A Computationally Efficient Approach to the Estimation of Two- and Three-Dimensional Hidden Markov Models

Dhiraj Joshi, *Student Member, IEEE*, Jia Li, *Senior Member, IEEE*, and James Z. Wang, *Senior Member, IEEE*

***Abstract*—Statistical modeling methods are becoming indispensable in today's large-scale image analysis. In this paper, we explore a computationally efficient parameter estimation algorithm for two-dimensional (2-D) and three-dimensional (3-D) hidden Markov models (HMMs) and show applications to satellite image segmentation. The proposed parameter estimation algorithm is compared with the first proposed algorithm for 2-D HMMs based on variable state Viterbi. We also propose a 3-D HMM for volume image modeling and apply it to volume image segmentation using a large number of synthetic images with ground truth. Experiments have demonstrated the computational efficiency of the proposed parameter estimation technique for 2-D HMMs and a potential of 3-D HMM as a stochastic modeling tool for volume images.**

***Index Terms*—Hidden Markov models (HMMs), maximum likelihood estimation, parameter estimation, three-dimensional (3-D) HMM, Viterbi training, volume image processing.**

I. Introduction

OVER THE last decade, we have see an explosion of digital image usages in a number of application domains. Each day, large quantities of multidimensional images are being acquired digitally. Scanners in satellites produce enormous amounts of data for surveillance and reconnaissance purposes. Multispectral and hyper-spectral images contain additional information in the form of a third spectral dimension. Modern telescopes are capable of producing high-resolution pictures of distant stars and galaxies. Magnetic resonance imaging (MRI) and computed tomography (CT) scanners in hospitals produce high-resolution three-dimensional (3-D) images of the human brain or the human body so that physicians and radiologists can look inside their patients noninvasively. At airports, 3-D CT scanners are being installed to monitor luggages checked in by travelers in order to detect dangerous substances. The amount of digital image information being generated is so enormous that it has become inevitable for computers to help analyze, segment, and classify these images. Because multidimensional images are typically stored in image databases, it is now possible for computers to extract patterns or semantic connections based on a large collection of annotated or classified images. Such automatically extracted patterns can also be used by computers to classify new images.

In the recent past, researchers in the image analysis community have successfully used statistical modeling techniques to segment, classify, and annotate images. Particularly, variations of hidden Markov models (HMMs) have been developed and successfully used for image and video applications. The key issue in using such complex models is estimation of parameters which is usually a computationally expensive task. In practice, often a trade off is accepted between accuracy of estimation and running time of the estimation algorithm.

In this paper, we introduce a new model for studying collections of volume images (3-D HMM). We also present a computationally efficient parameter estimation algorithm for two-dimensional (2-D) and 3-D HMMs. As discussed in Section IV-D, the estimation algorithm can be efficiently scaled from its version for 2-D HMMs to be used for 3-D HMMs. The proposed algorithm is polynomial-time in number of states and linear-time in the number of pixels of an image, for both 2-D HMM and 3-D HMMs. When applied to 2-D HMM-based supervised learning, the running time of the algorithm is found to be much less compared to the estimation algorithm for 2-D HMMs which was proposed in [24]. The techniques have been implemented and tested with a large number of images. Promising results have been obtained. The algorithms can potentially be applied to other analysis tasks involving high-dimensional data.

A. Extending Conventional HMMs

Over the years, HMMs have been explored as efficient statistical modeling techniques in speech recognition and image and video understanding [1], [17], [23], [36].

HMMs are a special case of graphical models in statistics [13]. Observed signals are modeled through a first order Markov process as follows. Suppose the observed signal is a sequence $\{u_1, u_2, \ldots, u_T\}$, where u_t could be multivariate. An HMM assumes that each observation in the sequence exists in one of a set of hidden states at each position t. Let the corresponding hidden state sequence be $\{s_1, s_2, \ldots, s_T\}$, $s_t \in \{1, 2, \ldots, M\}$. It is usually assumed that the state sequence follows a first order Markov chain. That is, $P(s_{t+1}|s_t, s_{t-1}, s_{t-2}, \ldots, s_1) =$

Manuscript received January 4, 2005, revised August 25, 2005. The Website http://riemann.ist.psu.edu provides more information related to this work. This work was supported in part by the U.S. National Science Foundation under Grants IIS-0219272, IIS-0347148, and ANI-0202007; in part by The Pennsylvania State University; in part by the PNC Foundation; and in part by SUN Microsystems under Grants EDUD-7824-010456-US and EDUD-7824-030469-US. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Thierry Blu.

D. Joshi is with the Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802 USA (e-mail: djoshi@cse.psu.edu).

J. Li is with the Department of Statistics and the Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802 USA (e-mail: jiali@stat.psu.edu).

J. Z. Wang is with the College of Information Sciences and Technology and the Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802 USA (e-mail: jwang@ist.psu.edu).

Digital Object Identifier 10.1109/TIP.2006.877039

$P(s_{t+1}|s_t)$. Furthermore, the time homogeneity of the Markov chain is normally assumed, i.e., the transition probabilities, $P(s_{t+1} = n|s_t = m) = a_{m,n}$, are invariant over t. Given the state s_t, the observation u_t is conditionally independent of the states and observations at the other positions. The conditional distribution of u_t given s_t is usually modeled parametrically with parameters varying with the conditioned state. The most widely used distribution for multivariate data is the multivariate Gaussian distribution which is parameterized by a mean vector and a covariance matrix.

To effectively account for the 2-D statistical dependence, 1-D HMMs have been extended to pseudo 2-D HMMs and pseudo 3-D HMMs for face recognition [11], [18]. Another line of research is to extend the underlying Markov chain assumed for sequential data in a 1-D HMM to a Markov mesh for spatial data [24] (for 2-D images, the observations $u_{i,j}$ reside on a 2-D grid.).

There is a surge of research to analyze volume images. Conventional 2-D image modeling paradigms may not be effective In the past, 3-D Markov random fields (MRFs) have been applied to medical image segmentation [27]. Given the great potential demonstrated by the HMM paradigm in various applications, it seems most natural to extend 2-D HMM to 3-D HMM for volume image analysis. In this paper, we construct a 3-D HMM that captures the 3-D statistical dependence in volume images.

In 3-D modeling, a volume image is represented by feature vectors on a 3-D grid. An image may be divided into cubes which could be overlapping. In such a case, every cube corresponds to one position in the grid. In our experiments, the size of the cube is 1 and each voxel represents a distinct grid point. The observation $u_{i,j,k}$ at each position comprises features extracted from the respective cube. It is assumed that each position exists in an underlying state $s_{i,j,k}$ and the observation $u_{i,j,k}$ is conditionally independent of the other positions when $s_{i,j,k}$ is known. The Markovian property that governs the transition of states in 2-D HMM is extended to one in 3-D. In particular, the transition probability takes into account the neighboring state along the extra dimension.

For the same number of states, the dependence in the additional dimension leads to a significant increase in the number of transition probabilities to be estimated. To robustly estimate these parameters, a model regularization approach is explored. Another challenge faced in the extension is the computational complexity of the estimation algorithm. The Viterbi training [24] approach is used here. The key computational issue is to search for the combination of states with the maximum *a posteriori* (MAP) probability conditioned on all the observations under a set of given parameters. A computationally efficient algorithm is developed to achieve a locally optimal solution by iteratively seeking the best states in a subset with the other states fixed. Our algorithm appears to extend to the case of Markov mesh random fields on higher dimensional finite rectangular lattices.

B. Related Work in Modeling

One of the key challenges of mathematical and statistical modeling has been to understand and model the neuro-physiological processes that form the basis of human visual cognition. Empowering a machine with such capabilities would have plausible applications in many areas such as robotics, medical diagnostics, and automated surveillance.

There is a rich resource of prior related work on statistical image modeling in computer vision, image processing, and machine learning. Space limitations do not allow us to present a broad survey. Instead we try to emphasize some of the work that is most related to what we propose. The references below are to be taken as examples of related work, not as the complete list of work in the cited areas.

Theories and methodologies related to MRFs [6], [10], [12], [20] have played important roles in the construction of many statistical image models which have widely been used for several applications in computer vision [26]. For a thorough introduction to MRFs and their applications, see Kindermann and Snell [20] and Chellappa and Jain [6]. An MRF extends Markovian dependence from 1-D to a general graph [10], [12].

A substantial amount of work has been done on applying MRFs to image segmentation and texture analysis [5], [7], [9], [12], [15], [29], [31]–[33], [35], [38]–[40]. Estimation and approximation techniques for MRF-based procedures have also been widely studied [3], [14], [16], [19], [34]. An interesting work on stationary MRFs has been presented in [4]. Particularly, MRFs on a finite rectangular lattice have been rigorously characterized.

Many applications use MRFs to model the pixel representation of images, which sometimes may not be the best way to model the dependencies occurring in real images. In some real applications (such as satellite image classification), certain directional patterns are seen to be more pronounced than others. In such cases, HMMs, which do not impose isotropy, are expected to capture dependencies, in a more precise fashion.

Inspired by HMMs, researchers have recently used MRFs to model the classes or states of image pixels rather than their direct values [2], [8], [22]. Pixel values are considered conditionally independent given their classes or states. We discuss here a number of such models. It is worthy to note that although these models constitute an important part of existing statistical image models, they represent, by no means, the complete set. Multiscale representations of MRFs have also been studied [2], [28]. Another multiresolution model based on HMMs is the model proposed for wavelet coefficients by Crouse *et al.* [8], where wavelet coefficients across resolutions are assumed to be generated by one-dimensional HMMs with resolution playing the role of time, in the Markov chain.

C. Outline of the Paper

The remainder of the paper is organized as follows. Section II introduces the 2-D HMMs. Section III describes the construction of the 3-D HMM. In Section IV, the parameter estimation method and the computationally efficient algorithm are introduced. Sections V and VI discuss the experiments and their results. We present our conclusions and suggest future research directions in Section VII. The Appendix includes a proof of an equation used in the development of the computationally efficient algorithm.

II. Two-Dimensional Hidden Markov Models

Two-dimensional HMMs and their multiresolution extension [23] have been studied, in the past and successfully applied to supervised image classification, compression [21] and annotation [25]. In this section, we briefly introduce the assumptions of a 2-D HMM with respect to 2-D image modeling.

Conventionally, 2-D images are divided into square blocks and block-based features are extracted. Each block represents a point in a 2-D grid and the features extracted from block at (i, j) are represented by $u_{i,j}$. In the limiting case when block size becomes 1, each pixel represents a point in the 2-D grid. The observation $u_{i,j}$ is usually a vector containing features computed from the pixel values in the block at (i, j). A lexicographic ordering of points (i, j) is defined as follows: $(i', j') < (i, j)$ if $i' < i$ or $i' = i$, $j' < j$.

Under the assumptions of the model, the underlying states $s_{i,j}$ are governed by a first order Markov mesh: $P(s_{i,j}|s_{i',j'} : (i', j') < (i, j)) = a_{m,n,l}$, where $m = s_{i-1,j}$, $n = s_{i,j-1}$ and $l = s_{i,j}$. That is, among the conditioned states, only states of the neighbor above and the neighbor to the left of the current position affect the transition probability. For points at boundaries, which miss one of the two needed neighbors, dependence upon only the states of the existing neighbor is assumed. Besides this, given the state $s_{i,j}$ of a point (i, j), the feature vector $u_{i,j}$ is assumed to follow a multivariate Gaussian distribution with a covariance matrix and a mean vector determined by its state.

For the 2-D HMM, we need to estimate the transition probabilities $a_{m,n,l}$, $m, n, l = 1, \ldots, M$, the mean vectors μ_l, and the covariance matrices Σ_l for each state $l \in \{1, 2, \ldots, M\}$.

III. Three-Dimensional Hidden Markov Models

In this section, we present the 3-D HMM in its general form. Before stating the statistical assumptions, we describe the following terminologies which will be used throughout the paper.

- *3-D point*: A point (i, j, k) where i, j and k are coordinates along the X, Y and Z axes respectively. Typically, an observation is associated with a 3-D point. For a 3-D image, every voxel may correspond to a point. In certain applications, a 3-D image may be divided into rectangular solids and features are extracted for each solid. In such cases, voxels in a rectangular solid are treated collectively as one point.
- *3-D grid*: A 3-D array of finite and equally spaced 3-D points (along X, Y and Z axes) in space represented as $\mathcal{C} = \{(i, j, k), 0 \le i \le w-1, 0 \le j \le w-1, 0 \le k \le w-1\}$. Here w is a finite positive integer. For brevity, we will assume a cubic 3-D grid. A graphical illustration for the 3-D grid is in Fig. 1.
- *frame*: We call a collection of points on a plane parallel to that of X and Y axes a frame. A frame is indexed by its Z coordinate: $\{(\cdot, \cdot, k)\}$.

A lexicographic order is defined among 3-D points as follows: $(i', j', k') < (i, j, k)$ if $k' < k$ or $k' = k$, $j' < j$ or $k' = k$, $j' = j$, $i' < i$. The observed feature vector of 3-D point (i, j, k) will be denoted by $u_{i,j,k}$ and its state by $s_{i,j,k}$. The number of states M is preselected.

The model attempts to capture statistical dependence among 3-D points. The assumptions of 3-D HMM are as follows.

1) $P\{s_{i,j,k} = l \mid context\} = a_{p,m,n,l}$, where $context = \{(s_{i'j'k'}, u_{i'j'k'})\colon (i', j', k') < (i, j, k)\}$ is the set of states and feature vectors of all points preceding (i, j, k) in the lexicographic order. In addition, $p = s_{i,j,k-1}$ is the state of the point at the same X and Y coordinates in the previous frame, $m = s_{i-1,j,k}$ and $n = s_{i,j-1,k}$ are the states of the neighboring points in the same frame and one position ahead along X and Y respectively. Given any point (i, j, k), the three neighboring points that affect it among all those that precede it are denoted as N_1, N_2, and N_3 in Fig. 1. For boundary points, dependence upon only the state of the existing neighbors is assumed. For example, if for a boundary point, the neighbor denoted as N_3 is missing, then dependence upon neighbors N_1 and N_2 is assumed.
2) Given the state $s_{i,j,k}$ of a point (i, j, k), the feature vector $u_{i,j,k}$ follows a multivariate Gaussian distribution parametrized by a covariance matrix and a mean vector determined by the state. For a state $l \in \{1, 2, \ldots, M\}$, denote the corresponding covariance matrix and mean vector by Σ_l and μ_l. Recall that the probability density function of a d-dimensional Gaussian distribution is
$$b_l(u) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_l|}} e^{-(1/2)(u-\mu_l)' \Sigma_l^{-1} (u-\mu_l)}.$$
3) If the state of point (i, j, k) is known, its observed vector $u_{i,j,k}$ is conditionally independent of the rest of the points in the 3-D grid.

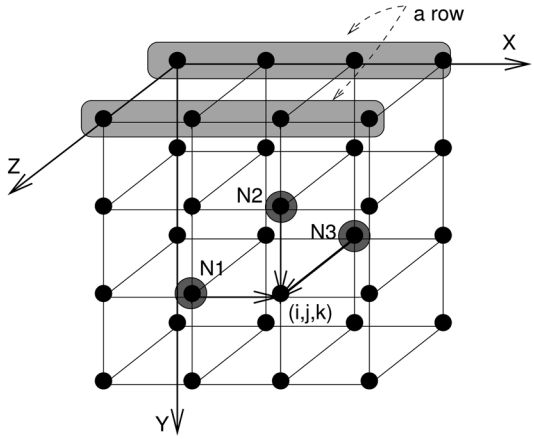

Fig. 1. Three-dimensional grid. Given the states of all the points that precede point (i, j, k), only the states of the three neighboring points affect the distribution of the state at (i, j, k). These are denoted as N_1, N_2, and N_3 in the figure.

For the 3-D HMM, the transition probabilities $a_{p,m,n,l}$, p, m, n, $l = 1, \ldots, M$, the mean vectors μ_l, and the covariance matrices Σ_l for each state $l \in \{1, 2, \ldots, M\}$ need to be estimated. Subject to the constraint $\sum_{l=1}^{M} a_{p,m,n,l} = 1$ for any $0 < p < w$, m, and n, the transition probabilities comprise $M^3(M-1)$ free parameters. By the assumptions of our 3-D HMM model, given a previous frame, the states in the current frame follow a 2-D HMM.

IV. Parameter Estimation

In this section, we present an algorithm designed to estimate parameters of two and 3-D HMMs. We present an efficient way to extend the Viterbi algorithm to 3-D. Our approach may not be unique or best but is computationally efficient as discussed later. For clarity, we present our version of the algorithm for 3-D HMMs. The version of the estimation algorithm for 2-D HMMs

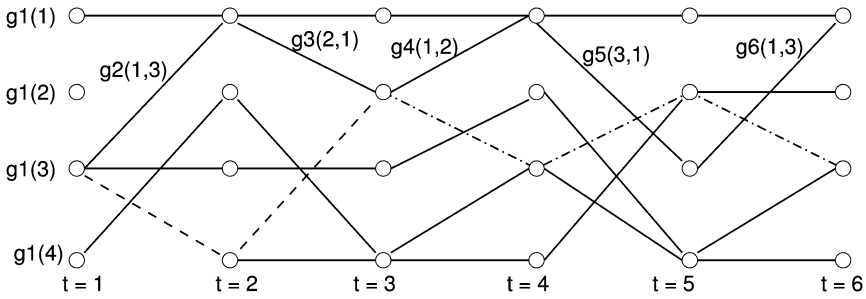

Fig. 2. Schematic of the Viterbi algorithm. Assume that the dashed path is the shortest path from $t = 1$ to $t = 6$.

follows naturally from its 3-D HMM counterpart, since a 2-D image can be thought of as a special case of a 3-D image with only one frame.

A. Viterbi Training

The Viterbi training approach [37] used to estimate 1-D HMM is adopted here. In particular, we iterate the search for the optimal combination of states and the update of parameters. From optimization perspective, the Viterbi training algorithm provides a local maximization of the so-called classification likelihood [30], i.e., the likelihood of the observed feature vectors under a given set of parameters as well as a given set of underlying states. The unknown states are regarded as parameters and are part of the estimation.

Consider the problem of predicting the optimal state sequence $\bar{s} = s_1, s_2, \ldots, s_w$ for a 1-D HMM. In many cases, as in our case, the classification likelihood, to be maximized, is separable and can be expressed in the form

$$G(\bar{s}) = g_1(s_1) + \sum_{u=2}^{w} g_u(s_u, s_{u-1}).$$

Suppose, in the optimal state sequence s^*, the state of the tth position is $s_t^* = k$, it is easy to see that maximization of $G(s_1, s_2, \ldots, s_w)$ is equivalent to maximization of two separate functions

$$G_{t,k}(s_1, \ldots, s_{t-1}) = g_1(s_1) + g_2(s_2, s_1) + \cdots + g_t(k, s_{t-1})$$

and

$$\bar{G}_{t,k}(s_{t+1}, \ldots, s_w) = g_{t+1}(s_{t+1}, k) + \cdots + g_w(s_w, s_{w-1}).$$

The expressions $G_{t,k}(\cdot)$ and $\bar{G}_{t,k}(\cdot)$ represent terms in $G(\bar{s})$, for $u < t$ and $u > t$ respectively, when the state at position t is k. Also note that

$$G_{t,l}(s_1, \ldots, s_{t-2}, k) = G_{t-1,k}(s_1, \ldots, s_{t-2}) + g_t(l, k).$$

The problem of finding the optimal state sequence can be seen as a shortest path problem as shown in Fig. 2. Here, a weight $g_t(k, l)$ is placed on a link from state l at $t - 1$ to state k at t. The starting nodes have a weight $g_1(k)$ for state k. Suppose, the path denoted by a dashed line is the optimal one, then it is easy to see that the dashed sub-path, ending at $t = 3$, is optimal among the sub-paths ending at $t = 3$ in state 2. Similarly, the dashed sub-path between $t = 3$ to $t = 6$ is shortest among the sub-paths starting at $t = 3$ in state 2. Viterbi algorithm exploits such an optimal sub-structure in an optimization problem. The sequential steps of the Viterbi algorithm are as follows.

1) At $t = 1$, record $G_{1,k}^* = g_1(k), \forall\, k \in \{1, \ldots, M\}$ where M represents the number of states.
2) For $t = 2 \ldots w$, $\forall\ k \in \{1, \ldots, M\}$, set $G_{t,k}^* = \max_{l=1,\ldots,M} \left\{ G_{t-1,l}^* + g_t(k, l) \right\}$.
3) Finally, M paths are formed each ending at a different state at $t = w$. Pick k^* which maximizes $G_{w,k}^*$ and trace the path backward from the last state k^*.

Note that this procedure requires $(w - 1)M(M - 1)$ comparisons and $(w - 1)M$ memory records. Hence, the computational and storage complexities are $\mathcal{O}(wM^2)$ and $\mathcal{O}(wM)$, respectively.

Denote the collection of parameters of 3-D HMM collectively as ψ. Mark the iteration step by a superscript. Suppose the initial parameter is $\psi^{(0)}$. In Viterbi training, to update $\psi^{(t+1)}$ from $\psi^{(t)}$, the following two steps are applied.

1) Under parameter $\psi^{(t)}$, the combination of states with the MAP probability conditioned on the observed vectors is identified, as shown in the equation at the bottom of the page. Details of this procedure are explained later.
2) The parameters are then computed by assuming states $s_{i,j,k}^*$ are the true underlying states. If the true states were known, the maximum likelihood estimation of the parameters would be easy to obtain. The mean vector μ_l and the covariance matrix Σ_l would simply be estimated by the sample mean and sample covariance matrix of all the observed vectors $u_{i,j,k}$ whose states $s_{i,j,k}^* = l$, shown in (1)–(4) at the bottom of the next page. The transition probabilities $a_{p,m,n,l}$ are computed by the empirical frequencies as shown in (3). where $I(\cdot)$ is the indicator function that equals 1 when the argument is true and 0

$$\begin{aligned}\left\{ s_{i,j,k}^* : (i,j,k) \in \mathcal{C} \right\} &= \underset{s_{i,j,k}:(i,j,k)\in\mathcal{C}}{\arg\max}\ P\left(s_{i,j,k} : (i,j,k) \in \mathcal{C} \mid u_{i,j,k} : (i,j,k) \in \mathcal{C}; \psi^{(t)} \right) \\ &= \underset{s_{i,j,k}:(i,j,k)\in\mathcal{C}}{\arg\max}\ P\left(s_{i,j,k}, u_{i,j,k} : (i,j,k) \in \mathcal{C}; \psi^{(t)} \right)\end{aligned}$$

otherwise. The set $\mathcal{C}^- = \{(i,j,k) : 1 \le i \le w-1, 1 \le j \le w-1, 1 \le k \le w-1\}$.

Various methods can be used to initialize the Viterbi training algorithm. In our experiment, K-means clustering is used to generate an initial set of $s^*_{i,j,k}$. The initialization of K-means is random. The iteration then starts with step 2) above. The final result could be affected by the initial states. In K-means clustering of points, directional dependencies, in the 3-D lattice are ignored. In such a scenario, the generative model is a Gaussian mixture model, a special case of a HMM.

Due to the large number of parameters even with a moderate M, we regularize the transition probabilities by a partial 3-D dependence. In particular, if the dependence along the Z axis is ignored, the model is reduced to a 2-D HMM and the transition probabilities are $\bar{a}_{m,n,l}$, where $m = s_{i-1,j,k}$ and $n = s_{i,j-1,k}$. The maximum likelihood estimation for $\bar{a}_{m,n,l}$ updated at iteration $t+1$ is given in (4).

The 3-D transition probabilities are regularized toward the 2-D ones by a linear combination shown in (5). The value $\alpha \in [0,1]$ controls the extent of 3-D dependence. $\alpha = 1$ corresponds to a pure 3-D model and $\alpha = 0$, a 2-D model. It is shown in the experiments that an intermediate value of α is often preferable. For notation simplicity, we denote the transition probabilities in general by $a_{p,m,n,l}$ even though (5) is used to compute them

$$\tilde{a}^{(t+1)}_{p,m,n,l} = \alpha a^{(t+1)}_{p,m,n,l} + (1-\alpha)\bar{a}^{(t+1)}_{m,n,l}. \tag{5}$$

B. Searching for Optimal States

The computational complexity of 1-D Viterbi algorithm has been discussed in Section IV-A. The key computational issue in Viterbi training is to find the states $\left\{s^*_{i,j,k} : (i,j,k) \in \mathcal{C}\right\}$, with MAP probability, under a given set of parameters. For 1-D HMM, the MAP sequence of states can be solved for by the Viterbi algorithm. For 3-D HMM, there are M^{w^3} possible combinations of states for the entire 3-D grid. When applied directly to the frame process, the Viterbi algorithm enables us to avoid exhaustive search along the Z axis. However, it is still necessary to consider all the possible combinations of states in every frame. In this method, the computational complexity of searching for the optimal set of states is at least $\mathcal{O}(wM^{2w^2})$.

To address the computational difficulty, we propose a locally optimal algorithm to search for the set of states with the MAP probability. Define the set of points with a fixed Y and Z coordinate in a 3-D grid as a row, shown in Fig. 1. Denote a row by $R_{j,k} = \{(i,j,k) : 0 \le i \le (w-1)\}$. Let $\mathcal{D} = \{(j,k) : 0 \le j \le w-1, 0 \le k \le w-1\}$. Denote a sequence of states in row $R_{j,k}$ by $\mathbf{s}_{j,k}$, $\mathbf{s}_{j,k} = \{s_{0,j,k}, s_{1,j,k}, \ldots, s_{w-1,j,k}\}$. For brevity, denote the sequence of observed vectors in row $R_{j,k}$ by $\mathbf{u}_{j,k}$, $\mathbf{u}_{j,k} = \{u_{0,j,k}, u_{1,j,k}, \ldots, u_{w-1,j,k}\}$. We also introduce a lexicographic order for the rows: $(j',k') < (j,k)$ if $k' < k$ or $k' = k$, $j' < j$. We process the rows in the lexicographic order. The procedure to iteratively update the states in each row is as follows, which is also illustrated by Fig. 3. Let the states in a row obtained at iteration t be $\mathbf{s}^{(t)}_{j,k}$.

1) Initialize all the states $\mathbf{s}^{(0)}_{j,k}$, $j,k = 0,1,\ldots,w-1$, in the 3-D grid.
2) Let $t = 0$.
3) Let $k = 0$, $j = 0$.
4) Search for $\mathbf{s}^*_{j,k}$ with the MAP probability given the observed vectors in the entire 3-D grid and the states in all the other rows: $\mathbf{s}^{(t+1)}_{j',k'}$, $(j',k') < (j,k)$ and $\mathbf{s}^{(t)}_{j'',k''}$, $(j,k) < (j'',k'')$. Note that the conditioned states of points in the other rows should always be their most updated values. Set $\mathbf{s}^{(t+1)}_{j,k} = \mathbf{s}^*_{j,k}$.
5) Let $j+1 \to j$.
6) If $j < w$, go back to step 4). Otherwise
   a) set $j = 0$, $k+1 \to k$;
   b) if $k < w$, go back to step 4). Otherwise
      i) let $t+1 \to t$;
      ii) if *stopping criteria* is achieved, stop. Otherwise, go to step 3).

$$\mu^{(t+1)}_{\ell} = \frac{\sum_{(i,j,k)\epsilon\mathcal{C}} u_{i,j,k} I\left(s^*_{i,j,k} = l\right)}{\sum_{(i,j,k)\epsilon\mathcal{C}} I\left(s^*_{i,j,k} = l\right)} \tag{1}$$

$$\sum^{(t+1)}_{l} = \frac{\sum_{(i,j,k)\epsilon\mathcal{C}} \left(u_{i,j,k} - \mu^{(t+1)}_{l}\right)\left(u_{i,j,k} - \mu^{(t+1)}_{l}\right)' I\left(s^*_{i,j,k} = l\right)}{\sum_{(i,j,k)\epsilon\mathcal{C}} I\left(s^*_{i,j,k} = l\right)} \tag{2}$$

$$a^{(t+1)}_{p,m,n,l} = \frac{\sum_{(i,j,k)\epsilon\mathcal{C}} I\left(s^*_{i,j,k-1} = p\right) I\left(s^*_{i-1,j,k} = m\right) I\left(s^*_{i,j-1,k} = n\right) I\left(s^*_{i,j,k} = l\right)}{\sum_{(i,j,k)\epsilon\mathcal{C}} I\left(s^*_{i,j,k-1} = p\right) I\left(s^*_{i-1,j,k} = m\right) I\left(s^*_{i,j-1,k} = n\right)} \tag{3}$$

$$\bar{a}^{(t+1)}_{m,n,l} = \frac{\sum_{(i,j,k)\epsilon\mathcal{C}} I\left(s^*_{i-1,j,k} = m\right) I\left(s^*_{i,j-1,k} = n\right) I\left(s^*_{i,j,k} = l\right)}{\sum_{(i,j,k)\epsilon\mathcal{C}} I\left(s^*_{i-1,j,k} = m\right) I\left(s^*_{i,j-1,k} = n\right)} \tag{4}$$

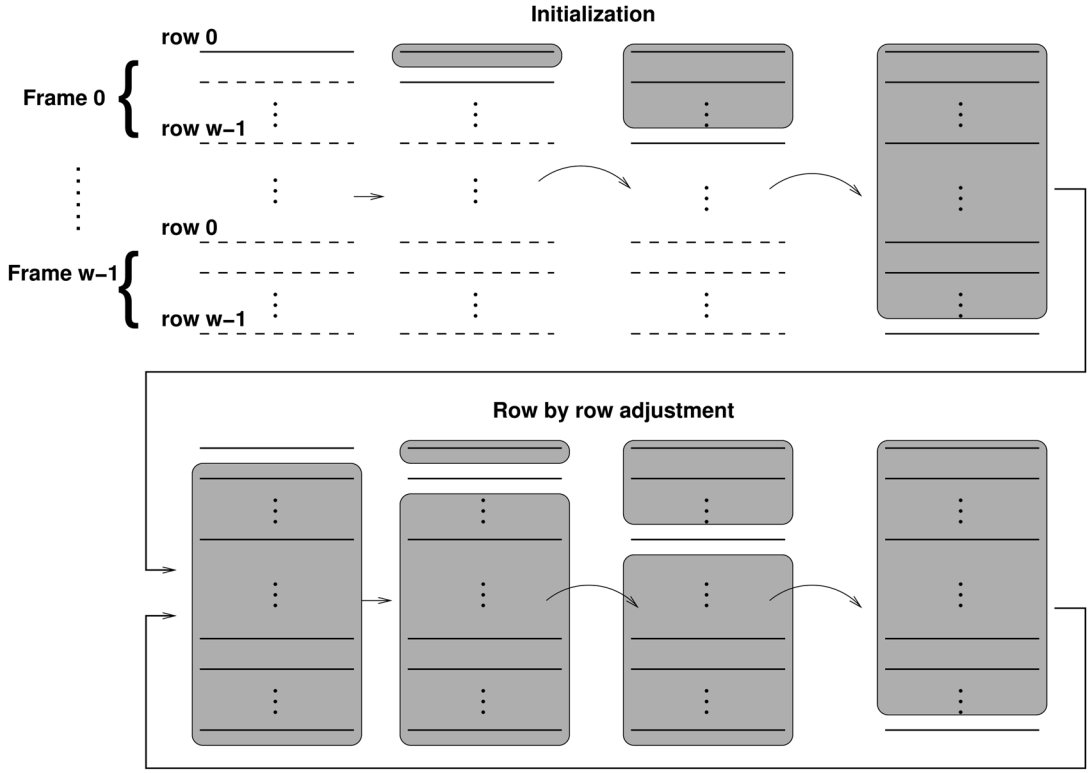

Fig. 3. Process of updating the sequence of states in each row recursively. States and observed vectors in the shaded rows are included in the condition for solving for the sequence of states in a current row that has the MAP probability. After initialization, the scan through all the rows can be repeated in several passes.

In our experiments, we stop the iterations when the number of points whose states are changed by the present iteration, is below a preselected threshold.

The objective function is guaranteed to decrease with each iteration. It is a possibility that cycles could develop after certain iterations. However, in such a case, the objective function remains constant while states keep changing. In all our experiments, we noticed that states as well as the objective function converged after a finite number of iterations. The preselected threshold was very small which guaranteed that the algorithm had achieved near convergence. The convergence could be to an optima different from the global optima. The initial states of the 3-D grid are obtained by a greedy algorithm. In particular, we scan the rows in the lexicographic order. For each row, the sequence of states with the MAP probability conditioned on the states in all the preceding rows and the observed vectors in the current row is selected. The algorithm is greedy because states in each row are determined without looking forward on succeeding rows. The difference between the search for states in each row in initialization and the search in step 4) above lies in the conditioned information. In initialization, only "past" information (in the lexicographic order) is conditioned on. During the adjustment of each row, information about all the other rows is conditioned on. This difference is illustrated in Fig. 3. We now outline steps in initialization.

1) Let $k = 0$, $j = 0$.
2) Search for $\mathbf{s}^*_{j,k}$ with the MAP probability given the states in preceding rows: $\mathbf{s}^*_{j',k'}$, $(j',k') < (j,k)$, and the observed vectors in the current row and the preceding rows: $\mathbf{u}_{j,k}$, $\mathbf{u}_{j',k'}$, $(j',k') < (j,k)$.
3) Let $j+1 \rightarrow j$.
4) If $j < w$, go back to step 2). Otherwise
   a) set $j = 0$, $k+1 \rightarrow k$;
   b) if $k < w$, go back to step 2). Otherwise, stop.

C. Local Optimization of States in a Row

Next, we present the Viterbi algorithm applied to search for the sequence of states $\mathbf{s}^*_{j,k}$ with the MAP probability given all the observed vectors and the states in all the other rows, i.e.,

$$\mathbf{s}^*_{j,k} = \arg\max_{\mathbf{s}_{j,k}} P\left(\mathbf{s}_{j,k} \mid \mathbf{u}_{j,k}, \mathbf{u}_{j',k'}, \bar{\mathbf{s}}_{j',k'} : (j',k') \neq (j,k), (j',k') \in \mathcal{D}\right).$$

Note that the conditioned states (assumed given) are denoted by $\bar{s}_{.,.,.}$ in order to distinguish from the states $s_{i,j,k}$ to be optimized. This state sequence $\mathbf{s}^*_{j,k}$ is needed in the row by row adjustment of states after initialization in the proposed locally optimal algorithm. During initialization of states, as mentioned before, $\mathbf{s}^*_{j,k}$ is defined as the MAP sequence conditioned only on states in the preceding rows. We describe the Viterbi algorithm in the context of row by row adjustment. Its usage in the initialization case is similar. We, thus, omit detailed discussion.

For brevity, we assume the (j,k)th row is not at the boundary. Modification for boundary rows is straightforward. It is proven in the appendix that $\mathbf{s}^*_{j,k}$ is given by the (6) and (7), shown at the bottom of the next page. The first term in the summand is the log likelihood of the observed vector $u_{i,j,k}$ given the state $s_{i,j,k}$. The probabilities in the log function in the other three terms are transition probabilities of entering a state at point (i,j,k), $(i,j+1,k)$, and $(i,j,k+1)$ respectively. Only states in the four

neighboring rows of $R_{j,k}$, specifically, $\bar{\mathbf{s}}_{j-1,k}$, $\bar{\mathbf{s}}_{j+1,k}$, $\bar{\mathbf{s}}_{j,k-1}$, and $\bar{\mathbf{s}}_{j,k+1}$, are relevant to the search of $\mathbf{s}^*_{j,k}$. For a point at the beginning of a row, it has no preceding point along the X axis. Correspondingly, in the above conditional probabilities, the conditioned state of the preceding neighbor along the X axis is dropped. For brevity, we use a unified expression here.

The optimal state sequence $\mathbf{s}^*_{j,k}$ can be solved by the following recursion in the Viterbi algorithm.

1) Let $\theta_l(-1) = 0$ for $l = 1, 2, \ldots, M$.
2) For $i = 0, 1, \ldots, w-1$, compute the following.
   a) $\theta_l(i) = \max_{m=1,\ldots,M} \theta_m(i-1) + \gamma_{m,l}(i)$, where

$$\begin{aligned}\gamma_{m,l}(i) = \log b_l(u_{i,j,k}) &+ \log a_{\bar{\mathbf{s}}_{i,j,k-1},m,\bar{\mathbf{s}}_{i,j-1,k},l} \\ &+ \log a_{\bar{\mathbf{s}}_{i,j+1,k-1},\bar{\mathbf{s}}_{i-1,j+1,k},l,\bar{\mathbf{s}}_{i,j+1,k}} \\ &+ \log a_{l,\bar{\mathbf{s}}_{i-1,j,k+1},\bar{\mathbf{s}}_{i,j-1,k+1},\bar{\mathbf{s}}_{i,j,k+1}}.\end{aligned}$$

   b) Let $\eta_l(i)$ be the number from $\{1, \ldots, M\}$ that maximizes the above function: $\eta_l(i) = \arg\max_{m=1,\ldots,M} \theta_m(i-1) + \gamma_{m,l}(i)$. In case more than one candidates exist for $\eta_l(i)$, which maximize the function, we choose one of them randomly, in the present implementation.
3) Let $s^*_{w-1,j,k} = \arg\max_{l=1,\ldots,M} \theta_l(w-1)$.
4) For $i = w-2, w-3, \ldots, 0$, let $s^*_{i,j,k} = \eta_{s^*_{i+1,j,k}}(i+1)$.

D. Computational Complexity

If the number of states of the model is M, $\mathcal{O}(M^2)$ computation and $\mathcal{O}(M)$ memory is required to find and store state combinations which maximize the likelihood term involving the current point and its previous one. In order to search for the MAP sequence of states in the entire row of w points, the computation required is of the order of wM^2. For a 2-D grid containing $w \times w$ points (or rather w rows), one iteration of the algorithm through the grid will take $\mathcal{O}(w^2M^2)$ time. A 3-D grid containing $w \times w \times w$ points has a total of w^2 rows. Therefore, the complexity of one iteration of the proposed algorithm through the 3-D grid is $\mathcal{O}(w^3M^2)$. Thus, we see that the computational complexity of the proposed algorithm is polynomial in number of states and linear in problem size for both 2-D HMM and 3-D HMM. The problem size, here, represents the number of pixels in an image (for 2-D HMM, the problem size is w^2 and for 3-D HMM, the problem size is w^3).

It was shown in [24] that the likelihood function for a 2-D image represented as a $w \times w$ grid could be expressed as a product of separable functions of state sequences in diagonals. If $\mathcal{N}$ represents all the tuples (i, j) in the grid, then the likelihood function can be factorized as follows:

$$\begin{aligned}&P\{s_{i,j}, u_{i,j} : (i,j) \in \mathcal{N}\} \\ &\quad = P(T_0)P(T_1|T_0)P(T_2|T_1)\ldots P(T_{2w-2}|T_{2w-3}) \\ &\qquad \times \prod_{(i,j)\in\mathcal{N}} P(u_{i,j}|s_{i,j}). \end{aligned} \tag{8}$$

In the previous expression, T_j denotes the state sequence on diagonal j. A path-constrained variable state Viterbi approach was proposed, which restricted the search for state sequences at each diagonal to the N sequences with largest posterior probabilities. Each term $P(T_j|T_{j-1})$ in (8), can be represented as a product of likelihood terms corresponding to individual points in diagonal j, owing to their conditional independence. If there are ν points on diagonal j, the unconstrained Viterbi approach would normally involve searching all M^ν state sequences for the N sequences which yield highest probabilities. However, because of the conditional independence of points in diagonal j, this search can be restricted to searching sequences composed of states, which give N highest probabilities at each point. For elaboration, suppose diagonal j contains ν points (numbered $1, \ldots, \nu$) and the log likelihood of point i being in state k is denoted as $\lambda_{i,k}$. The selection of N nodes is equivalent to finding N state sequences, $\{s_i : i = 1, \ldots, \nu\}$, with the largest $\sum_{i=1}^{\nu} \lambda_{i,s_i}$. As a consequence of conditional independence of points on the diagonal the following equality holds:

$$\begin{aligned}&\arg\max_{s_i:i=1,\ldots,\nu} \left\{\sum_{i=1}^{\nu} \lambda_{i,s_i}\right\} \\ &\qquad = \left\{s^*_i : i = 1, \ldots, \nu, s^*_i = \arg\max_{s_i} \lambda_{i,s_i}\right\}. \end{aligned} \tag{9}$$

Searching for $\arg\max_{s_i} \lambda_{i,s_i}$ is an $\mathcal{O}(M)$ operation (where M is the number of states of the model) and so finding the maximum sequence is an $\mathcal{O}(M\nu)$ operation. This approach can be easily extended to finding N sequences with largest posterior probabilities. For more details, readers are referred to [24].

Under the path-constrained variable state Viterbi approach, an image is broken down into square subimages of size $k \times k$ and each subimage is modeled as a separate 2-D HMM. Thus, for a 2-D image represented as a $w \times w$ grid, w^2/k^2 separate 2-D

$$\begin{aligned}\mathbf{s}^*_{j,k} = \arg\max_{\mathbf{s}_{j,k}} \sum_{i=0}^{w-1} \big(&\log P(u_{i,j,k} \mid s_{i,j,k}) + \log P(s_{i,j,k} \mid \bar{s}_{i,j,k-1}, s_{i-1,j,k}, \bar{s}_{i,j-1,k}) \\ &+ \log P(\bar{s}_{i,j+1,k} \mid \bar{s}_{i,j+1,k-1}, \bar{s}_{i-1,j+1,k}, s_{i,j,k}) \\ &+ \log P(\bar{s}_{i,j,k+1} \mid s_{i,j,k}, \bar{s}_{i-1,j,k+1}, \bar{s}_{i,j-1,k+1})\big) \end{aligned} \tag{6}$$

$$\begin{aligned}= \arg\max_{\mathbf{s}_{j,k}} \sum_{i=0}^{w-1} \big(&\log b_{s_{i,j,k}}(u_{i,j,k}) + \log a_{\bar{\mathbf{s}}_{i,j,k-1},s_{i-1,j,k},\bar{\mathbf{s}}_{i,j-1,k},s_{i,j,k}} \\ &+ \log a_{\bar{s}_{i,j+1,k-1},\bar{s}_{i-1,j+1,k},s_{i,j,k},\bar{s}_{i,j+1,k}} \\ &+ \log a_{s_{i,j,k},\bar{s}_{i-1,j,k+1},\bar{s}_{i,j-1,k+1},\bar{s}_{i,j,k+1}}\big) \end{aligned} \tag{7}$$

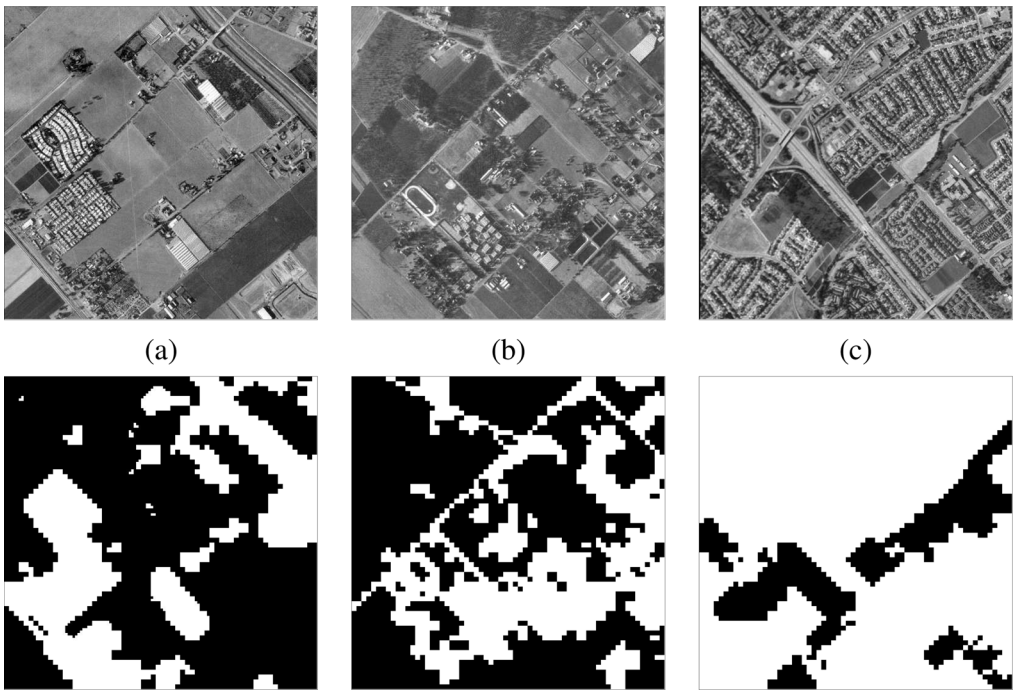

Fig. 4. (a)–(c) Three aerial images and their manual classification. Left: Original 8-bpp images, Right: Hand-labeled classified images. White represents man-made and gray represents natural.

HMMs are estimated. The value of the subimage size k is generally selected as 8. Path constrained variable-state Viterbi algorithm, for each $k \times k$ subimage, can be performed in $\mathcal{O}(kN^2)$ time and so the computational complexity for the entire image is roughly $\mathcal{O}(w^2N^2/k)$. It would be an interesting future work to extend this approach to be used for 3-D HMMs and compare with the proposed algorithm. For the remainder of the paper, we will refer to the estimation algorithm based on path-constrained variable state Viterbi as *Algorithm* $\mathcal{A}$ and the estimation algorithm proposed in this paper as *Algorithm* $\mathcal{B}$.

V. Experiments on 2-D Images

Our experiments with 2-D images were focused on comparing the proposed estimation algorithm ($\mathcal{B}$) to the algorithm based on variable state Viterbi ($\mathcal{A}$). The goal of the experiments was classification of aerial images into man-made and natural regions. The experimental setup was similar to the setup in [24]. Although classification results, for algorithm $\mathcal{A}$, are available in [24], we performed all the experiments again in order to correctly compare the learning and classification times of the two estimation programs, when run on similar processors.

The six images used are 512×512 grayscale images with 8 bits per-pixel (bpp). The images are the aerial images of the San Francisco Bay area provided by TRW [24]. The availability of hand-labeled classification for the six images proved helpful in evaluating the performances. The manually classified images were used as the truth set for our experiments. The six images and their respective hand-labeled classification are shown in Figs. 4 and 5.

The features extracted from images were the intra-block features used for experiments in [24]. No inter-block features were used in the experiments performed. For clarity, we elaborate the feature extraction process. The images were divided into 4×4 blocks, and the discrete cosine transform (DCT) coefficients or averages over them were calculated and used as features. If the DCT coefficients for a 4×4 block are denoted by $\{\mathcal{D}_{i,j} : i,j \in (0,1,2,3)\}$, the definitions of the six features used are

$$
\begin{aligned}
f_1 &= \mathcal{D}_{0,0} \\
f_2 &= |\mathcal{D}_{1,0}| \\
f_3 &= |\mathcal{D}_{0,1}| \\
f_4 &= \frac{1}{4}\sum_{i=2}^{3}\sum_{j=0}^{1}|D_{i,j}| \\
f_5 &= \frac{1}{4}\sum_{i=0}^{1}\sum_{j=2}^{3}|D_{i,j}| \\
f_6 &= \frac{1}{4}\sum_{i=2}^{3}\sum_{j=2}^{3}|D_{i,j}|.
\end{aligned}
$$

An initial K-means clustering was performed ($K = \text{number of classes}$). However, the number of states for each class (man-made class and natural class) were variable. For example, if M represents the total number of states of the model, then states $1,\ldots,L$ were assigned to the first class while $L+1,\ldots,M$ were assigned to the second class. The optimization in Viterbi was performed by constraining the feasible states to lie in the classes given by the training data. In case of multiple states per class, one of the designated states was randomly assigned to points, during initialization. The parameters L, and M were varied during the experiments. A six-fold cross validation was used to compare the classification performances of the algorithms. At each iteration, one image was used as test data and the other five were used as training

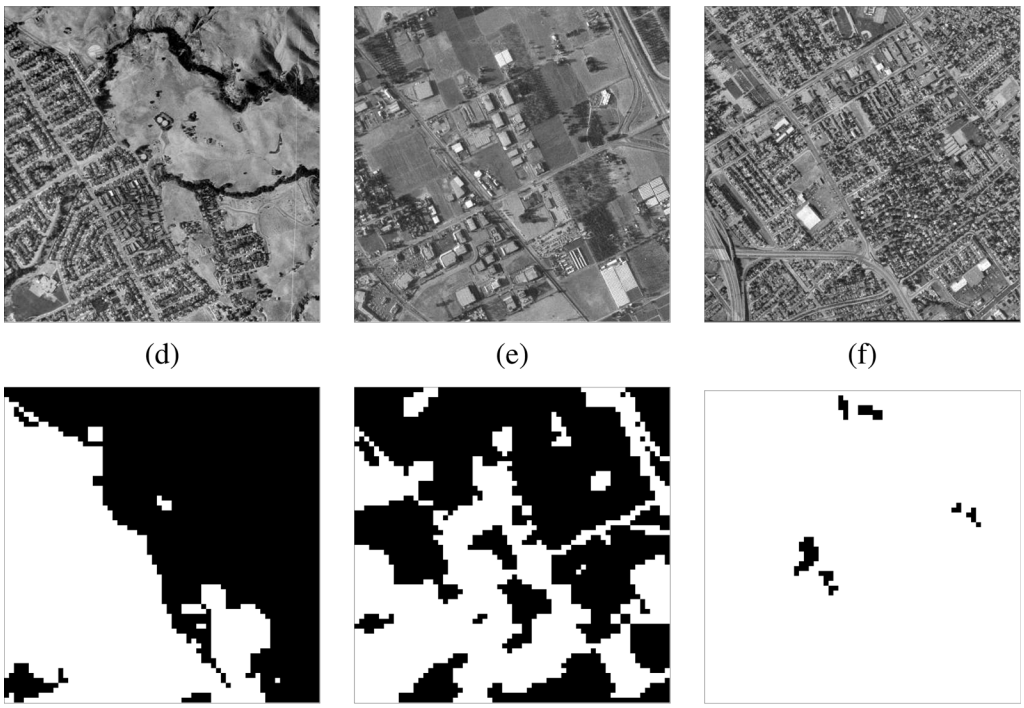

Fig. 5. (d)–(f) Three aerial images and their manual classification. Left: Original 8-bpp images, Right: Hand-labeled classified images. White represents man-made and gray represents natural.

TABLE I
PARAMETERS USED FOR ALGORITHM $\mathcal{B}$ FOR EACH ITERATION IN 2-D IMAGE CLASSIFICATION EXPERIMENT. N_1 AND N_2 REPRESENT THE NUMBER OF STATES FOR THE NATURAL AND MAN MADE CLASSES, RESPECTIVELY

Iteration Number	1	2	3	4	5	6
N_1	6	6	4	4	4	4
N_2	8	7	9	10	8	7
α	1.0	1.0	1.0	1.0	1.0	0.4

data. The five training images were tiled from top to bottom, to form one composite training image.

A regularization of 2-D transition probabilities toward 1-D probabilities was performed, during estimation, in a fashion similar to (5). The parameters to be used for each iteration (number of states to model each class and the value of α) were, in turn, estimated using a fivefold cross validation within each training data set. In our case, each training set consisted of five images and at each iteration, four of them were used for learning the model and one image was used as testing the model. The number of states per class was varied between 4 and 10, and α was varied between 0.0 and 1.0. At each step, the parameter combination yielding the best average performance over five iterations, was selected, for testing. The parameter combination used at each iteration is shown in Table I.

In our algorithm, the training phase involved estimating the optimal state configuration and the transition probabilities. At the testing phase, the learnt transition probabilities, means and covariances were used to find the optimal state configuration and, hence, the misclassification error. In this paper, we report the average performance over all six iterations. If the man-made class is assumed to be the target class, or the positive class, we define the following.

1) *Sensitivity* is the true positive ratio is the probability of detecting positive given that the truth is positive.
2) *Specificity* is the true negative ratio is the probability of detecting negative given that the truth is negative.
3) *Predictive positive value (PVP)* is the probability of being truly positive given a positive detection of the classifier.
4) *Misclassification error* $(\mathcal{P}_e)$ is the probability of not detecting the correct class (positive or negative).

Fig. 6 compares the classification performances of algorithms $\mathcal{A}$ and $\mathcal{B}$ for one particular image from the dataset. The original and hand labeled images are also shown and the classification error rates are indicated in the figure. The classification performances for the aerial images shown in Figs. 4 and 5 is reported in Table II. The values reported are the sensitivity, specificity, PVP and error rate averaged over all six iterations. All the experiments were performed on 2.6 GHz Xeon processors running Linux. The CPU times required by Algorithms $\mathcal{A}$ and $\mathcal{B}$ for learning and classification, respectively, are shown in Table III.

We notice from Tables II and III that as N is increased for Algorithm $\mathcal{A}$, the error rate decreases but the model learning and classification times go up. The average error rates $(\mathcal{P}_e)$ using algorithms $\mathcal{A}$ and $\mathcal{B}$ are found to be comparable. However, the running times for learning as well as classification using Algorithm $\mathcal{B}$ are much less compared to Algorithm $\mathcal{A}$. These results support the fact that the proposed algorithm $(\mathcal{B})$ has a lower computational complexity compared to algorithm $\mathcal{A}$.

VI. EXPERIMENTS ON 3-D IMAGES

The 3-D HMM has been applied to volume image segmentation in this paper. Experiments were performed using a large

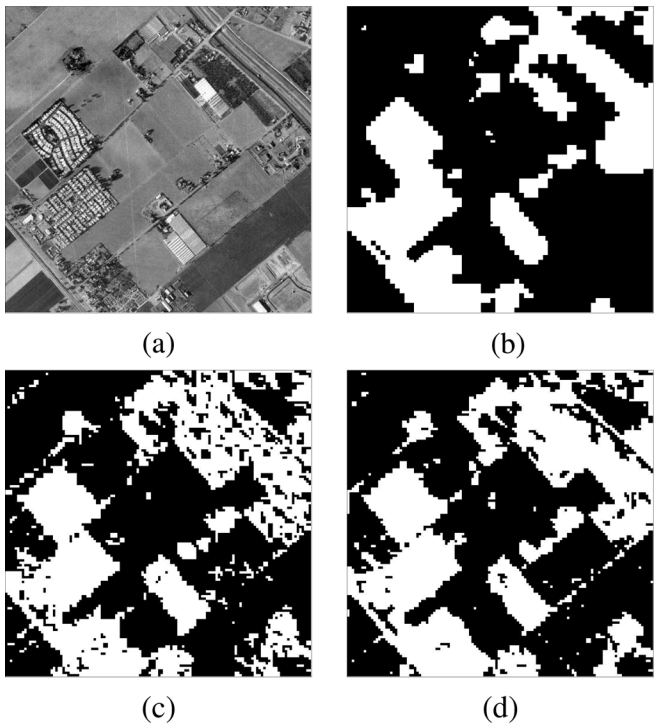

Fig. 6. Compare the performances of Algorithm $\mathcal{A}$ and $\mathcal{B}$. (a), (b) Original 8-bpp image and the hand-labeled classified image. (c) Algorithm $\mathcal{A}$ with classification error rate 15.8%. (d) Algorithm $\mathcal{B}$ with classification error rate 15.3%.

TABLE II
PERFORMANCE OF ALGORITHMS $\mathcal{A}$ AND $\mathcal{B}$ FOR 2-D IMAGE CLASSIFICATION. THE AVERAGE PERFORMANCE OVER SIX ITERATIONS IS SHOWN IN THE TABLE

Estimation algorithm	Sensitivity	Specificity	PVP	$\mathcal{P}_e$
$\mathcal{A}$ (N=8)	0.8034	0.7509	0.7728	0.2168
$\mathcal{A}$ (N=16)	0.8044	0.7615	0.7769	0.2149
$\mathcal{A}$ (N=32)	0.8012	0.7856	0.8013	0.1981
$\mathcal{B}$	0.7179	0.8822	0.8617	0.2011

TABLE III
RUNNING TIMES OF ALGORITHMS $\mathcal{A}$ AND $\mathcal{B}$ FOR 2-D IMAGE CLASSIFICATION

Estimation algorithm	Training time	Classification time
$\mathcal{A}$ (N=8)	36s	1s
$\mathcal{A}$ (N=16)	127s	5s
$\mathcal{A}$ (N=32)	474s	19s
$\mathcal{B}$	4s	1s

pool of synthetic volume images. Each image contained a standard mathematical 3-D shape centered about the image. The mathematical shapes used will be discussed later. The method of image generation was as follows. Points in the interior of the 3-D shape were assigned black color while the rest were white. Each color voxel, black ($\mu = 0$) and white ($\mu = 1$), was perturbed by an additive Gaussian noise $\sim N(0, \sigma^2)$ and the voxel values were truncated to lie in the interval $[-2\sigma, 1 + 2\sigma]$. This perturbed value formed the unidimensional feature for each voxel. For the purpose of displaying images, voxel values in the interval $[-2\sigma, 1 + 2\sigma]$ were scaled to $[0, 255]$. The 3-D shape parameters and the variance of noise were varied to form a large pool of images.

The term *class* denotes the classes into which an image is to be classified whereas *states* denotes the states of the 3-D HMM model. The number of classes is fixed by the classification problem definition whereas the number of states of the 3-D HMM model can be varied with experiments. When the number of classes K is known a priori, unsupervised segmentation using 3-D HMM involves the following steps.

1) An initial clustering using K-means algorithm is performed and each point is assigned a state depending upon its initial cluster. In case of more than one states per cluster, one of them is assigned randomly. The number of states, fixed for each cluster is arbitrary and is a parameter for the segmentation program. The number of clusters K corresponds to the number of classes, which is 2.
2) Initial parameters are estimated using ML criteria assuming the initial states as true states.
3) 3-D HMM-based unsupervised learning is performed iteratively. In other words, MAP states are assigned to points, as discussed before, and parameters re estimated iteratively.
4) Final classes are given to points based on the states they have been assigned by the algorithm.

A. Geometric Equations of 3-D Shapes

The mathematical equations for the topological shapes used for the experiments were adopted from *Mathworld*[1] and are given as follows.

1) *Torus*: A torus which is azimuthally symmetric about the z axis is represented by the following equation:
$$(c - \sqrt{(x^2 + y^2)})^2 + z^2 = a^2.$$
Here, c is the radius from the center of the hole to the center of the torus tube while a is the radius of the tube. A torus is depicted in Fig. 7(a).
2) *Ellipsoid*: An ellipsoid is a quadratic surface which is represented by the following equation:
$$\left(\frac{x^2}{a^2}\right) + \left(\frac{y^2}{b^2}\right) + \left(\frac{z^2}{c^2}\right) = 1.$$
Here, a, b and c represent the lengths of the semi-axes. An ellipsoid is depicted in Fig. 7(b).
3) *Hyperboloid*: A hyperboloid is a quadratic surface which is represented by the following equation:
$$\left(\frac{x^2}{a^2}\right) + \left(\frac{y^2}{b^2}\right) - \left(\frac{z^2}{c^2}\right) = 1.$$
Here, a, b, and c represent the lengths of the semi-axes. A hyperboloid is depicted in Fig. 7(c).

B. Algorithm for Shape Generation

We used the following simple algorithm to vary the parameters a, b, and c in order to generate images used for segmentation

[1]http://mathworld.wolfram.com

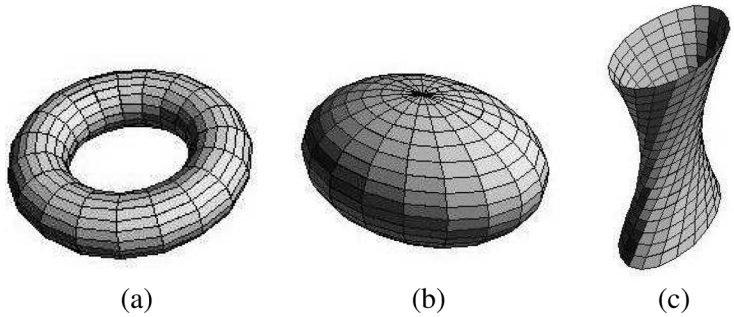

Fig. 7. Some 3-D shapes are illustrated in the figure. The pictures are adopted from Mathworld Website. The figures represent (a) Torus, (b) Ellipsoid, and (c) Hyperboloid.

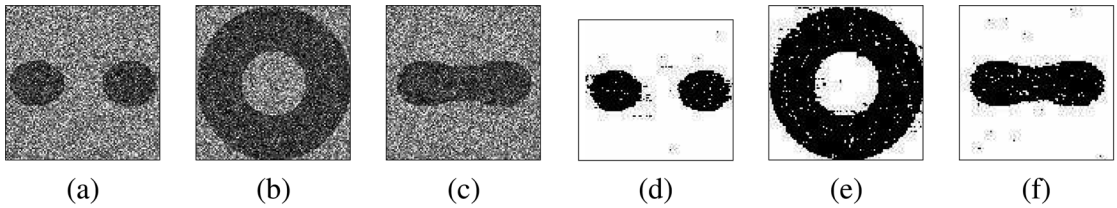

Fig. 8. Segmentation of a torus with 3-D HMM. The figure shows three frames, one in each of the X-Y, Y-Z and X-Z planes. (a)–(c) Original image. (d)–(f) Segmented image. The parameters are $a = 15$, $c = 35$, and $\sigma = 0.6$. The error rate is 1.2%.

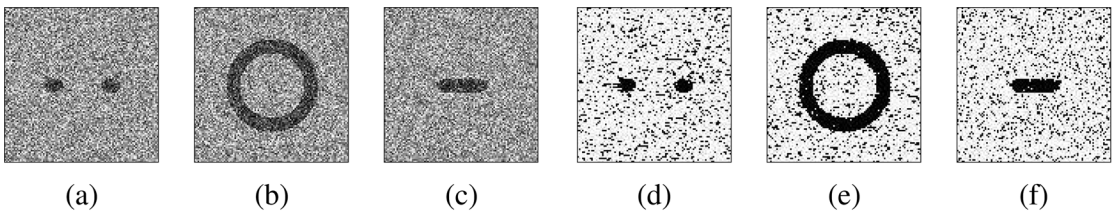

Fig. 9. Segmentation of a torus with 3-D HMM. The figure shows three frames, one in each of the X-Y, Y-Z and X-Z planes. (a)–(c) Original image. (d)–(f) Segmented image. The parameters are $a = 5$, $c = 25$, and $\sigma = 0.5$. The error rate is 12.8%.

study. For clarity, we restate that geometrically, the parameters a, b, and c represent the length of the semi-axes for ellipsoids and hyperboloids, while a and c represent the length of radii, for torii. In the following pseudo-code, subscripts *low*, *up*, and *step* represent the lower, upper limits, and the step size used to vary the respective parameter.

1) $a \leftarrow a_{\text{low}}$; $b \leftarrow b_{\text{low}}$; $c \leftarrow c_{\text{low}}$.
2) Generate 3-D image containing *Shape* (a, b, c) centered about the image containing Gaussian noise σ, as explained before.
3) If $c \leq c_{\text{up}}$; then $c \leftarrow c + c_{\text{step}}$ and goto step 2); else goto step 4).
4) If $b \leq b_{\text{up}}$; then $b \leftarrow b + b_{\text{step}}$ and goto step 2); else goto step 5).
5) If $a \leq a_{\text{up}}$; then $a \leftarrow a + a_{\text{step}}$ and goto step 2); else stop.

For the parametric studies performed here, the sizes of the images was fixed at $100 \times 100 \times 100$ and the limits, and the step sizes chosen were as follows:

- $a_{\text{low}} = 10$, $a_{\text{up}} = 40$, $a_{\text{step}} = 10$; $b_{\text{low}} = 10$, $b_{\text{up}} = 30$, $b_{\text{step}} = 10$; $c_{\text{low}} = a$, $c_{\text{up}} = 40$, $c_{\text{step}} = 10$.

A total of 70 images can be generated with these parameter settings (10 for class torii, 30 each for classes ellipsoids and hyperboloids) for a given noise parameter σ. The parameter σ was varied from 0.2 to 0.7 to generate images having a wide range of Gaussian noise.

C. Parametric Studies, Results and Discussion

The main advantage of evaluation of 3-D HMM under the chosen experimental setup is the availability of ground truth for every image. Moreover, performance over a large pool of images instills greater confidence in the suitability of the proposed algorithm for 3-D image segmentation. However, it must be mentioned that some of the resulting shapes are more difficult to segment than the others if certain extreme parameter setting is used (for example if the semi-major axes or the radii are too small). Figs. 8–13 illustrate segmentation of a few 3-D images containing torii, ellipsoids, and hyperboloids respectively. In each case, three frames of the original and segmented images are shown. The parameters used are indicated in the figure captions.

Three separate parametric studies were performed with the pool of 70 images in order to determine the sensitivity of the 3-D HMM algorithm to the various parameters. These are explained as follows.

1) The model parameter α was varied between 0 and 1 in steps of 0.2 and segmentation performance noted for each α. The number of states were fixed at 3 for each class. Results for $\sigma = 0.5$ and $\sigma = 0.6$ are shown here. For

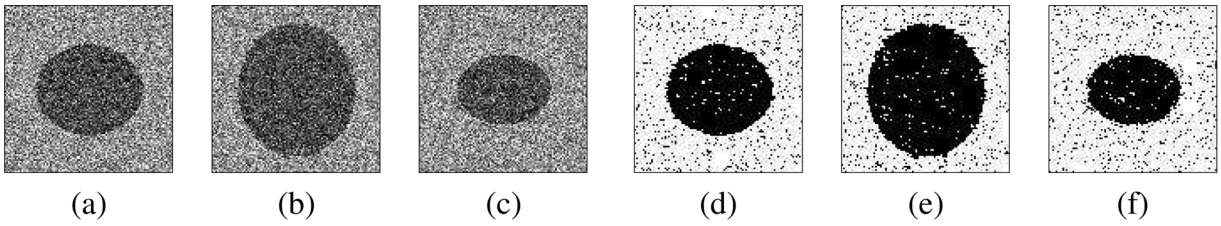

Fig. 10. Segmentation of an ellipsoid with 3-D HMM. The figure shows three frames, one in each of the X-Y, Y-Z and X-Z planes. (a)–(c) Original image. (d)–(f) Segmented image. The parameters are $a = 35$, $b = 40$, $c = 30$, and $\sigma = 0.6$. The error rate is 6.5%.

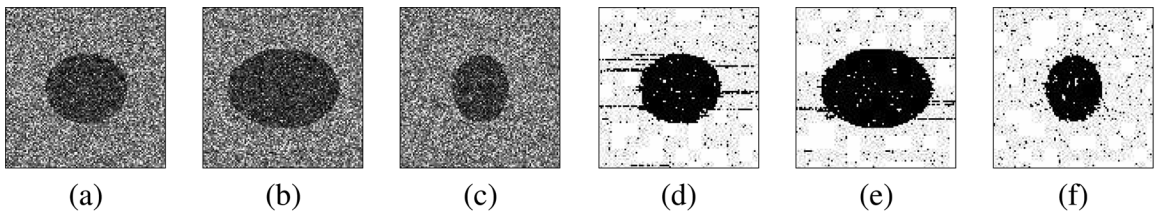

Fig. 11. Segmentation of an ellipsoid with 3-D HMM. The figure shows three frames, one in each of the X-Y, Y-Z and X-Z planes. (a)–(c) Original image. (d)–(f) Segmented image. The parameters are $a = 35$, $b = 25$, $c = 30$, and $\sigma = 0.6$. The error rate is 3.3%.

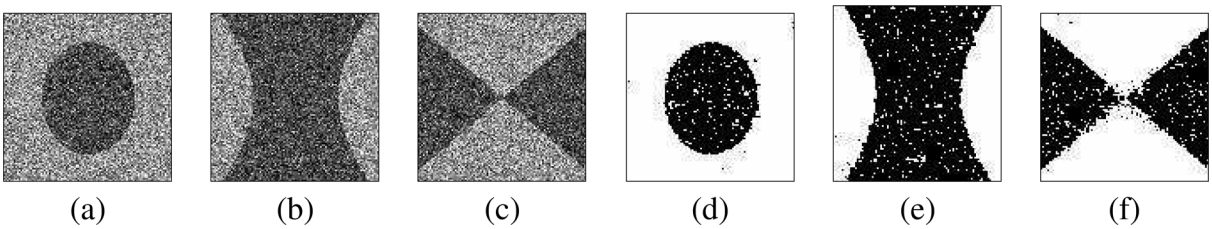

Fig. 12. Segmentation of a hyperboloid with 3-D HMM. The figure shows three frames, one in each of the X-Y, Y-Z and X-Z planes. (a)–(c) Original image. (d)–(f) Segmented image. The parameters are $a = 25$, $b = 35$, $c = 30$, and $\sigma = 0.6$. The error rate is 3.1%.

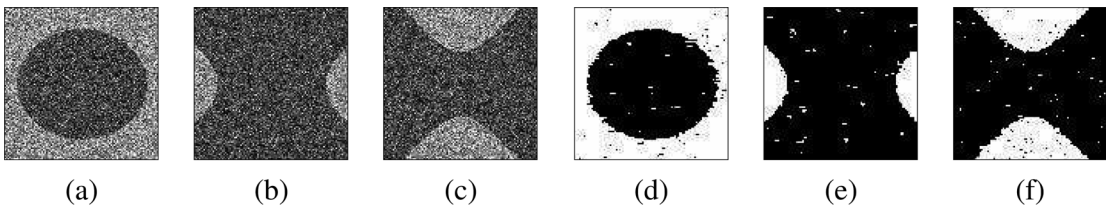

Fig. 13. Segmentation of a hyperboloid with 3-D HMM. The figure shows three frames, one in each of the X-Y, Y-Z and X-Z planes. (a)–(c) Original image. (d)–(f) Segmented image. The parameters are $a = 35$, $b = 25$, $c = 30$, and $\sigma = 0.6$. The error rate is 1.7%.

clarity, we have shown two tables (one each for $\sigma = 0.5$ and $\sigma = 0.6$) for torii, ellipsoids, and hyperboloids. Results for torii are shown in Tables IV and V; results for ellipsoids are shown in Tables VI and VII; while results for hyperboloids are shown in Tables VIII and IX. Due to lack of space, we have shown results for six images for each class and the respective shape parameters are also indicated. The best performance, which usually occurs at an intermediate value of α, is indicated in bold. The parameter α represents the extent of 3-D dependence in the model. A trade off between model complexity (complete 3-D model, $\alpha = 1$) and ease of estimation (2-D model, $\alpha = 0$), is preferred in most cases, and the results support this hypothesis.

2) The Gaussian noise parameter σ was varied between 0.2 and 0.7 in steps of 0.1. The number of states were fixed at 3 for each class and α was fixed at 0.6. Segmentation was performed for all 70 images for each σ. The best and median segmentation performances for different values of σ are shown in Table X. As is evident from the results, 3-D HMM performs reasonably well segmentation even for large values of σ.

3) The number of states for each class were varied from 3 to 9 in steps of 3. α was fixed at 0.6. The best and median performances for noise parameter $\sigma = 0.6$ are shown in Table XI. Results show that models with less number of states per class perform better than models with higher number of states per class.

We discovered that the best value of α varied largely with the shape to be segmented. Based on the experiments performed, an intermediate value of α, in the interval $[0.4, 0.6]$ usually performed better than extreme values. In the future, we would like to look into how we can theoretically characterize the performance of 3-D HMM model, with the smoothing parameter α.

TABLE IV
SEGMENTATION PERFORMANCES OF A FEW IMAGES OF CLASS TORII WITH SHOWN PARAMETERS (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.5$) AS THE PARAMETER α IS VARIED BETWEEN 0 AND 1. THE BEST PERFORMANCE IS INDICATED IN BOLD

a	c	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$
20	20	0.0144	0.0116	0.0068	0.0072	**0.0064**	0.0816
20	30	**0.0104**	0.0114	0.0110	0.0106	0.0105	0.0276
20	40	0.0129	0.0124	0.0120	**0.0115**	0.0119	0.0372
30	30	0.0365	0.0342	**0.0314**	0.0338	0.0372	0.0373
30	40	0.0338	0.0598	0.0449	0.0511	**0.0233**	0.0353
40	40	**0.0091**	0.0115	0.0468	0.0184	0.0687	0.0456

TABLE V
PERFORMANCES OF A FEW IMAGES OF CLASS TORII WITH SHOWN PARAMETERS (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.6$) AS THE PARAMETER α IS VARIED BETWEEN 0 AND 1. THE BEST PERFORMANCE IS INDICATED IN BOLD

a	c	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$
20	20	0.0261	**0.0170**	0.0212	0.0854	0.0472	0.0473
20	30	0.1334	0.0187	0.0412	0.0190	**0.0149**	0.0446
20	40	0.0215	0.0264	0.0305	**0.0164**	0.0170	0.0372
30	30	0.0565	**0.0433**	0.0497	0.0514	0.0515	0.0601
30	40	**0.0283**	0.0542	0.0711	0.0700	0.0630	0.0634
40	40	0.0045	**0.0045**	0.0048	0.1107	0.0450	0.1229

TABLE VI
SEGMENTATION PERFORMANCES OF A FEW IMAGES OF CLASS ELLIPSOIDS WITH SHOWN PARAMETERS (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.5$) AS THE PARAMETER α IS VARIED BETWEEN 0 AND 1. THE BEST PERFORMANCE IS INDICATED IN BOLD

a	b	c	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$
20	30	40	0.0767	0.0559	0.0617	0.0416	**0.0337**	0.1170
40	30	40	0.0068	**0.0056**	0.0080	0.0057	0.0292	0.0105
40	20	40	0.0083	0.0070	**0.0043**	0.0052	0.0776	0.2111
40	10	40	0.1600	0.1208	**0.0626**	0.1685	0.2411	0.2324
30	30	40	0.0264	0.0090	**0.0048**	0.0324	0.0063	0.0346
30	30	30	0.0490	0.0401	**0.0169**	0.0176	0.0253	0.1439

TABLE VII
SEGMENTATION PERFORMANCES OF A FEW IMAGES OF CLASS ELLIPSOIDS WITH SHOWN PARAMETERS (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.6$) AS THE PARAMETER α IS VARIED BETWEEN 0 AND 1. THE BEST PERFORMANCE IS INDICATED IN BOLD

a	b	c	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$
20	30	40	**0.0461**	0.1442	0.3037	0.0746	0.2687	0.2384
40	30	40	0.0126	0.0161	0.0089	**0.0089**	0.0585	0.0308
40	20	40	0.0912	**0.0064**	0.0119	0.1943	0.0340	0.0304
40	10	40	0.0740	**0.0576**	0.0121	0.2103	0.1179	0.2230
30	30	40	0.0093	0.0116	**0.0086**	0.0113	0.0108	0.0622
30	30	30	0.0676	0.0777	0.0698	0.0691	**0.0441**	0.2528

The running times of 3-D HMM segmentation program for image sizes $(w \times w \times w)$ where w takes values 50, 100, 150, and 200 were found out to be 32, 280, 798, and 938 s respectively. These numbers support the fact that the complexity of the algorithm is linear in the number of pixels in an image (w^3), when other parameters are constant.

TABLE VIII
SEGMENTATION PERFORMANCES OF A FEW IMAGES OF CLASS HYPERBOLOIDS WITH SHOWN PARAMETERS (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.5$) AS THE PARAMETER α IS VARIED BETWEEN 0 AND 1. THE BEST PERFORMANCE IS INDICATED IN BOLD

a	b	c	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$
10	10	10	0.0084	**0.0080**	0.0113	0.0096	0.0120	0.0138
10	20	30	0.0130	0.0148	0.0102	**0.0090**	0.0116	0.0154
10	30	30	0.0084	0.0078	**0.0074**	0.0080	0.0096	0.0245
10	20	40	0.0181	0.0144	0.0173	**0.0143**	0.0183	0.0337
10	30	40	0.0107	0.0119	**0.0097**	0.0117	0.0230	0.0541
20	30	20	**0.0074**	0.0076	0.0091	0.0187	0.0083	0.0352

TABLE IX
SEGMENTATION PERFORMANCES OF A FEW IMAGES OF CLASS HYPERBOLOIDS WITH SHOWN PARAMETERS (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.6$) AS THE PARAMETER α IS VARIED BETWEEN 0 AND 1. THE BEST PERFORMANCE IS INDICATED IN BOLD

a	b	c	$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$
10	10	10	0.0201	0.0192	0.0181	0.0239	**0.0112**	0.0132
10	20	30	0.0226	0.0231	0.0265	**0.0224**	0.0229	0.0612
10	30	30	0.0582	**0.0369**	0.0833	0.0970	0.0828	0.0770
10	20	40	0.0284	0.0271	0.0370	0.0398	**0.0199**	0.0687
10	30	40	0.0199	0.0175	**0.0124**	0.0162	0.0222	0.0607
20	30	20	0.0211	0.0142	**0.0119**	0.0149	0.0179	0.0895

TABLE X
BEST AND MEDIAN SEGMENTATION PERFORMANCES OVER 70 IMAGES (SIZE $100 \times 100 \times 100$) AS THE VARIANCE OF THE GAUSSIAN NOISE VARIES

σ	0.2	0.3	0.4	0.5	0.6	0.7
Best	0.0001	0.0005	0.0019	0.0037	0.0067	0.0058
Median	0.0004	0.0041	0.0157	0.0406	0.1207	0.1995

TABLE XI
BEST AND MEDIAN SEGMENTATION PERFORMANCES OVER 70 IMAGES (SIZE $100 \times 100 \times 100$ AND $\sigma = 0.6$) AS THE NUMBER OF STATES FOR EACH CLASS ARE VARIED

N_1	3	3	3	6	6	6	9	9	9
N_2	3	6	9	3	6	9	3	6	9
Best	0.0067	0.0112	0.0127	0.0057	0.0136	0.0192	0.0289	0.0592	0.0810
Median	0.1207	0.1366	0.1814	0.2887	0.2329	0.2609	0.3662	0.3220	0.3168

TABLE XII
SEGMENTATION PERFORMANCE OF A 3-D IMAGE OF AN ELLIPSOID (SIZE $100 \times 100 \times 100$, $a = 40$, $b = 25$, $c = 30$, $\sigma = 0.5$) UNDER VARYING DEGREES OF ROTATION. THE VALUE OF α USED IS 0.6

axis of rotation	0°	10°	20°	30°	40°	50°	60°	70°	80°	90°
X	0.005	0.005	0.004	0.005	0.032	0.009	0.022	0.066	0.014	0.052
Y	0.005	0.004	0.004	0.004	0.006	0.008	0.035	0.006	0.004	0.004
Z	0.005	0.004	0.004	0.067	0.038	0.090	0.006	0.015	0.041	0.031

D. Studying Effect of Rotation

In order to study the effect of lexicographic ordering on unsupervised segmentation performance, a 3-D image of an ellipsoid

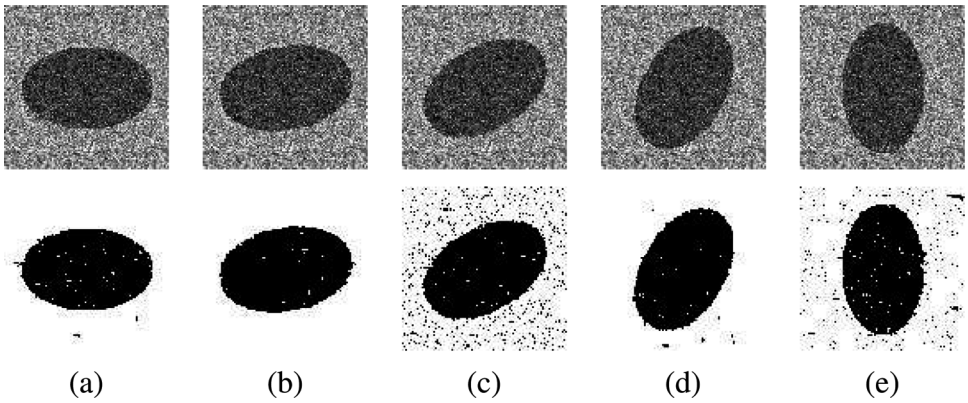

Fig. 14. Section of a 3-D image of an ellipsoid (size $100 \times 100 \times 100$, $a = 40$, $b = 25$, $c = 30$, $\sigma = 0.5$) under varying degrees of rotation. (a) 0°, (b) 10°, (c) 30°, (d) 60°, and (e) 90°.

($a = 40$, $b = 25$, $c = 30$, $\sigma = 0.5$), was subjected to independent rotations about the X, Y, and Z axes. The angle of rotation was varied from 0° to 90°, in steps of 10°.

The misclassification error rate for varying degrees of rotation is shown in Table XII. Fig. 14 shows a section of the original as well as segmented image under different angles of rotation. As can be seen from the results, segmentation performance is quite stable over a range of rotations.

VII. Conclusion and Future Work

In this paper, we proposed 3-D HMMs and implemented a computationally fast parameter estimation technique which can be used for both 2-D HMMs and 3-D HMMs. Supervised classification of 2-D aerial images was performed using 2-D HMMs with the proposed estimation algorithm and compared with the estimation algorithm which had been proposed initially for 2-D HMMs. The classification performances were found to be comparable while the running time of the proposed estimation algorithm was much faster than the old algorithm. Next, we performed unsupervised segmentation of a large number of synthetically generated 3-D images using 3-D HMM. Parametric studies varying model and noise parameters were performed and results were reported. Results show that 3-D HMM can be used as a powerful tool for modeling 3-D images. We identify the following issues for future work.

- It would be interesting to compare 3-D HMM-based segmentation with segmentation based on 3-D MRF algorithms.
- Multiresolution information has proved helpful in 2-D image classification. 2-D multiresolution HMMs (2-D MHMMs) have been successfully applied to 2-D image modeling. In 2-D MHMMs, feature vectors are extracted from an image at multiple resolutions and spatially arranged on a pyramid grid. Thus, a 2-D MHMM models both intra and inter-resolution relationships between extracted feature vectors. The proposed estimation algorithm can be incorporated into 2-D multiresolution HMMs for better computational efficiency. Extending 3-D HMMs to 3-D multiresolution HMMs is also of considerable interest to us.
- A plausible application of 3-D HMM could be in hyper-spectral image analysis and classification where the third dimension is the spectral dimension. Our 3-D HMM model can capture statistical dependence among a pixel and its spatial or spectral neighbors and would, thus, be of use. Another application of our model could be in video modeling, where the third dimension is the temporal dimension.

Appendix

We now prove (7), that is

$$\mathbf{s}^*_{j,k} = \arg\max_{\mathbf{s}_{j,k}} P\big(\mathbf{s}_{j,k} \mid \mathbf{u}_{j,k}, \mathbf{u}_{j',k'}, \bar{\mathbf{s}}_{j',k'} : (j',k') \neq (j,k), (j',k') \in \mathcal{D}\big) \quad (10)$$

is solved by (11), shown at the bottom of the page. Define $\mathcal{D}_1 = \{(j',k') : (j',k') < (j,k)\}$ and $\mathcal{D}_2 = \{(j'',k'') : (j'',k'') > (j,k)\}$, which are the collections of indices for rows preceding $R_{j,k}$ and those succeeding $R_{j,k}$, respectively (see the equation shown at the bottom of the next page).

The second equality is obtained by the chain rule of conditional probability and the assumption that given the state of a point, the observed vector is conditionally independent of any

$$\begin{aligned}\mathbf{s}^*_{j,k} = \arg\max_{\mathbf{s}_{j,k}} \sum_{i=0}^{w-1} \big(&\log P(u_{i,j,k} \mid s_{i,j,k}) \\ &+ \log P(s_{i,j,k} \mid \bar{s}_{i,j,k-1}, s_{i-1,j,k}, \bar{s}_{i,j-1,k}) \\ &+ \log P(\bar{s}_{i,j+1,k} \mid \bar{s}_{i,j+1,k-1}, \bar{s}_{i-1,j+1,k}, s_{i,j,k}) \\ &+ \log P(\bar{s}_{i,j,k+1} \mid s_{i,j,k}, \bar{s}_{i-1,j,k+1}, \bar{s}_{i,j-1,k+1})\big)\end{aligned} \quad (11)$$

other point. By deleting terms that are irrelevant to $\mathbf{s}_{j,k}$, we have

$$\mathbf{s}^*_{j,k} = \arg\max_{\mathbf{s}_{j,k}} P\big(\mathbf{s}_{j,k}, \mathbf{u}_{j,k} \mid \bar{\mathbf{s}}_{j',k'} : (j',k') \in \mathcal{D}_1\big) \times P\big(\bar{\mathbf{s}}_{j'',k''} : (j'',k'') \in \mathcal{D}_2 \mid \mathbf{s}_{j,k}, \bar{\mathbf{s}}_{j',k'}(j',k') \in \mathcal{D}_1\big). \quad (12)$$

Since given the states of all the preceding points, the probability of a point being in a certain state only depends on the three neighboring points that are one position ahead along each axis, the first term in (12) can be reduced to

$$\begin{aligned} &P\big(\mathbf{s}_{j,k}, \mathbf{u}_{j,k} \mid \bar{\mathbf{s}}_{j',k'} : (j',k') \in \mathcal{D}_1\big) \\ &\quad = P\big(\mathbf{s}_{j,k}, \mathbf{u}_{j,k} \mid \bar{\mathbf{s}}_{j,k-1}, \bar{\mathbf{s}}_{j-1,k}\big) \\ &\quad = P\big(\mathbf{s}_{j,k} \mid \bar{\mathbf{s}}_{j,k-1}, \bar{\mathbf{s}}_{j-1,k}\big) P\left(\mathbf{u}_{j,k} \mid \mathbf{s}_{j,k}\right). \end{aligned} \quad (13)$$

The second term in (12) can be expanded by the chain rule of conditional probability and the 3-D Markovian assumption on the states

$$\begin{aligned} &P\left(\bar{\mathbf{s}}_{j'',k''} : (j'',k'') \in \mathcal{D}_2 \mid \mathbf{s}_{j,k}, \bar{\mathbf{s}}_{j',k'} : (j',k') \in \mathcal{D}_1\right) \\ &\quad = P\left(\bar{\mathbf{s}}_{j+1,k} \mid \bar{\mathbf{s}}_{j+1,k-1}, \mathbf{s}_{j,k}\right) P\left(\bar{\mathbf{s}}_{j,k+1} \mid \mathbf{s}_{j,k}, \bar{\mathbf{s}}_{j-1,k+1}\right) \\ &\qquad \times \prod_{(j'',k'') \in \mathcal{D}_3} P\left(\bar{\mathbf{s}}_{j'',k''} \mid \bar{\mathbf{s}}_{j'',k''-1}, \bar{\mathbf{s}}_{j''-1,k''}\right) \end{aligned} \quad (14)$$

where $\mathcal{D}_3 = \mathcal{D}_2 - \{(j+1,k),(j,k+1)\}$. Note that the last term in (14) does not involve $\mathbf{s}_{j,k}$ and, hence, can be dropped when searching for $\mathbf{s}^*_{j,k}$. Combining (12)–(14), we have

$$\begin{aligned} \mathbf{s}^*_{j,k} &= \arg\max_{\mathbf{s}_{j,k}} P\left(\mathbf{s}_{j,k} \mid \bar{\mathbf{s}}_{j,k-1}, \bar{\mathbf{s}}_{j-1,k}\right) P\left(\mathbf{u}_{j,k} \mid \mathbf{s}_{j,k}\right) \\ &\quad \times P\left(\bar{\mathbf{s}}_{j+1,k} \mid \bar{\mathbf{s}}_{j+1,k-1}, \mathbf{s}_{j,k}\right) \\ &\quad \times P\left(\bar{\mathbf{s}}_{j,k+1} \mid \mathbf{s}_{j,k}, \bar{\mathbf{s}}_{j-1,k+1}\right) \\ &= \arg\max_{\mathbf{s}_{j,k}} \prod_{i=0}^{w-1} [P(u_{i,j,k} \mid s_{i,j,k}) \\ &\quad \times P(s_{i,j,k} \mid \bar{s}_{i,j,k-1}, s_{i-1,j,k}, \bar{s}_{i,j-1,k}) \\ &\quad \times P(\bar{s}_{i,j+1,k} \mid \bar{s}_{i,j+1,k-1}, \bar{s}_{i-1,j+1,k}, s_{i,j,k}) \\ &\quad \times P(\bar{s}_{i,j,k+1} \mid s_{i,j,k}, \bar{s}_{i-1,j,k+1}, \bar{s}_{i,j-1,k+1})]. \end{aligned}$$

The last equality comes from expanding the conditional probabilities of states along the rows. For computational convenience, we can search equivalently the maximum of the log of the objective function, which leads to (11).

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers of the initial manuscript.

REFERENCES

[1] J. Boreczky and L. Wilcox, "A hidden Markov model framework for video segmentation using audio and image features," in *Proc. IEEE Conf. Acoustics, Speech, Signal Processing*, vol. 6, 1998, pp. 3741–3744.

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

[3] "Markov random fields with efficient approximations," in *Proc. Computer Vision Pattern Recognition*, Santa Barbara, CA, Jun. 1998, pp. 648–655.

[4] F. Champagnat, J. Idler, and Y. Goussard, "Stationary Markov random fields on a finite rectangular lattice," *IEEE Trans. Inf. Theory*, vol. 44, no. 7, pp. 2901–2916, Jul. 1998.

[5] R. Chellappa and S. Chatterjee, "Classification of textures using Gaussian Markov random fields," *IEEE Trans. Acoustics, Speech, Signal Process.*, vol. ASSP-33, no. 4, pp. 959–963, Aug. 1985.

[6] R. Chellappa and A. K. Jain, *Markov Random Fields: Theory and Applications*. New York: Academic, 1993.

[7] G. R. Cross and A. K. Jain, "Markov random field texture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-5, no. 1, pp. 25–39, Jan. 1983.

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

[9] H. Derin, H. Elliott, H. Cristi, and D. Geman, "Bayes smoothing algorithms for segmentation of images modeled by Markov random fields," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, San Diego, CA, Mar. 1984, pp. 682–685.

[10] P. L. Dobruschin, "The description of a random field by means of conditional probabilities and conditions of its regularity," *Theory Probl. Appl.*, vol. 13, no. 2, pp. 197–224, 1968.

[11] S. Eickeler, S. Muller, and G. Rigoll, "Improved face recognition using pseudo 2-D hidden Markov models," presented at the Workshop Advances in Facial Image Analysis and Recognition Technology in conjunction with ECCV, Freiburg, Germany, Jun. 1998.

[12] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 6, pp. 721–741, Jun. 1984.

[13] Z. Ghahramani, "An introduction to hidden Markov models and Bayesian networks," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 15, no. 1, pp. 9–42, 2001.

[14] M. I. Gurelli and L. Onural, "On a parameter estimation method for Gibbs-Markov random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 4, pp. 424–430, Apr. 1994.

[15] F. R. Hansen and H. Elliott, "Image segmentation using simple Markov field models," *Comput. Graph. Image Process.*, vol. 20, no. 8, pp. 101–132, 1982.

[16] D. S. Hochbaum, "An efficient algorithm for image segmentation, Markov random fields and related problems," *J. ACM*, vol. 48, no. 4, pp. 686–701, 2001.

[17] X. D. Huang, Y. Ariki, and M. A. Jack, *Hidden Markov Models for Speech Recognition*. Edinburgh, U.K.: Edinburgh Univ. Press, 1990.

[18] F. Hulsken, F. Wallhoff, and G. Rigoll, "Facial expression recognition with pseudo-3D hidden Markov models," in *Proc. DAGM-Symp., Lecture Notes Computer Science*, vol. 2191, 2001, pp. 291–297.

[19] F. -C. Jeng and J. W. Woods, "Compound Gauss-Markov random fields for image estimation," *IEEE Trans. Signal Process.*, vol. 39, no. 3, pp. 683–697, Mar. 1991.

$$\begin{aligned} \mathbf{s}^*_{j,k} &= \arg\max_{\mathbf{s}_{j,k}} P\big(\mathbf{s}_{j,k} \mid \mathbf{u}_{j,k}, \mathbf{u}_{j',k'}, \bar{\mathbf{s}}_{j',k'} : (j',k') \neq (j,k), (j',k') \in \mathcal{D}\big) \\ &\overset{(1)}{=} \arg\max_{\mathbf{s}_{j,k}} P\big(\mathbf{s}_{j,k}, \mathbf{u}_{j,k}, \bar{\mathbf{s}}_{j',k'}, \mathbf{u}_{j',k'}, \bar{\mathbf{s}}_{j'',k''}, \mathbf{u}_{j'',k''} : (j',k') \in \mathcal{D}_1, (j'',k'') \in \mathcal{D}_2\big) \\ &\overset{(2)}{=} \arg\max_{\mathbf{s}_{j,k}} P\big(\bar{\mathbf{s}}_{j',k'}, \mathbf{u}_{j',k'} : (j',k') \in \mathcal{D}_1\big) \times P\big(\mathbf{s}_{j,k}, \mathbf{u}_{j,k} \mid \bar{\mathbf{s}}_{j',k'} : (j',k') \in \mathcal{D}_1\big) \\ &\quad \times P\big(\bar{\mathbf{s}}_{j'',k''} : (j'',k'') \in \mathcal{D}_2 \mid \mathbf{s}_{j,k}, \bar{\mathbf{s}}_{j',k'} : (j',k') \in \mathcal{D}_1\big) \\ &\quad \times P\big(\mathbf{u}_{j'',k''} : (j'',k'') \in \mathcal{D}_2 \mid \bar{\mathbf{s}}_{j'',k''} : (j'',k'') \in \mathcal{D}_2\big) \end{aligned}$$

[20] R. Kindermann and L. Snell, *Markov Random Fields and Their Applications*. Providence, RI: Amer. Math. Soc., 1980.
[21] J. Li, R. M. Gray, and R. A. Olshen, "Joint image compression and classification with vector quantization and a two dimensional hidden Markov model," in *Proc. Data Compression Conf.*, Snowbird, UT, Mar. 1999, pp. 23–32.
[22] ——, "Multiresolution image classification by hierarchical modeling with two dimensional hidden Markov models," *IEEE Trans. Inf. Theory*, vol. 46, no. 5, pp. 1826–1841, May 2000.
[23] J. Li and R. M. Gray, *Image Segmentation and Compression Using Hidden Markov Models*. New York: Kluwer, 2000.
[24] J. Li, A. Najmi, and R. M. Gray, "Image classification by a two dimensional hidden Markov model," *IEEE Trans. Signal Process.*, vol. 48, no. 2, pp. 517–533, Feb. 2000.
[25] J. Li and J. Z. Wang, "Automatic linguistic indexing of pictures by a statistical modeling approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1075–1088, Sep. 2003.
[26] S. Z. Li, *Markov Random Field Modeling in Computer Vision*. London, U.K.: Springer-Verlag, 1995.
[27] M. Lievin, N. Hanssen, P. Zerfass, and E. Keeve, "3D Markov random fields and region growing for interactive segmentation of MR data," in *Proc. Medical Image Computing Computer-Assisted Intervention*, Utrecht, The Netherlands, Oct. 2001, pp. 1191–1192.
[28] M. R. Leuttgen, W. C. Karl, A. S. Willsky, and R. R. Tenney, "Multiscale representations of Markov random fields," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3377–3396, Dec. 1993.
[29] B. S. Manjunath and R. Chellappa, "Unsupervised texture segmentation using Markov random field models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 5, pp. 478–482, May 1991.
[30] G. J. McLachlan and D. Peel, *Finite Mixture Models*. New York: Wiley, 2000.
[31] D. E. Melas and S. P. Wilson, "Double Markov random fields and Bayesian image segmentation," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 357–365, Feb. 2002.
[32] D. Panjwani and G. Healey, "Markov random field models for unsupervised segmentation of textured color images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 10, pp. 939–954, Oct. 1995.
[33] W. Pieczynski and A.-N. Tebbache, "Pairwise Markov random fields and its application in textured images segmentation," in *Proc. Southwest Symp. Image Analysis and Interpretation*, Austin, TX, Apr. 2000, pp. 106–110.
[34] H. Rue, I. Steinsland, and S. Erland, "Approximating hidden Gaussian Markov random fields," *J. Roy. Statist. Soc.*, vol. 66, no. 4, pp. 877–892, 2004.
[35] C. S. Won and H. Derin, "Unsupervised segmentation of noisy and textured images using Markov random fields," *Graph. Models Image Process.*, vol. 54, no. 4, pp. 308–328, 1992.
[36] L. Xie, S. F. Chang, A. Divakaran, and H. Sun, "Learning hierarchical hidden Markov Models for video structure discovery," ADVENT Group, Columbia Univ., New York, 2002, L. Xie, S. F. Chang, A. Divakaran, H. Sun, "Unsupervised discovery of multilevel statistical video structures using hierarchical hidden Markov models," *Proc. International Conf. Multimedia Expo.*, vol. 3, pp. 29-32, Baltimore, MD, Jul. 2003.
[37] S. Young, J. Jansen, J. Odell, D. Ollason, and P. Woodland, *HTK—Hidden Markov Model Toolkit*. Cambridge, U.K.: Cambridge Univ., 1995.
[38] J. Zhang, J. W. Modestino, and D. A. Langan, "Maximum-likelihood parameter estimation for unsupervised stochastic model-based image segmentation," *IEEE Trans. Image Process.*, vol. 3, no. 4, pp. 404–420, Jul. 1994.
[39] Y. Zhang, M. Brady, and S. Smith, "Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm," *IEEE Trans. Med. Imag.*, vol. 20, no. 1, pp. 45–57, Jan. 2001.
[40] S. C. Zhu, X. W. Liu, and Y. N. Wu, "Exploring texture ensembles by efficient Markov chain Monte Carlo—Toward a Trichromacy theory of texture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 6, pp. 554–569, Jun. 2000.

Dhiraj Joshi (S'02) received the Integrated Masters degree in mathematics and scientific computing from the Indian Institute of Technology (IIT), Kanpur, in 2002, where he worked on real-parameter evolutionary algorithms. He is currently pursuing the Ph.D. degree in computer science and engineering at The Pennsylvania State University (Penn State), University Park.

He is currently a Research Assistant at the Intelligent Information Systems Research Laboratory, Penn State. He was a Research Intern at the Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP), Switzerland, where he worked on social network analysis. His current research interests include statistical modeling, content analysis, and multimedia information retrieval.

Jia Li (S'95–M'99–SM'05) received the B.S. degree in electrical engineering from Xi'an Jiao Tong University, China, in 1993, and the M.Sc. degree in electrical engineering, the M.Sc. degree in statistics, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1995, 1998, and 1999, respectively.

She is an Assosciate Professor of statistics and by courtesy appointment in computer science and engineering at The Pennsylvania State University, University Park. She was a Research Associate in the Computer Science Department, Stanford University, in 1999. She was a Researcher at the Xerox Palo Alto Research Center, Palo Alto, CA, from 1999 to 2000. Her research interests include statistical classification and modeling, data mining, bioinformatics, image processing, and image retrieval.

James Z. Wang (S'96–M'00–SM'06) received the B.S. degree (summa cum laude) in mathematics and computer science from the University of Minnesota, Minneapolis, in 1994, the M.Sc. degrees in mathematics and computer science from Stanford University, Stanford, CA, in 1997, and the Ph.D. degree in medical information sciences from the Biomedical Informatics Program and Computer Science Database Group, Stanford University, in 2000.

He is an Associate Professor at the College of Information Sciences and Technology and the Department of Computer Science and Engineering, The Pennsylvania State University, University Park. From 2000 to 2006, he was the holder of the endowed PNC Technologies Career Development Professorship.

Dr. Wang is a recipient of an NSF Career award in support of his research program.