A NEW BAYESIAN APPROACH TO TEXTURED IMAGE SEGMENTATION: TURBO SEGMENTATION

Frederic Lehmann

INSTITUT TELECOM, TELECOM SudParis, departement CITI, UMR-CNRS 5157
9, rue Charles Fourier, 91011, Evry, France
phone: + (33) 1 60 76 46 33, fax: + (33) 1 60 76 44 33, email: frederic.lehmann@it-sudparis.eu

ABSTRACT

We consider the problem of semi-supervised segmentation of textured images. In this paper, we propose a new Bayesian framework by modeling two-dimensional textured images as the concatenation of two one-dimensional hidden Markov autoregressive models for the lines and the columns, respectively. A new segmentation algorithm, which is similar to turbo decoding in the context of error-correcting codes, is obtained based on a factor graph approach. The proposed method estimates the unknown parameters using the Expectation-Maximization algorithm.

1. INTRODUCTION

An image texture can be defined as the local spatial variations in pixel intensities and orientation [1]. In order to recognize objects and scenes in computer vision, it is essential to be able to partition an image into meaningful regions with respect to texture characteristics. This task, referred to as texture segmentation in the image processing literature, is a challenging problem due to the complexity and diversity of natural textures. We restrict the focus of this paper to semi-supervised segmentation, where the number of texture classes, denoted by M, is known, but no ground truth data are available to train the features or parameters associated with each class.

We consider model-based segmentation, where the textures are described by a stochastic process. Existing methods model the intensity field as a Gauss-Markov random field (GMRF) [2], which takes into account the spatial dependencies between the pixels. The segmentation task can be carried out in several ways. One possibility is to apply a classical clustering algorithm to texture features based on the estimated GMRF parameters, such as a neural network or the *k-means* algorithm [3]. A more accurate approach is obtained by labeling the pixels to one of M texture classes and modeling the label field as a Markov random field (MRF), so that pixels close together tend to have the same texture class. The labels of all pixels can then be estimated using simulated annealing [4].

In this paper, we propose a texture segmentation method based on one-dimensional hidden Markov autoregressive models (1D HMM-AR). The main advantage of the 1D HMM-AR over the MRF approach is that segmentation and parameter estimation can be performed at lower computational cost with a flexible and versatile forward-backward procedure known as the Baum-Welch algorithm [5]. However, a single 1D HMM-AR is unable to capture the 2D properties of textures. Therefore, we propose to handle 2D textured images by converting them to two 1D signals by scanning with a horizontal and a vertical raster scan. Each 1D signal, corresponding respectively to the lines and the columns of the pixels, is modeled with a 1D HMM-AR. Using the factor graph framework [7], a new segmentation algorithm is obtained by applying the sum-product algorithm [7] to a factor graph representing the joint *a posteriori* probability distribution of the class labels given the pixel intensities. This scheme is analog to the turbo decoding algorithm proposed by Berrou [8] in the context of error-correcting codes. Hence the name turbo segmentation for the proposed method. Moreover, the unknown parameters can be estimated using the Expectation-Maximization (EM) framework [9].

This paper is organized as follows. Sec. 2 presents our new model for textured images based on two 1D HMM-AR corresponding to the lines and the columns of the pixels. The derivation of turbo segmentation using a factor graph approach is introduced in Sec. 3. Finally, experimental results are given in Sec. 4 to investigate the performances of the proposed method.

2. PROPOSED MODEL FOR TEXTURED IMAGES

In this section, we introduce a 1D HMM-AR model of a textured image (Sec. 2.1). The model is visualized by means of a factor graph. In order to capture the 2D nature of textures, the image must be modeled as the concatenation of two 1D HMM-AR, one for the lines and another for the columns (Sec. 2.2). The complete factor graph corresponding to the concatenated HMM-AR model is also drawn.

2.1 1D HMM-AR model

Let $S = \{s = (i, j), 1 \le i \le H, 1 \le j \le W\}$ be a 2D lattice representing the grid points of an image. It is well known that a 2D image can be converted to a 1D signal through horizontal raster scanning. This is achieved by visiting the points of the 2D lattice S, using the space filling curve given by Fig. 1 a). With this pixel ordering, the texture labels and the gray-level pixel values can be written as $\{l_k\}$ and $\{y_k\}$, respectively, where the discrete time index $k \in \{1, 2, \ldots, WH\}$.

We would like our model to capture the spatial dependencies between pixel values for each texture class. Let $\mathbf{B}(n) = [a_1(n), \ldots, a_P(n)]^T$ and $v(n)$ represent the coefficients and the variance corresponding to the n-th texture class autoregressive (AR) model, $n \in \{1, \ldots, M\}$. We assume an HMM-AR model of order P [6] defined by the difference equation

$$y_k = \sum_{p=1}^{P} a_p(l_k) y_{k-p} + n_k(l_k), \quad 1 \le k \le WH, \tag{1}$$

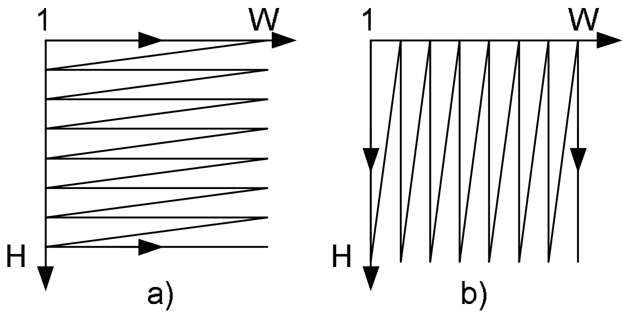

Figure 1: a) Horizontal raster scanning - b) Vertical raster scanning.

where $n_k(l_k)$ is an independent zero mean Gaussian noise of variance $v(l_k)$. The texture labels are assimilated to hidden states whose dynamics are governed by the transition matrix $\mathbf{P} = \{p_{m,n}\}$, where

$$p_{m,n} = P(l_k = n | l_{k-1} = m), \quad 1 \leq m,n \leq M.$$

We collect the unknown parameters of the HMM-AR in $\lambda = (\mathbf{P}, \mathbf{B}(n), v(n), n \in \{1, \ldots, M\})$.

We define the vector of hidden labels as $l_{1:WH} = (l_1, l_2, \ldots, l_{WH})$ and the vector of pixel intensities as $y_{1:WH} = (y_1, y_2, \ldots, y_{WH})$. Under the assumption of first-order Markov model for the label process, we obtain the factorization

$$\begin{aligned} p(l_{1:WH}|y_{1:WH}) \propto\ & p(y_1|l_1)P(l_1)p(y_2|l_2,y_1)P(l_2|l_1) \times \ldots \\ & \times p(y_P|l_P, y_{P-1}, \ldots, y_1)P(l_P|l_{P-1}) \\ & \times \prod_{k=P+1}^{WH} p(y_k|l_k, y_{k-1}, \ldots, y_{k-P})P(l_k|l_{k-1}). \end{aligned} \tag{2}$$

From an image modeling viewpoint, it also seems reasonable to assume that the texture label at the beginning of a new line is independent from the label at the end of the previous line, so that the lines become independent. With this simplification, the corresponding factor graph [7] is depicted in Fig. 2 (when $P = 2$). Variable nodes are represented as circles and the local functions appearing in the factorization of the *a posteriori* probability density function, denoted by

$$\begin{cases} f_k = P(l_k|l_{k-1}) \\ g_k = p(y_k|l_k, y_{k-1}, \ldots, y_{k-P}) \end{cases} \tag{3}$$

are represented as squares.

2.2 Concatenated HMM-AR model

The 1D HMM-AR introduced in Sec. 2.1 models the dynamics of the texture labels as a discrete-valued Markov chain along the lines of the image. In the sequel, this model will be referred to as the horizontal 1D HMM-AR. In general, this is not sufficient to capture the 2D features of textures. Visiting the points of the 2D lattice S with the vertical raster scanning curve given by Fig. 1 b), we define a second 1D HMM-AR to model the dependencies along the columns. In the sequel,

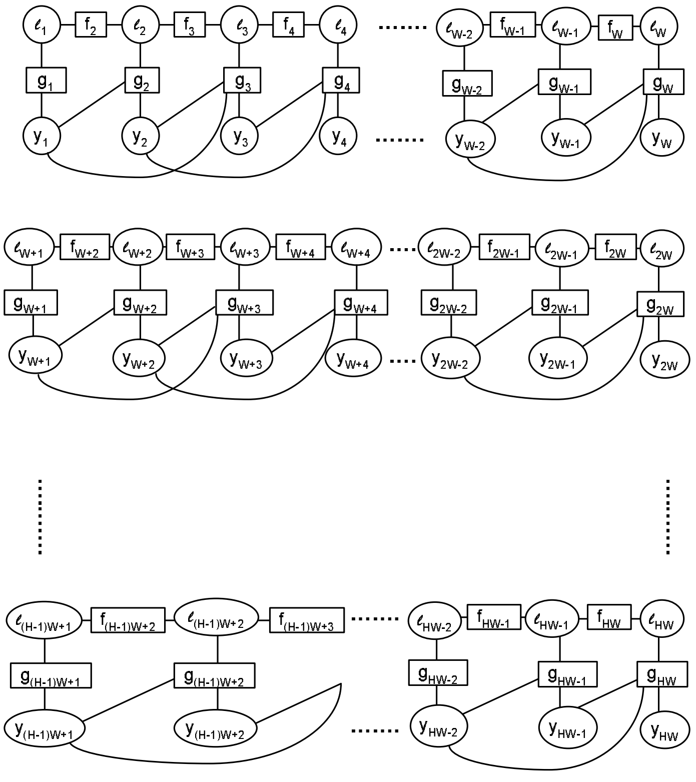
Figure 2: Factor graph of an HMM-AR model of order-2 for horizontal raster scanning of the pixels.

this model will be referred to as the vertical 1D HMM-AR. The corresponding factor graph is the same as the one depicted in Fig. 2 (when the order $P = 2$), except that the line dependencies are replaced by column dependencies.

As a result, each pixel is now associated with two texture labels, corresponding to the horizontal and vertical 1D HMM-AR model, respectively. The first (resp. second) label is governed by the dynamics imposed by the state transition matrix of the horizontal (resp. vertical) 1D HMM-AR. Since both labels correspond in fact to the same pixel, we must impose an equality constraint on those variables. We call the resulting model a concatenated HMM-AR model. The corresponding factor graph (ignoring the local functions g_k corresponding to pixel intensities for the sake of readability) is drawn in Fig. 3. The graph in thick (resp. thin) line corresponds to the horizontal (resp. vertical) 1D HMM-AR model of a textured image. Equality constraints between hidden labels are represented in a box.

3. TURBO SEGMENTATION

3.1 Bayesian inference algorithm

We apply the sum-product algorithm (SPA) [7] to the complete factor graph in Fig. 3, which implements belief propagation [7].

In order to derive the messages exchanged by the SPA, we focus on a small portion of the complete factor graph centered at time index k, illustrated by Fig. 4. This small portion corresponds to the horizontal 1D HMM-AR, whose parameters $\lambda = (\mathbf{P}, \mathbf{B}(n), v(n), n \in \{1, \ldots, M\})$ are fixed. Of course,

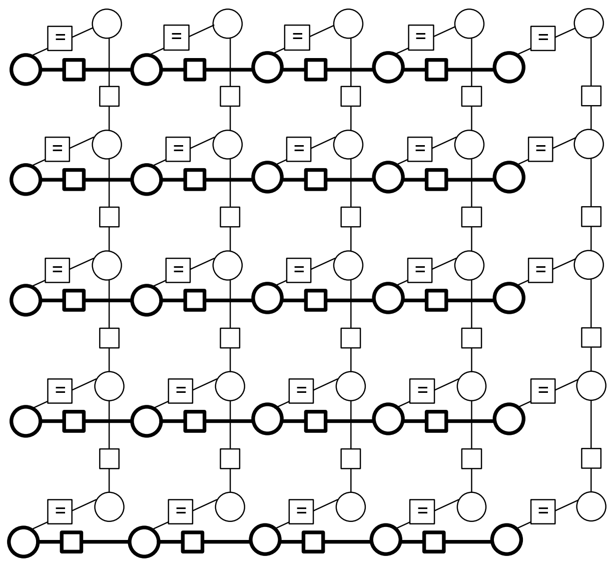

Figure 3: Complete factor graph for the concatenated HMM-AR model (here $H=5$, $W=5$).

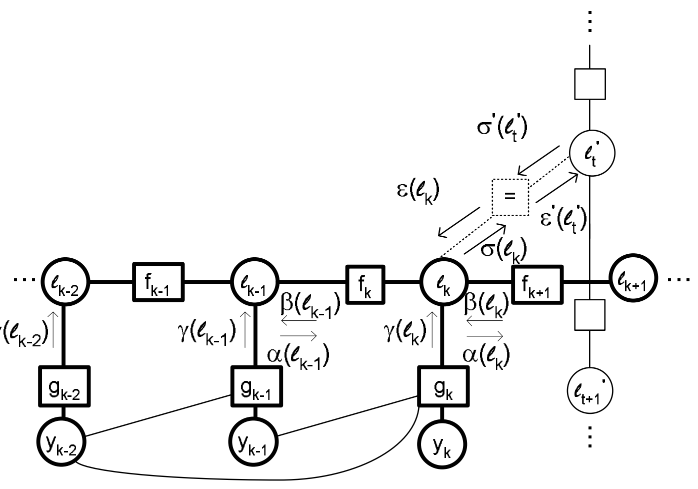

Figure 4: Messages exchanged on a portion of the complete factor graph.

one could just as well draw a small portion of the graph for the vertical 1D HMM-AR, with exactly the same graph structure and therefore the same message-passing algorithm.

Let $\gamma(l_k)$ be the message sent by the local function node g_k to the variable node l_k, according to (1) we have

$$\gamma(l_k) = p(y_k|l_k, y_{k-1}, \ldots, y_{k-P})$$
$$= \frac{1}{\sqrt{2\pi v(l_k)}} \exp\left\{ -\frac{1}{2v(l_k)} \left[y_k - \sum_{p=1}^{P} a_p(l_k) y_{k-p} \right]^2 \right\}.$$

Denote by l'_t the texture label of the vertical HMM-AR related to l_k by an equality constraint and by $\sigma'(l'_t)$ the message sent by l'_t to the equality function node. Applying the sum-product rule to the equality constraint, the message sent by the equality function node to l_k is given by

$$\varepsilon(l_k) = \sum_{l'_t=1}^{M} \delta(l_k - l'_t)\sigma'(l'_t) = \sigma'(l_k), \tag{4}$$

where $\delta(.)$ represents the Dirac function. Conversely, let $\sigma(l_k)$ be the message sent by l_k to the equality function node, the message sent by the equality function node to l'_t is given by

$$\varepsilon'(l'_t) = \sum_{l_k=1}^{M} \delta(l_k - l'_t)\sigma(l_k) = \sigma(l'_t). \tag{5}$$

Let $\alpha(l_{k-1})$ be the message sent by l_{k-1} to the function node f_k, then the message sent by f_k to l_k has the form

$$\sum_{l_{k-1}=1}^{M} P(l_k|l_{k-1})\alpha(l_{k-1}).$$

Now, applying the sum-product rule at the variable node l_k, we get the recursion

$$\alpha(l_k) = \sum_{l_{k-1}=1}^{M} P(l_k|l_{k-1})\alpha(l_{k-1})\gamma(l_k)\varepsilon(l_k), \quad 1 \le l_k \le M. \tag{6}$$

Similarly, let $\beta(l_k)$ be the message sent by the function node f_{k+1} to l_k, we obtain the recursion

$$\beta(l_{k-1}) = \sum_{l_k=1}^{M} P(l_k|l_{k-1})\beta(l_k)\gamma(l_k)\varepsilon(l_k), \quad 1 \le l_{k-1} \le M. \tag{7}$$

We recognize that (6) and (7) are a modified version of the forward and backward recursion of the well-known Baum-Welch algorithm [5], with $\varepsilon(l_k)$ as an additional multiplicative term.

Finally, the message sent by l_k to the equality constraint node under the sum-product rule is given by

$$\sigma(l_k) = \alpha(l_k)\beta(l_k)/\varepsilon(l_k), \quad 1 \le l_k \le M.$$

In fact, as suggested in [10], the message sent by l_k to the equality constraint node should be slightly modified to

$$\sigma(l_k) = w\alpha(l_k)\beta(l_k)/\varepsilon(l_k), \quad 1 \le l_k \le M, \tag{8}$$

where $0 \le w \le 1$ is a weight factor left as a design parameter, which increases with the number of iterations.

Due to the presence of many cycles in the proposed factor graph illustrated by Fig. 3, instead of stopping the message computations once all the nodes have been visited, the messages are recomputed iteratively until a stopping criterion is reached [7].

The proposed inference procedure is summarized in Table I.

MaxIt, the total number of allowed iterations, is the number of iterations needed so that a consensus is found between the texture labels computed by the horizontal and vertical pass. The final estimate of the texture labels is given by the maximum posterior mode (MPM) decision rule as [7]

$$\hat{l}_k = \arg \max_{l_k \in \{1,\ldots,M\}} \alpha(l_k)\beta(l_k), \quad 1 \le k \le WH, \tag{9}$$

where the $\alpha's$ and $\beta's$ are the forward and backward messages computed during the last vertical pass.

1. Initialize all α, β, ε messages to 1 (no prior information) and the set of parameters λ for the horizontal and vertical 1D HMM-AR. Set the iteration counter to $it = 1$.
2. Horizontal pass: Compute the forward recursion (6), the backward recursion (7) and (8) for each line of the image. Pass the updated ε messages (5) to the columns. Update λ for the horizontal 1D HMM-AR.
3. Vertical pass: Compute the forward recursion (6), the backward recursion (7) and (8) for each column of the image. Pass the updated ε messages (4) to the lines. Update λ for the vertical 1D HMM-AR.
4. While $it < MaxIt$, increment it and return to step 2)

Table 1: Turbo segmentation algorithm

3.2 Parameter estimation

Since our aim is to propose a semi-supervised segmentation algorithm, after each horizontal or vertical pass of the inference algorithm proposed in Table I, the set of parameters λ of the corresponding 1D HMM-AR must be updated. We use the EM approach developed in [6] to achieve this goal.

The 1D HMM-AR state transition matrix is re-estimated according to

$$p_{m,n} = \frac{\sum_{k=1}^{WH-1} \zeta_k(m,n)}{\sum_{k=1}^{WH-1} \sum_{n=1}^{M} \zeta_k(m,n)}, \quad 1 \le m,n \le M. \tag{10}$$

where $\zeta_k(m,n) = 0$ in case l_k is the last label of a line/column and l_{k+1} is the first label of the next line/column, otherwise

$$\zeta_k(m,n) = \frac{p_{m,n}\alpha(l_k = m)\beta(l_{k+1} = n)\gamma(l_{k+1} = n)\varepsilon(l_{k+1} = n)}{\sum_{m=1}^{M}\sum_{n=1}^{M} p_{m,n}\alpha(l_k = m)\beta(l_{k+1} = n)\gamma(l_{k+1} = n)\varepsilon(l_{k+1} = n)}, \tag{11}$$

where it is understood that is the old value of $p_{m,n}$ is used in (11). The expression of the quantities α, β, γ, ε is given in Sec. 3.1.

The AR parameters corresponding to the n-th texture class are re-estimated according to

$$\begin{cases} \mathbf{B}(n) = \left[\sum_{k=1}^{WH} \eta_k(n)\mathbf{z}_k\mathbf{z}_k^T\right]^{-1} \left[\sum_{k=1}^{WH} \eta_k(n) y_k \mathbf{z}_k\right] \\ v(n) = \dfrac{\sum_{k=1}^{WH} \eta_k(n)\left(y_k - \mathbf{z}_k^T\mathbf{B}(n)\right)^2}{\sum_{k=1}^{WH}\eta_k(n)}, \end{cases} \tag{12}$$

where

$$\begin{cases} \mathbf{z}_k = [y_{k-1}, y_{k-2}, \ldots, y_{k-P}]^T \\ \eta_k(m) = \dfrac{\alpha(l_k = m)\beta(l_k = m)}{\sum_{m=1}^{M}\alpha(l_k = m)\beta(l_k = m)}, \quad 1 \le m \le M. \end{cases}$$

In order to avoid convergence problems, the transition probabilities are constraint to be greater than a threshold, say $p_{m,n} \ge 10^{-2}$.

3.3 Parameter initialization

The initialization procedure for the AR parameters corresponding to each texture class is inspired from the coarse segmentation algorithm introduced in [2]. The lattice S is partitioned into non-overlapping rectangular regions. For the purpose of parameter initialization, we assume that each region contains only a single texture. Let R denote the lattice corresponding to such a region and R_I the interior subset of R [2]. After horizontal raster scanning of R, a 1D signal $\{y_k(R)\}$ corresponding to the pixel intensities is obtained, so that each pixel belonging to R_I can be modeled by an horizontal AR model

$$y_k(R) = \mathbf{z}_k^T\mathbf{B}_H + n_k,$$

where $\mathbf{z}_k(R) = [y_{k-1}(R), y_{k-2}(R), \ldots, y_{k-P}(R)]^T$, $\mathbf{B}_H$ is the column vector of AR coefficients and n_k is a zero mean Gaussian noise of variance v_H. A LS estimate of $\mathbf{B}_H$ and v_H is obtained as

$$\hat{\mathbf{B}}_H = \left[\sum_{R_I}\mathbf{z}_k(R)\mathbf{z}_k(R)^T\right]^{-1}\left[\sum_{R_I} y_k(R)\mathbf{z}_k(R)\right]$$

$$\hat{v}_H = \frac{1}{\mathrm{Card}(R_I)}\sum_{R_I}\left(y_k(R) - \mathbf{z}_k(R)^T\hat{\mathbf{B}}_H\right)^2.$$

Similarly, after vertical raster scanning of R, a 1D signal $\{y_k(R)\}$ corresponding to the pixel intensities is obtained, so that each pixel belonging to R_I can be modeled by an vertical AR model

$$y_k(R) = \mathbf{z}_k^T\mathbf{B}_V + n_k,$$

where $\mathbf{z}_k(R) = [y_{k-1}(R), y_{k-2}(R), \ldots, y_{k-P}(R)]^T$, $\mathbf{B}_V$ is the column vector of AR coefficients and n_k is a zero mean Gaussian noise of variance v_V. Also a LS estimate of $\mathbf{B}_V$ and v_V denoted by $\hat{\mathbf{B}}_V$ and $\hat{v}_V$ can be obtained. Each region R is then associated with the corresponding feature vector

$$\mathbf{F} = \left(\hat{\mathbf{B}}_H^T, \hat{\mathbf{B}}_V^T\right)^T.$$

Applying the *k-means* algorithm [3] to the feature vectors, a coarse estimate of the label field $\{\hat{l}_s^0, s \in S\}$ is obtained. This step implements a coarse segmentation where the number of classes M is assumed to be known. Assuming that the label field estimate $\{\hat{l}_s^0, s \in S\}$ is correct, a LS parameter estimation of $\mathbf{B}(n), v(n)$ for the n-th texture class, $n = 1, \ldots, M$, is recomputed for the horizontal and the vertical 1D HMM-AR. These values will be used as initial AR parameters.

The initialization of the state transition matrix for the horizontal and vertical 1D HMM-AR is obtained as follows. Raster scan the label field $\{\hat{l}_s^0, s \in S\}$ obtained from the coarse segmentation and set the initial value of $p_{m,n}$, $1 \le m,n \le M$ to

$$p_{m,n} = \frac{\text{number of transitions from label } m \text{ to label } n}{\text{number of transitions from label } m}. \tag{13}$$

If $p_{m,n} = 0$ during initialization, set $p_{m,n}$ to 10^{-2} and renormalize the resulting state transition matrix $\mathbf{P} = \{p_{m,n}\}$, so that the lines sum to one.

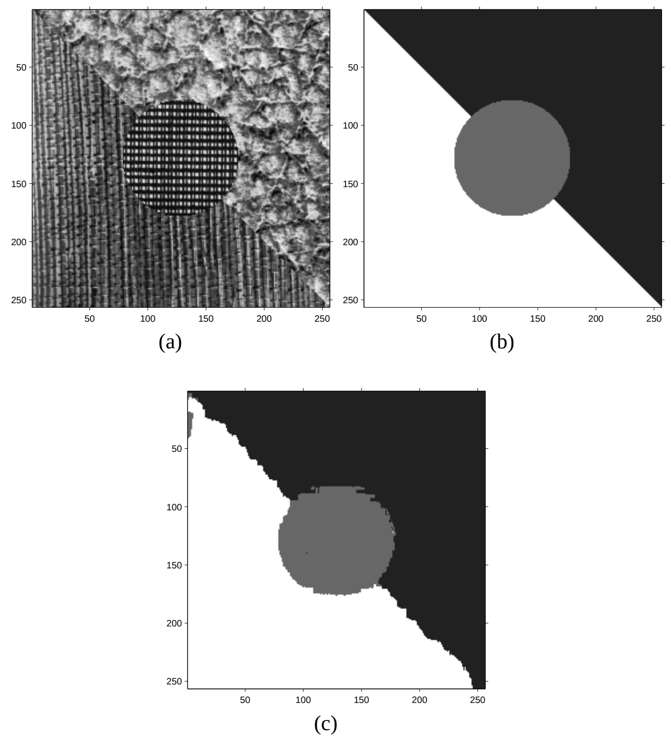
Figure 5: (a) 256×256 three-texture mosaic - (b) correct segmentation - (c) turbo segmentation with error rate of 1.72%.

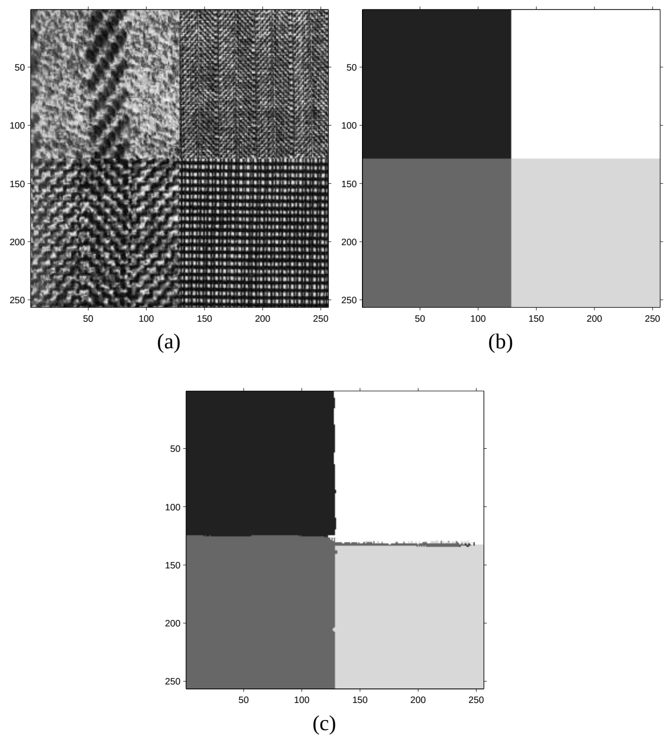
Figure 6: (a) 256×256 four-texture mosaic - (b) correct segmentation - (c) turbo segmentation with error rate of 1.67%.

4. EXPERIMENTAL RESULTS

Fig. 5a and 6a are texture mosaics taken from the Brodatz album [11]. Fig. 5c and 6c show the turbo-segmentation results when the order of the 1D HMM-AR models is $P = 5$ and the maximum number of iterations is set to $MaxIt = 5$. The evolution of the weight factors [10] (affecting the messages sent by the lines to the columns and the messages sent by the columns to the lines) with the iteration number was optimized experimentally to

$$w = [0.1, 0.1, 0.1, 0.1, 1.0].$$

The parameter initialization procedure described in Sec. 3.3 partitions the image into non-overlapping square regions of size 32×32 pixels. The obtained classification error rates are very good. Other numerical experiments, although not shown due to lack of space, demonstrated that turbo segmentation generally converges within a few iterations and can outperform existing Bayesian GMRF-based segmentation.

REFERENCES

[1] M. Tuceryan and A.K. Jain, *Texture Analysis* in *The handbook of pattern recognition and computer vision*, second ed.: World Scientific Publishing Co., 1998.

[2] 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.

[3] A.K. Jain and R.C. Dubes, *Algorithms for clustering data*, Englewood Cliffs, New Jersey: Prentice Hall, 1988.

[4] S. Geman and D. Geman, ”Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images,” IEEE Trans. Pattern Anal. Mach. Intell., vol.6, no.6, pp. 721-741, Nov 1984.

[5] L.E. Baum, T. Petrie, G. Soules and N. Weiss, ”A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains,” Ann. Math. Statist. , vol.41, no.1, pp. 164-171, 1970.

[6] J.D. Hamilton, ”Analysis of time series subject to changes in regime,” J. of Econometrics. , vol.45, pp. 39-70, 1990.

[7] F.R. Kschischang, B.J. Frey and H.-A. Loeliger, , ”Factor graph and the sum-product algorithm ,” IEEE Trans. Information Theory, vol.47, no.2, pp. 498-519, Feb 2001.

[8] C. Berrou, A. Glavieux and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: turbo codes,” *Proc. IEEE Int. Conf. Comm.,* pp. 1064–1070, Geneva, Switzerland, May 1993.

[9] A.P. Dempster, N.M. Laird and D.B. Rubin, ”Maximum likelihood from incomplete data via the EM algorithm,” J. Roy. Statist. Soc., Ser. B, vol.39, no.1, pp. 1-38, 1977.

[10] R.M. Pyndiah, ”Near optimum decoding of product codes: block turbo codes,” IEEE Trans. Comm., vol. 46, no. 8, pp. 1003-1010, Aug. 1998.

[11] P. Brodatz, *Textures: a photographic album for artists and designers*, New York: Dover Publications, 1966.