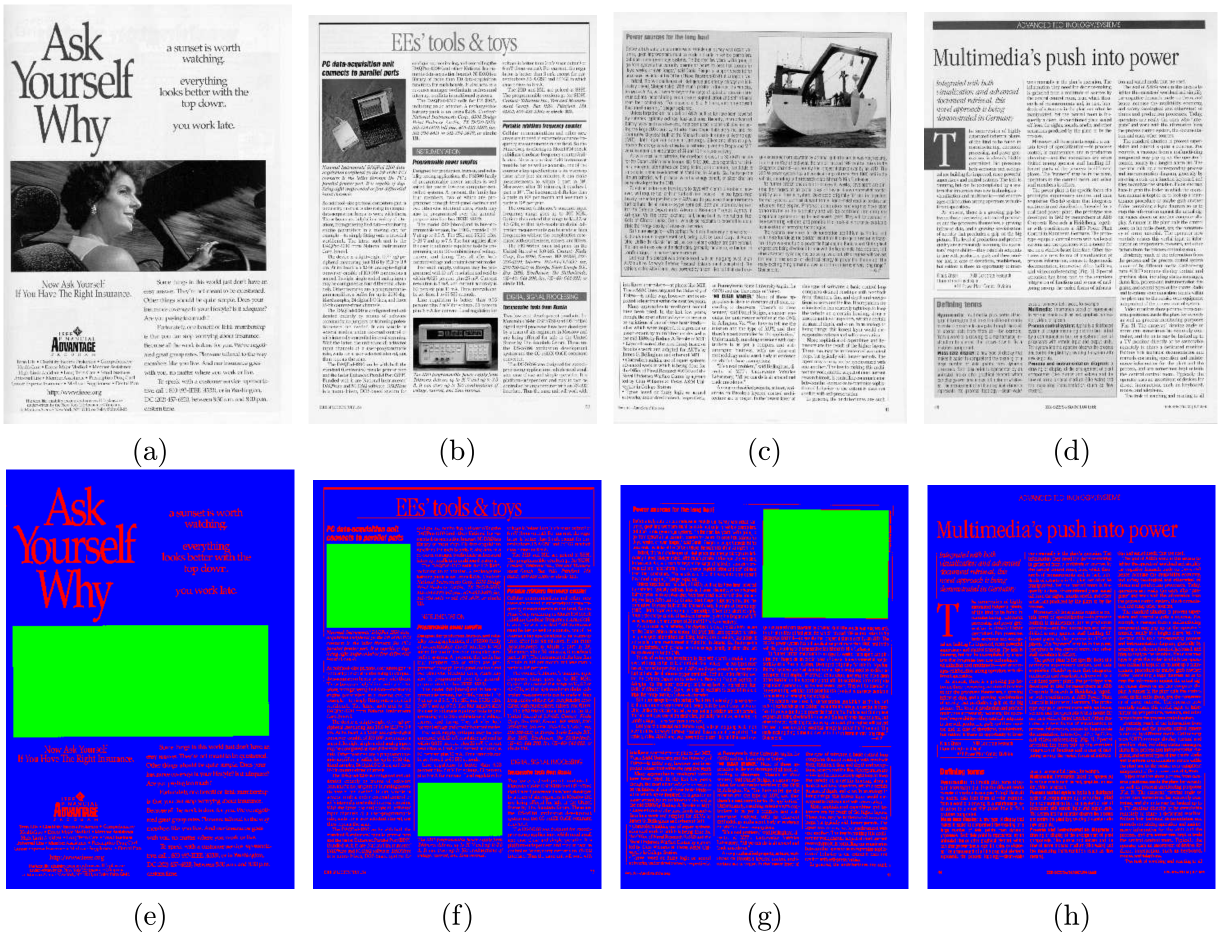



Figure 10: Training images and their corresponding ground truth segmentations: (a)-(d) are training images, and (e)-(h) are ground truth segmentations. Red, green, blue represent text, picture, and background, respectively.

resolution, and level 7 is the coarsest. For each resolution, prediction errors were modeled using the Gaussian mixture model discussed in section 3.2. Each Gaussian mixture density contained 15 or fewer mixture components. Unless otherwise stated, a $5 \times 5$ coarse neighborhood was used. We found that this neighborhood size gave the best overall performance while minimizing computation. For all our segmentation results, we use "red", "green", and "blue" to represent text, picture and background regions respectively.

Figure 11 illustrates the segmentation of a document image in the testing set. Figure 11(a) is the original image, Figure 11(b) shows the result of segmentation using the proposed segmentation algorithm, referred as TSMAP algorithm, with a $5\times5$ coarse scale neighborhood, Figure 11(c) shows the segmentation using TSMAP with a $1 \times 1$ coarse scale neighborhood, and Figure 11(d) shows the segmentation using only the finest resolution features combined with the simple Markov random field as the context model. The MRF uses an eight-point neighborhood system, and its parameters are manually adjusted for the best results. Notice that the results using the simple MRF model is only used to give reader a baseline comparison. Figures 12-13 show the segmentation results for another 8 images outside the training set using TSMAP segmentation with a $5 \times 5$ neighborhood.

Notice that the larger $5 \times 5$ neighborhood substantially improves the accuracy of segmentation when compared to the $1 \times 1$ neighborhood. This is because the large neighborhood can more accurately account for large scale contextual structure in the image. For the $5 \times 5$ neighborhood, the "image" regions are enforced to be uniform, while "text" regions are allowed to be small with fine detail. Even single text lines, reverse text (white text on dark background) and page numbers are correctly segmented. The algorithm also works robustly in the presences of different types of background. For example, white paper and halftoned color background have different textual behavior, but the model allows them to both be handled correctly. The result produced using a simple MRF prior model is much poorer. This is not surprising since the simple MRF prior model can not capture the structure of a document image. Regions between text lines are frequently misclassified and edges of the picture regions are quite irregular. Of course, a more complex MRF can be used. However, an MRF with a large neighbor can make parameter estimation difficult.

From the TSMAP segmentations shown in Figure 11, 12 and 13, we notice that boundaries of two color regions and boundaries between pictures and background are often classified as text. These happened because the likelihood of an edge pixel belonging to text class is so high that the log likelihood term in (8) dominates the classification process. This is also the reason why the middle portion of thick text strokes is often classified as background. For our application, document compression [41], these kinds of segmentations are desirable because text regions are compressed in a way designed for coding edge information, and background is compressed in a different way which is efficient for coding uniform regions. Also, in our training set, there are ground truth segmentations which classify boarders of pictures as text. However, for other applications, classifying picture boundaries as text or classifying the middle part of thick text strokes as background might be un-desirable. In these cases, we believe that larger coarse neighborhood and larger training set would be important to achieve the desired segmentations.

Figure 14 shows the effect of the training set size on the quality of the resulting segmentation. The TSMAP algorithm with a $5 \times 5$ coarse scale neighborhood is trained on three training sets which consist of 20, 10, 5 training images, respectively. The resulting segmentations are shown in Figure 14(c)-(h). Notice that the segmentation quality degrades as the number of training images is decreased, but that good results are obtained with as few as 10 training images. However, when the number of training images is too small, such as 5, the segmentation results (see Figure 14(g)-(h)) can become unreliable.

# 6 Conclusion

We propose a new approach to multiscale Bayesian image segmentation which allows for accurate modeling of complex contextual structure. The method uses a Markov chain in scale to model both the texture features and the contextual dependencies for the image. In order to capture the complex dependencies, we use a class probability tree to model the transition probabilities of the Markov chain. The class probability tree allows us to use a large neighborhood of dependencies while simultaneously limiting the number of parameters that must be estimated. We also propose a novel training technique which allows the context model parameters to be efficiently estimated in a noniterative coarse-to-fine procedure.

In order to test our algorithm, we apply it to the problem of document segmentation. This problem is interesting both because of its practical significance and because the contextual structure of documents is complex. Experiments with scanned document images indicate that the new approach is computationally efficient and improves the segmentation accuracy over fixed scale Bayesian segmentation methods.

## A Computing Log Likelihood Terms

In this appendix, we will derive the recursive formulas for computing $l_s^{(n)}(k)$ which are given in (10) and (11). For a pixel $s \in S^{(n)}$, we define $z_s$ as the set of pixels which consists of $s$ and its descendents. If we assume the quadtree context model and let

$$l_s^{(n)}(k) = \log\, p(\tilde{y}_{z_s} | x_s^{(n)} = k) \ , \tag{15}$$

then it is easy to verify that (6) holds. When $n \geq 1$, we have

$$\begin{aligned} l_s^{(n)}(k) &= \log\, p(\tilde{y}_{z_s} | x_s^{(n)} = k) \\ &= \log\, p(\tilde{y}_s^{(n)} | x_s^{(n)} = k) \\ &\quad + \sum_{i=1}^{4} \log \left[ \sum_{m=0}^{M-1} p(\tilde{y}_{z_{s_i}} | x_{s_i}^{(n-1)} = m)\, p(x_{s_i}^{(n-1)} = m | x_s^{(n)} = k) \right] \\ &= \log p(\tilde{y}_s^{(n)} | x_s^{(n)} = k) + \sum_{i=1}^{4} \log \left\{ \sum_{m=0}^{M-1} \exp\left[ l_{s_i}^{(n-1)}(m) \right] \theta_{m,k,n-1} \right\} \end{aligned}$$

where $s_i$ for $i = 1, 2, 3, 4$ are the four children of $s$. This shows that (11) is true.

When $n = 0$, $s \in S^{(0)}$ and $z_s = \{s\}$. Then (15) can be rewritten as

$$l_s^{(0)}(k) = \log\, p(\tilde{y}_s^{(0)} = \tilde{y}_s^{(0)} | x_s^{(0)} = k) \ .$$

This verifies that (10) is true.

## B Computation of EM Update Using Stochastic Sampling

To compute the EM update using stochastic sampling, the parameters are first initialized to

$$\theta_{i,j,n}^{(0)} = \begin{cases} 0.7 & if \ \ i = j \\ 0.3/(M-1) & if \ \ i \neq j \end{cases}$$

and then we generate samples of $X^{(>0)}$ using a Gibbs sampler [39]. Notice that in the quadtree model, $x_s^{(n)}$ depends only on $x_{\partial s}^{(n+1)}$ and $x_{s_i}^{(n-1)}$, where $s_1$, $s_2$, $s_3$, and $s_4$ are the four children of $s$ (see Figure 8). Therefore, at iteration $j+1$, a sample of $x_s^{(n)}$ can be generated from the conditional probability distribution

$$p(x_s^{(n)} = k | x_{\partial s}^{(n+1)} = m, x_{s_i}^{(n-1)}) = \frac{h_s^{(j)}(k, m, n)}{\sum_{l=0}^{M-1} h_s^{(j)}(l, m, n)}$$

where

$$h_s^{(j)}(k,m,n) = \theta_{k,m,n}^{(j)} \prod_{i=1}^{4} \theta^{(j)}_{x_{s_i}^{(n-1)},k,n-1} .$$

The Gibbs samples are generated from fine to coarse scales. At each scale, we perform $\lfloor 1.5^n \rfloor$ passes through the samples, so that we only do one pass at the finest scale. Each update of the EM algorithm uses two full fine-to-coarse passes of the Gibbs sampler. After the samples are generated, $\sigma_{k,m,n}^{(j)}$ is estimated by histogramming the $x_s^{(n)}$ results from the two passes of the Gibbs sampler.

$$\sigma_{k,m,n}^{(j)} = \sum_{s \in S^{(n)}} \delta(x_s^{(n)} - k, x_{\partial^{(n)} s} - m)$$

# References


[1] K. Y. Wong, R. G. Casey, and F. M. Wahl, "Document analysis system," *IBM J. of Res. & Develop.*, vol. 26, no. 6, pp. 647–656, November 1982.

[2] R. M. Haralick, "Document image understanding: Geometric and logical layout," *Proc. of IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, vol. 8, June 21-23 1994, Seattle, WA, pp. 385–390.

[3] X. Wu and Y. Fang, "A segmentation-based predictive multiresolution image coder," *IEEE Trans. on Image Processing*, vol. 4, no. 1, pp. 34–47, January 1995.

[4] G. M. Schuster and A. K. Katsaggelos, *Rate-distortion based video compression.* Boston: Kluwer Academic Publishers, 1997.

[5] H. Derin, H. Elliott, R. Cristi, and D. Geman, "Bayes smoothing algorithms for segmentation of binary images modeled by Markov random fields," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 707–719, November 1984.

[6] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society B*, vol. 48, no. 3, pp. 259–302, 1986.

[7] H. Derin and H. Elliott, "Modeling and segmentation of noisy and textured images using Gibbs random fields," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 1, pp. 39–55, January 1987.

[8] J. Besag, "Efficiency of pseudolikelihood estimation for simple Gaussian fields," *Biometrica*, vol. 64, no. 3, pp. 616–618, 1977.

[9] H. Derin and P. A. Kelly, "Discrete-index Markov-type random processes," *Proc. of the IEEE*, vol. 77, no. 10, pp. 1485–1510, October 1989.

[10] J. Zhang, J. W. Modestino, and D. A. Langan, "Maximum-likelihood parameter estimation for unsupervised stochastic model-based image segmentation," *IEEE Trans. on Image Processing*, vol. 3, no. 4, pp. 404–420, July 1994.

[11] X. Descombes, R. Morris, J. Zerubia, and M. Berthod, "Estimation of Markov random field prior parameters using Markov chain Monte Carlo maximum likelihood," Tech. Rep. 3015, INRIA-Institut National de Recherche en Informatique et en Automatique, October 1996.

[12] S. S. Saquib, C. A. Bouman, and K. Sauer, "ML parameter estimation for Markov random fields with applications to Bayesian tomography," *IEEE Trans. on Image Processing*, vol. 7, no. 7, pp. 1029–1044, July 1998.

[13] P. J. Burt, T. Hong, and A. Rosenfeld, "Segmentation and estimation of image region properties through cooperative hierarchical computation," *IEEE Trans. on Systems Man and Cybernetics*, vol. SMC-11, no. 12, pp. 802–809, December 1981.

[14] I. Ng, J. Kittler, and J. Illingworth, "Supervised segmentation using a multiresolution data representation," *Signal Processing*, vol. 31, pp. 133–163, March 1993.

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

[16] K. Etemad, D. Doermann, and R. Chellappa, "Multiscale segmentation of unstructured document pages using soft decision integration," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 1, pp. 92–96, January 1997.

[17] M. Unser and M. Eden, "Multiresolution feature extraction and selection for texture segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 717–728, July 1989.

[18] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Trans. on Image Processing*, vol. 4, no. 11, pp. 1549–1560, November 1995.

[19] E. Salari and Z. Ling, "Texture segmentation using hierarchical wavelet decomposition," *Pattern Recognition*, vol. 28, no. 12, pp. 1819–1824, December 1995.

[20] B. Gidas, "A renormalization group approach to image processing problems," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 2, pp. 164–180, February 1989.

[21] C. A. Bouman and B. Liu, "Multiple resolution segmentation of textured images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 99–113, February 1991.

[22] P. Perez and F. Heitz, "Multiscale Markov random fields and constrained relaxation in low level image analysis," *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, vol. 3, March 23-26 1992, San Francisco, CA, pp. 61–64.

[23] C. A. Bouman and M. Shapiro, "Multispectral image segmentation using a multiscale image model," *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, vol. 3, March 23-26 1992, San Francisco, California, pp. 565–568.

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

[25] J. M. Laferte, F. Heitz, P. Perez, and E. Fabre, "Hierarchical statistical models for the fusion of multiresolution image date," *Proc. Int'l Conf. on Computer Vision*, June 20-23 1995, Cambridge, MA, pp. 908–913.

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

[27] Z. Kato, M. Berthod, and J. Zerubia, "Parallel image classification using multiscale Markov random fields," *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, vol. 5, April 27-30 1993, Minneapolis, MN, pp. 137–140.

[28] M. L. Comer and E. J. Delp, "Segmentation of textured images using a multiresolution Gaussian autoregressive model," *IEEE Trans. on Image Processing*, to appear.

[29] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees.* Belmont, CA: Wadsworth International Group, 1984.

[30] S. B. Gelfand, C. S. Ravishankar, and E. J. Delp, "An iterative growing and pruning algorithm for classification tree design," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 163–177, February 1984.

[31] H. Cheng, C. A. Bouman, and J. P. Allebach, "Multiscale document segmentation," *Proc. of IS&T's 50th Annual Conf.*, May 18-23 1997, Cambridge, MA, pp. 417–425.

[32] H. Cheng and C. A. Bouman, "Trainable context model for multiscale segmentation," *Proc. of IEEE Int'l Conf. on Image Proc.*, October 4-7 1998, Chicago, IL.

[33] J. M. Shaprio, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, December 1993.

[34] K. Daoudi, A. B. Frakt, and A. S. Willsky, "Multiscale autoregressive models and wavelets," *IEEE Trans. on Information Theory*, to appear.

[35] M. Aitkin and D. B. Rubin, "Estimation and hypothesis testing in finite mixture models," *Journal of the Royal Statistical Society B*, vol. 47, no. 1, pp. 67–75, 1985.

[36] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, no. 1, pp. 1–38, 1977.

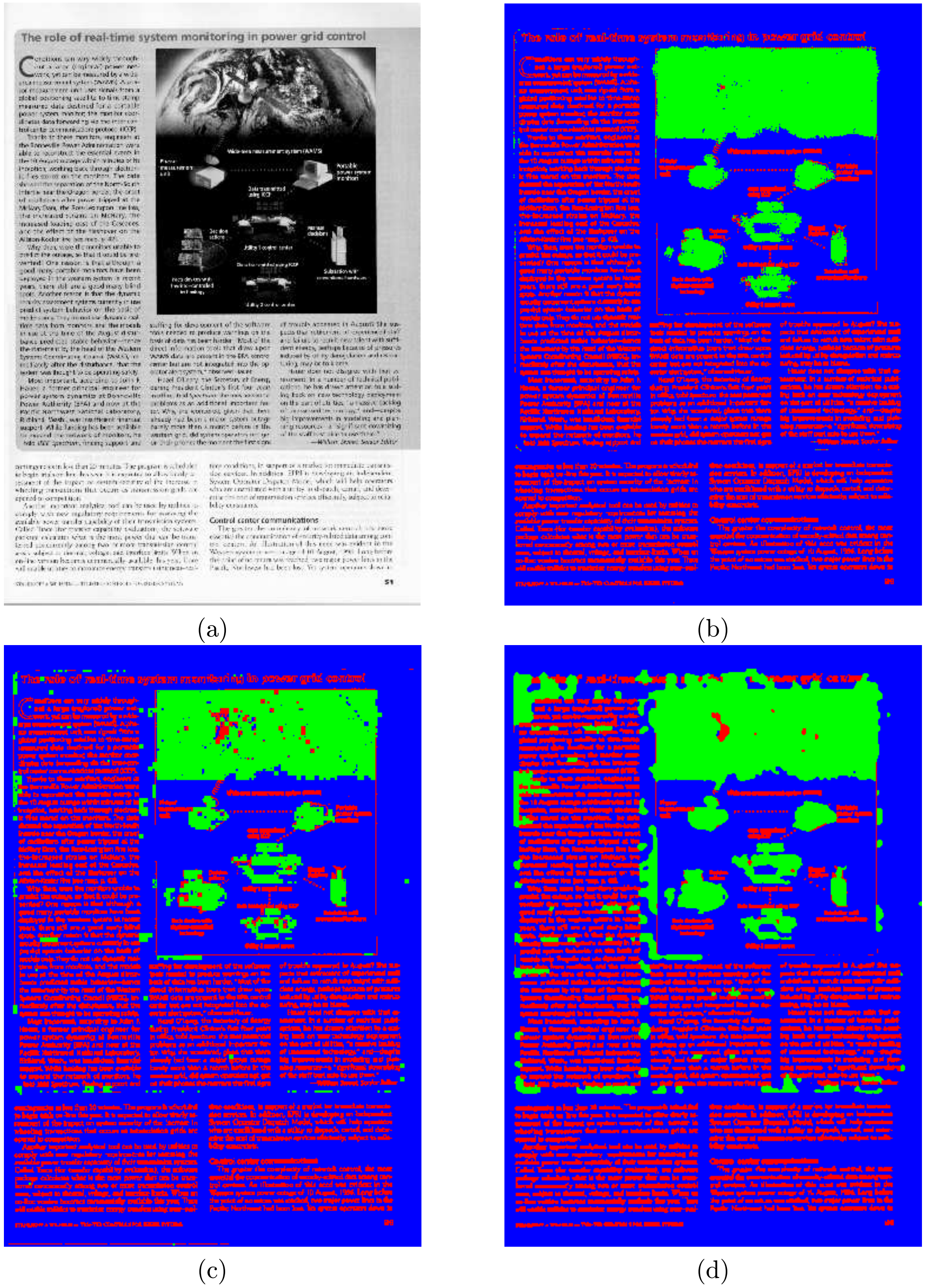
[37] O. Ronen, J. R. Rohlicek, and M. Ostendorf, "Parameter estimation of dependence tree models using the EM algorithm," *IEEE Signal Processing Letters*, vol. 2, no. 8, pp. 157–159, August 1995.

[38] H. Lucke, "Bayesian belief networks as a tool for stochatic parsing," *Speech Communication*, vol. 16, no. 1, pp. 89–118, January 1995.

[39] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 721–741, November 1984.

[40] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *The Annals of Statistics*, vol. 11, no. 2, pp. 417–431, September 1983.

[41] H. Cheng and C. A. Bouman, "Multilayer document compression algorithm," *Proc. of IEEE Int'l Conf. on Image Proc.*, October 24-28 1999, Kobe, Japan.



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

Figure 11: (a) Original image. (b) Segmentation result using TSMAP with a $5 \times 5$ neighborhood. (c) Segmentation result using TSMAP with a $1 \times 1$ neighborhood. (d) Segmentation result using Markov random field. Red, green and blue represent text, picture and background, respectively.

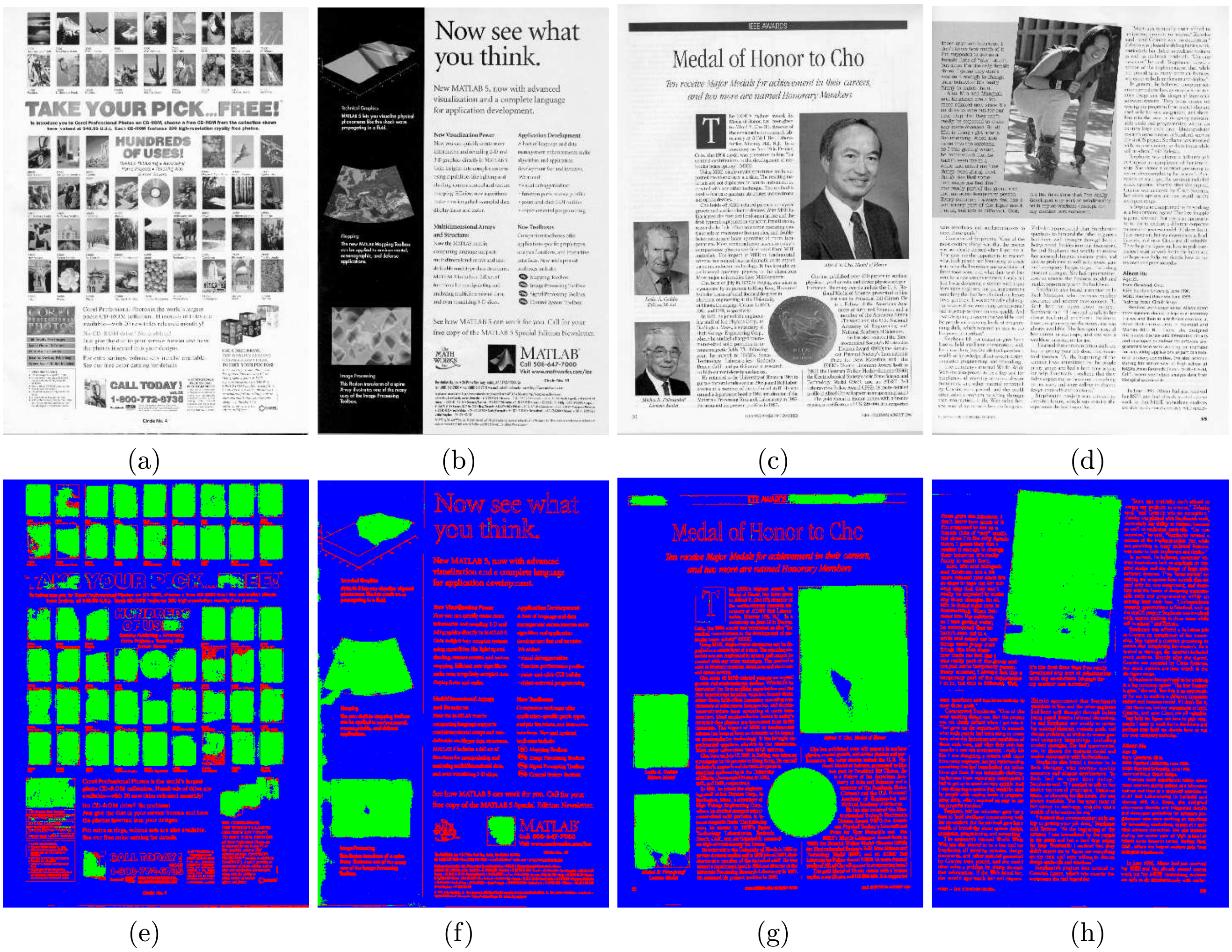


Figure 12: (a)-(d) Original images. (e)-(h) Segmentation results using TSMAP with a $5 \times 5$ neighborhood. Red, green, and blue represent text, picture and background, respectively.

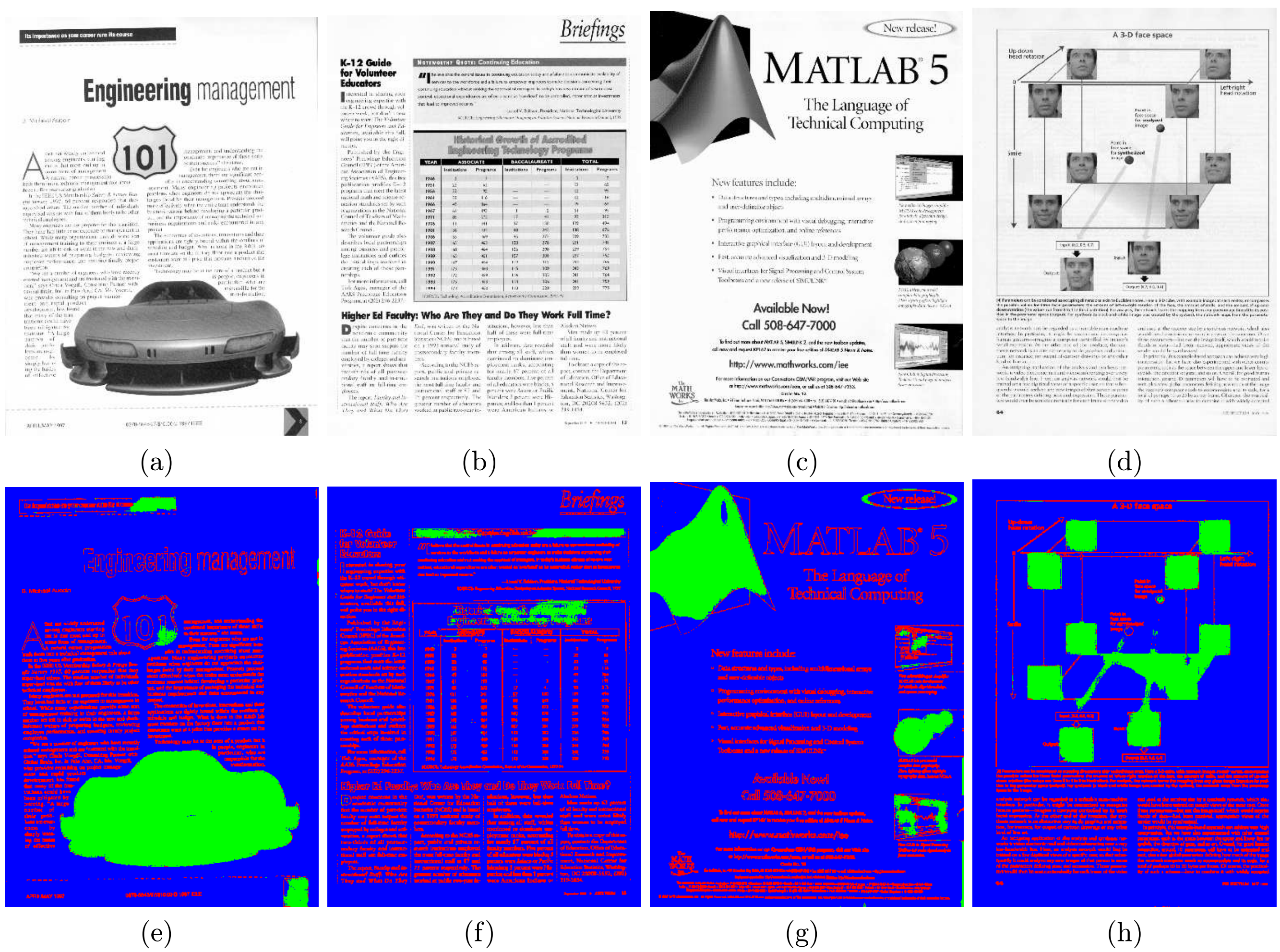



Figure 13: (a)-(d) Original images. (e)-(h) Segmentation results using TSMAP with a $5 \times 5$ neighborhood for 4 different test images. Red, green, blue represent text, picture, and background respectively.

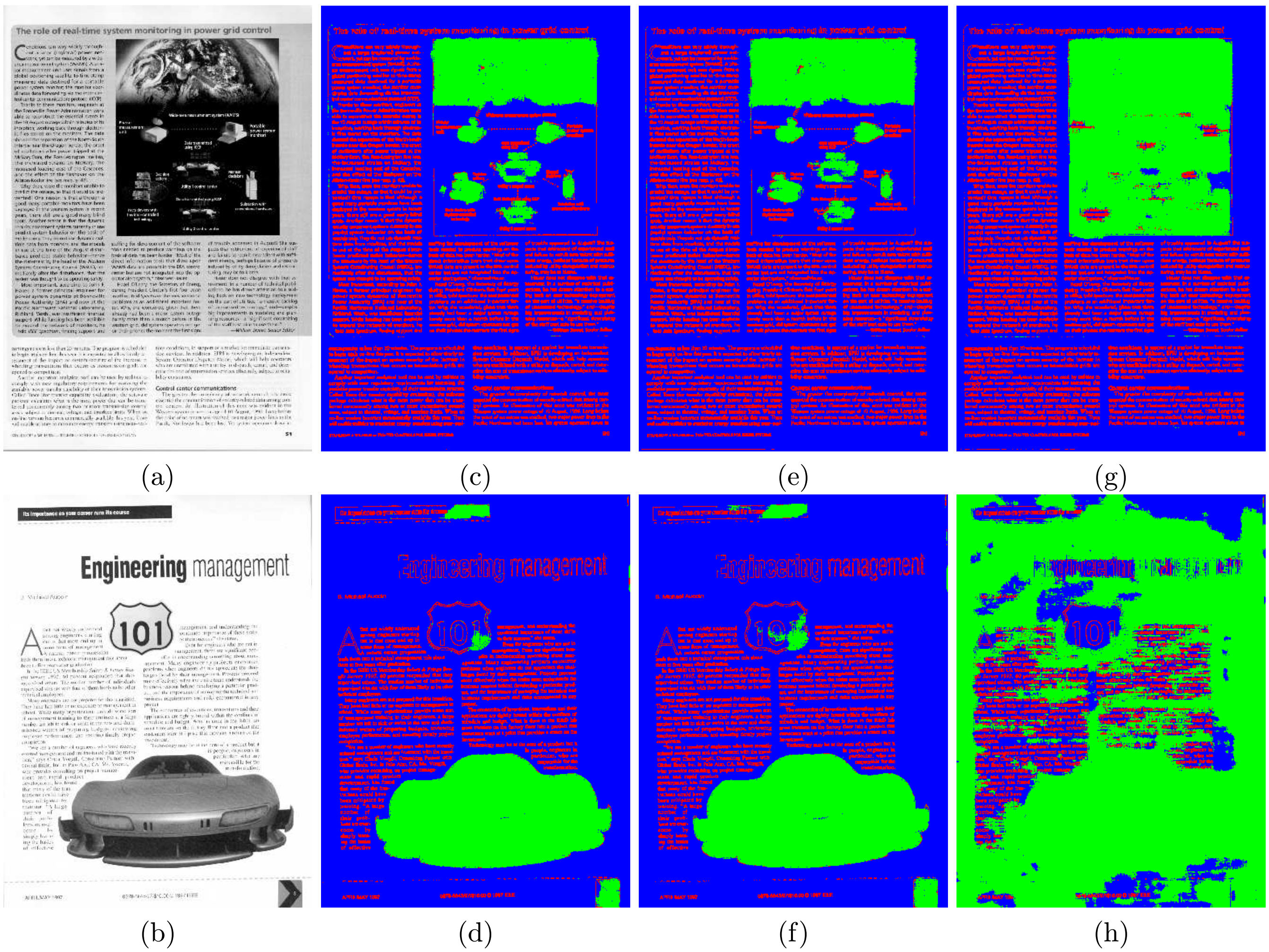



Figure 14: (a)-(b) Original images. (c)-(d) TSMAP segmentation results when trained on 20 images. (e)-(f) TSMAP segmentation results when trained on 10 images. (g)-(h) TSMAP segmentation results when trained on 5 images. For all cases, a $5 \times 5$ coarse neighborhood is used. Red, green and blue represent text, picture and background, respectively.