Textured Image Synthesis and Segmentation via Neural Network Probabilistic Modeling*

Jenq-Neng Hwang, Eric Tsung-Yen Chen

Information Processing Laboratory
Department of Electrical Engineering, FT-10
University of Washington
Seattle, WA 98195

ABSTRACT

It has been shown that a trained back-propagation neural network (BPNN) classifier with Kullback-Leibler criterion produces outputs which can be interpreted as estimates of Bayesian *a posteriori* probabilities. Based on this interpretation, we propose a back-propagation neural network (BPNN) approach for the estimation of the local conditional distributions of textured images, which are commonly represented by a Markov random field (MRF) formulation. The proposed BPNN approach overcomes many of the difficulties encountered in using MRF formulation. In particular our approach does not require the trial-and-error selection of clique functions or the subsequent laborious and unreliable estimation of clique parameters. Simulations show that the images synthesized using BPNN modeling produced desired artificial/real textures more consistently than MRF based methods. Application of the proposed BPNN approach to segmentation of artificial and real-world textures is also presented.

1 Introduction

In statistical image analysis we usually try to estimate the true scene (in image restoration) or texture labels (in image segmentation) based on the observed images. Markov random field (MRF) modeling has been widely used in both tasks [1, 10, 3, 7]. The MRF (and its variant Gaussian MRF) is a parametric representation of images and requires a careful choice of models (clique functions). In addition, the clique parameters which generate the local conditional probabilities require tedious and laborious estimation. More specifically, various estimation techniques have been proposed for clique parameter estimation: e.g., maximum likelihood method [10, 3, 1], maximum pseudo likelihood [2], linear least squares with histograming [4], and multi-start maximum likelihood [13]. These methods usually

*This research was partially supported through grants from the National Science Foundation under Grant No. ECS-9014243, and from NASA under Contract No. NAGW-1702.

generate unreliable or inconsistent solutions due to the requirements of either the formulation of a set of nonlinear equations that are difficult and cumbersome to solve [1, 10, 2], the least square formulation to a big set of linear equations based on very ad-hoc clique functions [4], or a nonlinear optimization problem subject to hundreds of thousands of inequality constraints [3, 13]. Moreover, most real-world textured images may possess approximately local neighboring pixel interactions but not necessarily follow the assumptions of MRF, where the global distribution of the image is constituted from the local distribution in an exponential form based on the linearly weighted *clique* potentials (i.e., Gibbs distributed). The method proposed here is a much relaxed probabilistic model based on the nonparametric representation of the local pixel interactions using feedforward back-propagation neural networks (BPNNs). We believe our BPNN model is a parsimonious representation whose local interactions are consistent with a much higher order MRF. More specifically, the simple linear weighting interaction of cliques in the Gibbs distribution for a very large order neighborhood may well result in local nonlinear interactions that can be approximated by a BPNN model using only the pixels in a smaller order neighborhood [9].

In the next section, we will investigate the Bayes probability approximation capability of a BPNN network. Section 3 will cover the MRF model and our proposed BPNN probabilistic model. In Section 4 we compare the performance of the proposed BPNN method with existing MRF methods in synthesizing artificial and real-world textures. Application of the proposed BPNN model to segmentation of artificial and real-world textures is illustrated in Section 5 and followed by concluding remarks in Section 6.

2 Bayes Probability Estimation of BPNNs

In a classification application, it is normally assumed that the input vector, $\mathbf{z} \in \mathbb{R}^n$, belongs to one of M classes, g_m, $1 \leq m \leq M$. The main objective of a classification task is to decide to which of the M classes the vector $\mathbf{z}$ belongs. The decision can be made based on some forms of deterministic discriminant function, e.g., the Euclidean distance measure. A more general decision rule is based on the probabilistic decision, such as the *maximum a posteriori* (MAP) approach which guarantees the minimum classification error. In a MAP approach, for each of the classes one requires to estimate the posterior probability, $P(g_m|\mathbf{z})$, which is usually computed via the Bayes' rule:

$$P(g_m|\mathbf{z}) = \frac{P(\mathbf{z}|g_m)P(g_m)}{P(\mathbf{z})} \propto P(\mathbf{z}|g_m)P(g_m) \tag{1}$$

where $P(\mathbf{z}|g_m)$ is the conditional distribution (also known as likelihood) and $P(g_m)$ is the *a priori* probability of the class g_m.

Since $P(g_m)$ is relatively easier to compute, most conventional pattern recognition literatures have been focusing on the research of estimating the likelihood $P(\mathbf{z}|g_m)$. On the other hand, when a BPNN is used for this classification task, there is usually an input layer of n neurons corresponding to the n-dimensional input vector $\mathbf{z}$, one or two layers of "appropriately chosen" hidden neurons, and one output layer of M neurons with each one representative of one of the M different classes. It has been shown that the output activations $\mathbf{y}$ of a BPNN trained by the standard back-propagation learning, which minimizes the

mean squared error (MSE) between the actual outputs $\mathbf{y}$ and the desired binary targets $\mathbf{t}$, indeed provide a least squares estimate of the posterior probabilities $\{P(g_m|\mathbf{z}),\ m = 1, ..., M\}$ [11]. Therefore neural network classifiers bypass the stage of estimating the likelihood and directly estimate the posterior probability in a classification task.

2.1 Kullback-Leibler Training Criterion

Unfortunately, the MSE criterion tends to give better estimates at large values than at small values [11]. In case of a noisy input vector $\mathbf{z}$, there is no clear winner by simply judging from the output activation values of the neural network, it then becomes very important to consider all the posterior probabilities generated by the neural network to make the final decision. The performance is usually severely degraded by the deficiency created by the MSE criterion. One criterion called Kullback-Leibler (KL) criterion which overcomes this deficiency and provides a better probability estimate can thus be adopted. The KL criterion, derived from information theory, suggests the *relative entropy* E_{KL} to be a better measure of the difference between the desired probabilities $\{t_m = P(g_m|\mathbf{z})\}$ and the actual estimated probabilities $\{y_m = \hat{P}(g_m|\mathbf{z})\}$ determined by the activation value of output neurons [12]. More specifically, when the desired outputs are binary valued and given N training vectors $\{\mathbf{z}^{(n)},\ n = 1, \ldots, N\}$, E_{KL} can be approximated by:

$$E_{KL} = -\sum_{m=1}^{M}\sum_{n=1}^{N}\left[t_m^{(n)} \log y_m^{(n)} + (1 - t_m^{(n)}) \log (1 - y_m^{(n)})\right]. \tag{2}$$

where $y_m^{(n)} = y_m^{(n)}(\mathbf{z}^{(n)}, \mathbf{W})$ is a function of inputs $\mathbf{z}^{(n)}$ and all the BPNN interconnection weights $\mathbf{W}$. The cross-entropy cost function differs from MSE in that it weights errors more heavily when actual outputs are near zero and one.

2.2 Imposing the Constraint in Ensuring Valid Probabilities

In order to ensure the BPNN's outputs to be valid probabilities (i.e., their values should be nonnegative and sum to one), additional care is required. For example, the positivity is guaranteed by the use of exponential activation functions in the output layer. However, to make network outputs sum to one without further normalization, an additional (fixed weights) layer of neuron was added on top of the output layer to impose this constraint [11]. For example, in a standard 2-layer (one hidden layer and one output layer, the input is not counted as a layer) BPNN with sigmoid activations, the activation function in the original output layer is replaced by a exponential function (instead of sigmoids). The additional layer (with the same number of units as the output layer, i.e., M units) generates the actual outputs y_m based on the following formulation:

$$y_m = \frac{exp(u_m)}{\sum_{k=1}^{M} exp(u_k)} \tag{3}$$

where u_k is the weighted sum from all hidden outputs to the k-th output unit. Although the introduction of the additional layer complicates the learning, the chain-rule involved in the back-propagation learning can still be worked out through the fixed-weight layer.

2.3 Posterior Probability Approximation Capabilities

To examine the posterior probability approximation capabilities of the proposed BPNN trained with Kullback-Leibler criterion, we conducted a simple binary (two-class) classification task with data generated from 2-D mixture Gaussians (class 1 has two Gaussian-mixtures and class 2 has three) [9]:

$$P(z_1, z_2|g_1) = \frac{1}{2}[N_{z_1}(-0.2, 0.2)N_{z_2}(-0.2, 0.1) + N_{z_1}(-0.6, 0.01)N_{z_2}(-0.6, 0.6)] \quad (4)$$

$$P(z_1, z_2|g_2) = \frac{1}{3}[N_{z_1}(0.5, 0.2)N_{z_2}(0.6, 0.2) + N_{z_1}(0.0, 0.1)N_{z_2}(0.0, 0.6) + N_{z_1}(0.9, 0.2)N_{z_2}(0.5, 0.2)] \quad (5)$$

where $N_{z_i}(m, \sigma)$ denotes a Gaussian (normal) distribution in i-th dimension with mean m and standard deviation σ.

A subset of simulated data is shown in Figure 1, where "$*$" denotes class 1 and "$\circ$" denotes class 2. There are 6,000 random samples generated for each class for BPNN training. Another 10,000 uniform grid samples are used for testing classification accuracy. Table 1 shows the classification accuracy of the 10,000 testing data after training and the *percentage of variance explained* (PVE) [5] between the true (desired) posterior probability and the estimated posterior probability:

$$PVE = 100(1 - E/Var)\% \quad (6)$$

where $E = \frac{1}{N}\sum_n (P_n - \hat{P}_n)^2$ denotes the mean squared error between the true posterior probability $P_n(g_m|z^{(n)})$ and the estimated one $\hat{P}_n(g_m|z^{(n)})$ over N testing data, and $Var = \frac{1}{N}\sum_n (P_n - \bar{P}_n)^2$ denotes the sample variance of the true probability.

Figure 2 illustrates the perspective plots of the true and estimated posterior probabilities for the two classes under evaluation. It can be seen that a BPNN trained with Kullback-Leibler criterion can give quite satisfactory approximation of posterior probability in a classification task.

No. of Hidden Neurons	Classification Accuracy	PVE Measure
10 Neurons	86.12%	75.39%
20 Neurons	85.81%	68.25%
30 Neurons	85.83%	74.82%

Table 1: Classification accuracy of the 10,000 testing data and the PVE measures between the true Bayesian density and the estimated density.

3 Markov Random Field and BPNN Probabilistic Modeling

An MRF model, which specifies the *a priori* joint distribution $P(\mathbf{I})$ of an image (random field) $\mathbf{I} = \{x_{ij}\}$, induces the Bayes probability for the restoration or segmentation purposes and is determined by the local conditional distribution $\{P(x_{ij}|\eta_{ij})\}$ of all pixels $\{x_{ij}\}$ (where η_{ij} is the set of neighboring pixels surrounding x_{ij}). The Gibbs sampler [7], which makes optimal use of the univariate conditional distribution, is a simple but powerful technique for image synthesis and segmentation. In a Gibbs sampling procedure, the univariate component is successively updated according to its conditional distribution, and the convergence of this procedure towards a maximum configuration of the joint distribution is guaranteed under rather general conditions where Brooks expansion is valid [2].

3.1 Markov Random Field and Gibbs Distribution

In an MRF setting, an image $\mathbf{I}$ is treated as a discrete 2-D random field defined over $N_1 \times N_2$ rectangular lattice of points, $\mathbf{I} = \{x_{ij} |\ 1 \leq i \leq N_1,\ 1 \leq j \leq N_2\}$. A neighborhood system η defined on $\mathbf{I}$ is a collection of subsets of $\mathbf{I}$, $\eta = \{\eta_{ij} |\ x_{ij} \in \mathbf{I},\ \eta_{ij} \subseteq \mathbf{I}\}$, that satisfies the following two conditions [4]:

1. $x_{ij} \notin \eta_{ij}$, and
2. if $x_{mn} \in \eta_{ij}$, then $x_{ij} \in \eta_{mn}$ for any $x_{ij} \in \mathbf{I}$.

A neighborhood system can be extended to any order hierarchically (see Figure 3), e.g., the 1st-order neighborhood system η^1_{ij} contains the nearest four pixels surrounding x_{ij}, and the 2nd-order neighborhood system $\eta^2_{i,j}$ contains the nearest eight pixels surrounding x_{ij}, and so on.

A *clique* $\mathbf{c}$ associated with a specific pixel x_{ij} and its neighbors η_{ij} is a subset of $\mathbf{I}$ such that

1. $\mathbf{c}$ contains the single pixel x_{ij}, or
2. for $x_{ij} \neq x_{mn}$, $x_{ij} \in \mathbf{c}$ and $x_{mn} \in \mathbf{c}$ implies that $x_{ij} \in \eta_{mn}$.

The collection of all cliques associated with each specific pixel x_{ij} is denoted as C_{ij}. The collection of all cliques associated with all $\{x_{ij}\}$ is denoted as $\mathbf{C}$. Figures 4(a) and 4(b) show a 2nd-order neighborhood system η^2 and its associated 10 cliques.

A random field $\mathbf{I} = \{x_{ij}\}$ is *Gibbs distributed* with respect to η if its joint distribution is of the form

$$P(\mathbf{I}) = \frac{1}{Z} exp(-U_{\mathbf{C}}(\mathbf{I})) \tag{7}$$

where $U_{\mathbf{C}}(\mathbf{I}) = \sum_{\mathbf{c} \in \mathbf{C}} V_{\mathbf{c}}(\mathbf{I})$ is the *energy function*, $V_{\mathbf{c}}(\mathbf{I})$ is the *potential* associated with each individual clique $\mathbf{c}$, and Z is a normalization constant.

Note that the Gibbs distribution is a kind of exponential distribution which states that the joint distribution of $\mathbf{I}$ can be described by the local conditional distribution

$$P(x_{ij}|\eta_{ij}) \propto exp(-U_{\mathbf{C}_{ij}}(x_{ij},\eta_{ij})). \tag{8}$$

The local energy function (or clique function) $U_{\mathbf{C}_{ij}}(x_{ij},\eta_{ij})$ defined over cliques in the neighborhood system η_{ij} is usually based on an *ad-hoc* or *common* parametric form with each (weighting) parameter controlling each clique type, e.g., a widely used clique functional formulation for images of small number of grey levels is

$$U_{\mathbf{C}_{ij}}(x_{ij},\eta_{ij}) = \sum_{\mathbf{c}_k \in C_{ij}} V_{\mathbf{c}_k}(x_{ij},\eta_{ij}) \tag{9}$$

$$V_{\mathbf{c}_k}(x_{ij},\eta_{ij}) = \begin{cases} -a\beta_k, & all\ x_{mn}\ in\ \mathbf{c}_k\ are\ equal \\ a\beta_k, & otherwise \end{cases} \tag{10}$$

where a is an arbitrarily chosen scaling constant, and $\{\beta_k\}$ are the clique parameters.

Although this class of clique functions have been successfully used in modeling artificial textures of small number of grey levels or small order of neighborhood, they cannot be easily extended and applied to most real-world textures. Gaussian MRF has been proposed for overcoming some of these difficulties, more specifically

$$U_{\mathbf{C}_{ij}}(x_{ij},\eta_{ij}) = \frac{1}{2\sigma^2}\left[(x_{ij}-\mu) - \sum_k \beta_k(x_{mn}-\mu)\right]^2, \quad \forall x_{mn} \in \eta_{ij} \tag{11}$$

where μ is the average grey levels of the image, and for every $x_{mn} \in \eta_{ij}$ there is an associated clique parameter β_k which is symmetric with respect to the relative position of x_{mn} to x_{ij}.

Various estimation techniques have been proposed for solving the clique parameters. These methods usually generate unreliable or inconsistent solutions due to either the difficulty involved in solving a set of nonlinear equations [1, 2], in solving a big set of least square solution using an over-simplified histogramming approach [4], or in solving a nonlinear optimization problem subject to several hundred thousands of inequality constraints [3, 13]. Moreover, most real-world textured images may possess approximately local neighborhood pixel interactions but not necessarily follow the assumptions of MRF (i.e., Gibbs distributed), where the global joint distribution of the image is constituted from the neighborhood local conditional distribution in an exponential form based on the "linearly" weighted contribution from either the ad-hoc *clique* potentials (for images of small number of grey levels) or pairwise clique interaction (for images of large number of grey levels).

We are thus motivated to investigate a much relaxed probabilistic model based on the nonparametric representation of the local interactions of pixels using feedforward back-propagation neural networks (BPNNs).

3.2 The BPNN Probabilistic Modeling

In an MRF setting, each clique type associated with the assumed neighborhood system should be well defined. Since the full local conditional distributions determine the joint, we are interested in the distribution of a center pixel x_{ij} conditioned on its neighbors η_{ij}, i.e., $P(x_{ij}|\eta_{ij})$. Instead of using the complicated parametric form of MRF clique functions

in representing $P(x_{ij}|\eta_{ij})$, we can formulate this local conditional probability easily in a BPNN setting for classification problems as discussed in Section 2.

In our BPNN probabilistic modeling, the training is constructed by simply feeding the BPNN with neighborhood configuration η_{ij} of each pixel x_{ij} and assigning center value as target $t = g_m$ in a classification formulation (i.e., each grey level of x_{ij} is assigned as one class). More specifically, take for example of a 2nd-order neighborhood system (see Figure 5), the BPNN contains 8 inputs to represent 8 neighboring pixels, appropriately chosen number of hidden neurons, and M output neurons to represent M grey levels. The desired outputs are set so that only the m-th output neuron is "1" and the rest are "0" if the grey level value of the center pixel x_{ij} is equal to m. After trained by some pairs of training data obtained from existing textured images, the BPNN responds to a new neighborhood configuration inputs by providing the local conditional distribution (or the classifier's posterior probability) $\{P(x_{ij} = m|\eta_{ij}),\ m = 1, ..., M\}$ as outputs for the center site being assigned to all grey levels, i.e., the BPNN represents a model of underlying field which is conventionally modeled by Gibbs distribution in parametric form for $P(x_{ij}|\eta_{ij})$.

3.3 Gibbs Sampler

After an MRF model or a BPNN probabilistic model is established, the Gibbs sampling algorithm can be applied for image synthesis or segmentation. The Gibbs sampler [7] provides a Bayesian inference mechanism for applications like image analysis and other areas of multivariate statistics that previously suffered from computational prohibition [6].

The Gibbs sampler updating scheme proceeds as follows. Given an arbitrary starting set of values $\mathbf{S}^{(0)} = (\mathrm{s}_1^{(0)}, \cdots, \mathrm{s}_p^{(0)})$, we update a single component $\mathrm{s}_1^{(0)}$ to $\mathrm{s}_1^{(1)}$ according to the distribution $P(\mathrm{s}_1|\mathrm{s}_2^{(0)}, \cdots, \mathrm{s}_p^{(0)})$ in a stochastic manner (i.e., based on the resulting CDF value and a uniform distributed random number). The Gibbs sampler procedure then updates $\mathrm{s}_2^{(0)}$ to $\mathrm{s}_2^{(1)}$ according to the distribution $P(\mathrm{s}_2|\mathrm{s}_1^{(1)}, \mathrm{s}_3^{(0)}, \cdots, \mathrm{s}_p^{(0)})$, and so on up to $\mathrm{s}_p^{(1)}$ from $P(\mathrm{s}_p|\mathrm{s}_1^{(1)}, \cdots, \mathrm{s}_{p-1}^{(1)})$ to complete one iteration. Under certain conditions [7], $\mathbf{S}^{(t)}$ can be regarded as a simulated observation from (near) maximum(s) of joint distribution $P(\mathbf{S})$ as $t \to \infty$.

4 Comparative Simulation Studies in Textured Image Synthesis

To support the claims of unreliable and inconsistent estimation of clique parameters based on the existing MRF approaches and to demonstrate the usefulness of our proposed BPNN probabilistic modeling, application of the existing MRF approaches and our BPNN method to synthesizing artificial and real-world textures is presented. It is our belief that the textured image synthesis task is a much more difficult job than the textured image segmentation, which will be presented in the later section.

4.1 Synthesis of Artificial Textures

Figures 6(a) and 6(b) show two different types of artificial textures of 4 grey levels (these two textured images were used in [4]). These two 128×128 artificial textures of a 2nd-order

	A	**E4**	**E9**
β_1	1.0	0.64	0.71
β_2	1.0	0.80	0.87
β_3	-1.0	-0.47	1.58
β_4	1.0	1.27	1.01
β_5	0.0		-1.41
β_6	0.0		0.17
β_7	0.0		0.43
β_8	0.0		-1.62
β_9	0.0		0.58

(a)

	A	**E4**	**E9**
β_1	2.0	1.31	0.98
β_2	2.0	1.29	0.93
β_3	-1.0	-0.57	-0.70
β_4	-1.0	-0.67	-0.68
β_5	0.0		0.00
β_6	0.0		-0.16
β_7	0.0		0.00
β_8	0.0		0.06
β_9	0.0		0.61

(b)

Table 2: (a) True and estimated clique parameters for texture type 1. (b) True and estimated clique parameters for texture type 2.

MRF were generated by Gibbs sampling technique [7] based on 4 nonzero clique parameters given in column **A** of Table 2. In fact, there should be 9 clique parameters for 9 nontrivial cliques (i.e., excluding the single pixel clique) as evidenced by Figure 4(b).

Linear Least Squares MRF Approach: Based on the original textured images shown in Figures 6(a) and 6(b), the clique parameters were estimated using the popular linear least squares MRF method [4]. The definition of clique functions follow the formulation of Eq. (10) with a scaling constant $a = 0.5$. All the 128×128 pixels of each image were used as training data. In this paper, all the images assume a wrap around configuration. To have a better estimate, we first assumed that the knowledge of 5-out-of-9 parameters being zero was available. The estimated clique parameters are shown in Column **E4** of Table 2. The resynthesized textured images based on this estimate are shown in Figures 7(a) and 7(b).

Column **E9** in Table 2 denotes the linear least squares estimates for all the 9 clique parameters (assuming the knowledge of some parameters being zero is not available) based on the same clique functions. The resynthesized textured images based on this estimate are also shown in Figures 8(a) and 8(b).

It can thus be concluded that the estimation results based on existing MRF approaches for artificial textures are quite unreliable and very sensitive to the number of parameters to be estimated and the selection of alternative clique functions (e.g., auto-logistic, auto-binomial, auto-poisson, etc [1]).

BPNN Probabilistic Modeling Approach: We then applied our BPNN probabilistic modeling for synthesizing these two artificial textures. The BPNN contains 8 inputs to represent 8 neighboring pixels, 10 hidden neurons, and 4 output neurons to represent 4 grey levels. All the 128×128 pixels are used as training data. 1000 iterations of training is used. After training, the BPNN responds to the neighborhood configuration inputs by providing

the local distribution $\{P(x_{ij} = m|\eta_{ij})\}$ as outputs for the center site being assigned to any of the 4 grey levels.

The resynthesized images using BPNN modeling and the Gibbs sampling procedure are shown in Figures 9(a) and 9(b). Two BPNNs are trained (one for each texture type) and the training accuracies are 90% and 89% for texture type 1 and 2, respectively. Compared with those resynthesized images using linear least squares MRF approaches given in Figures 7 and 8, we found our BPNN approach gives desired textures more consistently. Moreover, we also noticed that the Gibbs sampling used in iteratively updating and synthesizing the textured images required much less number of iterations (around 10) in using the local conditional distribution generated by BPNN model than the number in usig those generated by MRF model (around 100 iterations). This is another indication of more effective approximation of the local conditional distribution in representing the random field.

It can be argued that the proposed BPNN modeling approach requires much more parameters (interconnection weights of the BPNN) than the number of clique parameters. Even though this challenge is valid, there is no way to increase the number of clique parameters for a better performance without increasing the order of the neighborhood system under the quite restricted MRF formulation. More importantly, when the order of neighborhood system increases in an MRF model, the number of clique parameters increases exponentially and the parameter estimation performance degrades also rapidly. On the other hand, the BPNN probabilistic modeling increase its number of interconnection weights linearly (i.e., the input dimension increases) with slight performance degradation observed from our experience.

4.2 Synthesis of Real-World Textures

To further illustrate the consistency of the proposed BPNN for modeling textures, simulations were conducted for synthesizing real-world textures. Twelve real-world textured images of size 128×128 obtained from Brodatz database were used. To save the computer simulation time, these textured images were down-quantized from 256 grey levels to 16 grey levels without apparent visualization degradation. Figure 10 shows three typical 16-grey-level Brodatz textures (brick wall, wood grain, and beach sand).

Multi-Start Gaussian MRF Approach: We adopted a 5×5 neighborhood system in our Gaussian MRF formulation of the Brodatz textures. Twelve distinct clique parameters associated with each texture were then estimated using the multi-start maximum likelihood estimation algorithm [13]. The definition of clique functions follow the formulation of Eq. (11). All the 128×128 pixels of each image were used as training data. We exactly used the same values of control parameters (d, c, T) reported in [13] and a complete checking of constraint satisfication was used to improve the performance. The resynthesized Brodatz textured images based on these estimates are shown in Figures 11.

BPNN Probabilistic Modeling Approach : A BPNN of 24-30-16 configuration (24 inputs, 30 hiddens, and 16 outputs) was used to represent each type of Brodatz textures of 16 grey levels. The BPNN method was applied in a straightforward manner as the BPNN for the representation of artificial textures. The estimated and resynthesized textures by

BPNN modeling and Gibbs sampling were able to give quite desirable results as shown in Figure 12. In addition to the slight performance improvement of BPNN modeling over the Gaussian MRF approach indicated by subjective visual judgement of performance, we again observed the much faster convergence (one order of magnitude faster) in Gibbs sampling synthesis when using the BPNN approach.

5 Application to Texture Segmentation

The proposed BPNN probabilistic modeling, when incorporated with Gibbs sampling procedure, can be employed in a hierarchical manner for Bayesian textured image segmentation. In this application, the Gibbs sampler focuses on estimation of the global *maximum a posteriori* (MAP) configuration through a sequential and iterative updating of the labels of all the pixels. More specifically, given Q textures in an image, the goal is to assign each pixel x_{ij} a label $l_{ij} \in \mathbf{L}$ (where $l_{ij} = 1, \cdots, Q$) so that $P(\mathbf{L}|\mathbf{I})$ is maximized. Again, based on the Bayes rule (with x_{ij} and η_{ij} being held constant),

$$\begin{aligned} P(l_{ij} = q|x_{ij}, \eta_{ij}) &= \frac{P(x_{ij}, \eta_{ij}|l_{ij} = q)P(l_{ij} = q)}{P(x_{ij}, \eta_{ij})} \\ &\propto P(x_{ij}, \eta_{ij}|l_{ij} = q)P(l_{ij} = q) \\ &= P_q(x_{ij}, \eta_{ij})P(l_{ij} = q) \\ &= P_q(x_{ij}|\eta_{ij})P_q(\eta_{ij})P(l_{ij} = q) \\ &\propto P_q(x_{ij}|\eta_{ij})P(l_{ij} = q) \end{aligned} \tag{12}$$

where the prior probability $P(l_{ij} = q)$ is the high level spatial interaction specifying region formation that can be modeled by Gibbs distribution [4], and $P_q(x_{ij}, \eta_{ij}) = P(x_{ij}, \eta_{ij}|l_{ij} = q)$ specifies the low level texture characteristics of type q. Note that $P_q(x_{ij}, \eta_{ij})$ can be further determined by the local conditional distribution $P_q(x_{ij}|\eta_{ij})$ in a neighborhood interaction system.

More specifically, for each pixel x_{ij} we first calculate the prior probabilities $\{P(l_{ij} = q),\ q = 1, ..., Q\}$, to which texture type it belongs based on a low order Gibbs distribution. We then send the pixel values of its neighbors η_{ij} to the Q trained BPNNs to get the local conditional distributions $\{P_q(x_{ij}|\eta_{ij}),\ \forall q\}$. Since the center site pixel value is fixed, a single local conditional probability (out of M probabilities) is obtained from each BPNN. Based on the product of these two probabilities as given in Eq. (12), the Gibbs sampling procedure can be applied to update its label l_{ij} in evolving to the most likely texture type. We then keep updating each pixel's label until all pixels are updated which completes one iteration. Several iterations may be needed before convergence. For noisy textured images, a noise process can be incorporated in this framework [2].

5.1 Segmentation of Artificial Textures

Given two 4-grey-level artificial textures (see Figure 13) synthesized respectively from two 2nd-order MRFs with their corresponding clique parameters given in Table 3, we are to segment a 128×128 combined textured image (see Figure 14(a)) that consists of these

	Texture 1	Texture 2
β_1	1.0	1.0
β_2	1.0	1.0
β_3	1.0	1.0
β_4	1.0	-1.0
β_5	0.0	0.0
β_6	0.0	0.0
β_7	0.0	0.0
β_8	0.0	0.0
β_9	0.0	0.0

Table 3: Two sets of clique parameters for synthesizing the textures of 2nd-order MRFs.

two textures. These two textures were first represented (modeled) by two BPNNs by the procedures described in Section 4.1. The segmentation mask is modeled by a 2nd-order Gibbs distribution as shown in Figure 14(b). The segmentation result after 150 iterations of the hierarchical Gibbs sampling is shown in Figure 14(c). The segmentation result can be further smoothed and improved by using a simple 3×3 median filter over the segmented label image (see Figure 14(d)).

5.2 Segmentation of Brodatz Textures

We also tried to segment a textured image (see Figure 15(a)), that consists of two 16-grey-level brodatz textures (the brick wall and the beach sand textures) as shown in Figures 10(a) and 10(b). These two textures have been well represented (modeled) by two BPNNs as described in Section 4.2 (also see Figure 12). The segmentation mask is artificially created by a hand-drawing shape as shown in Figure 15(b). The segmentation result after 150 iterations of the hierarchical Gibbs sampling is shown in Figure 15(c). The smoothed image using a 3×3 median filter is shown in Figure 15(d).

6 Concluding Remarks

We have proposed a BPNN approach for modeling the probabilistic local interaction of a random field. Our simulation results show that a BPNN trained with Kullback-Leibler criterion can provide a fairly good estimate of Bayesian *a posteriori* probabilities for image modeling. While conventional MRF modeling relies on difficult parameter estimation techniques, our method gives a simpler and more reliable solution which provides local conditional distribution required by Gibbs sampler. The proposed technique is verified by successful application to textured image synthesis and segmentation of both artificial and real-world textures.

Acknowledgments

The authors wish to thank Dr. Alan Lippman for his valuable and constructive suggestions, from which this paper has benefited significantly.

References

[1] J. Besag. Spatial interaction and the statistical analysis of lattice systems (with discussion). *J. R. Statist. Soc.* B(36):192-326, 1974.

[2] J. Besag. On the statistical analysis of dirty pictures. J. R. Statist. Soc. B. 48:259-302, 1986

[3] R. Chellappa and S. Chatterjee. Classification of textures using Gaussian Markov random fields. IEEE Trans. on ASSP, 33:959-963, 1985.

[4] H. Derin and H. Elliott. Modeling and segmentation of noisy and textured images using Gibbs random fields. IEEE Trans. on PAMI, 9(1):39-55, January 1987.

[5] J. H. Friedman, W. Stuetzle, A. Schroeder Projection Pursuit Density Estimation. Journal of the American Statistical Association, Vol 79, pp. 599-608, 1984.

[6] A. Gelfand, S. E. Hills, A. Racine-Poon, A. F. M. Smith. Illustration of Bayesian inference in normal data models using Gibbs Sampling. Journal of the American Statistical Association, 85(412):972-985, 1990.

[7] S. Geman, D. Geman. Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE trans. on PAMI*, 6(6):721-741, November 1984.

[8] U. Grenander Tutorial in pattern theory. Technical Report in Division of Applied Mathematics, Brown University, Providence, 1983.

[9] J. N. Hwang, Eric T. Y. Chen. Textured image segmentation via neural network probabilistic modeling. Int'l Conf. on Neural Networks, Vol. 3, pp. 1702-1707, San Francisco, CA, March 1993.

[10] R. Kashyap and R. Chellappa. Estimation and choice of neighbors in spatial interaction model of images. IEEE Trans. on Information Theory, 29:60-72, 1983.

[11] J. Makhoul. Pattern recognition properties of neural networks. IEEE-SP Workshop on Neural Networks for Signal Processing, pp. 173-186, Princeton NJ, 1991.

[12] B. S. Wittner, J. S. Denker Strategies for teaching layered networks classification tasks. In Neural Information Processing Systems. ed. D. Z. Anderson, ppi. 850-859, New York: American Institute of Physics. 1988

[13] C. S. Won and H. Derin. Maximum likelihood estimation of Gaussian Markov random field parameters. Int'l Conf. on ASSP, pp. 1040-1043, New York, 1988.

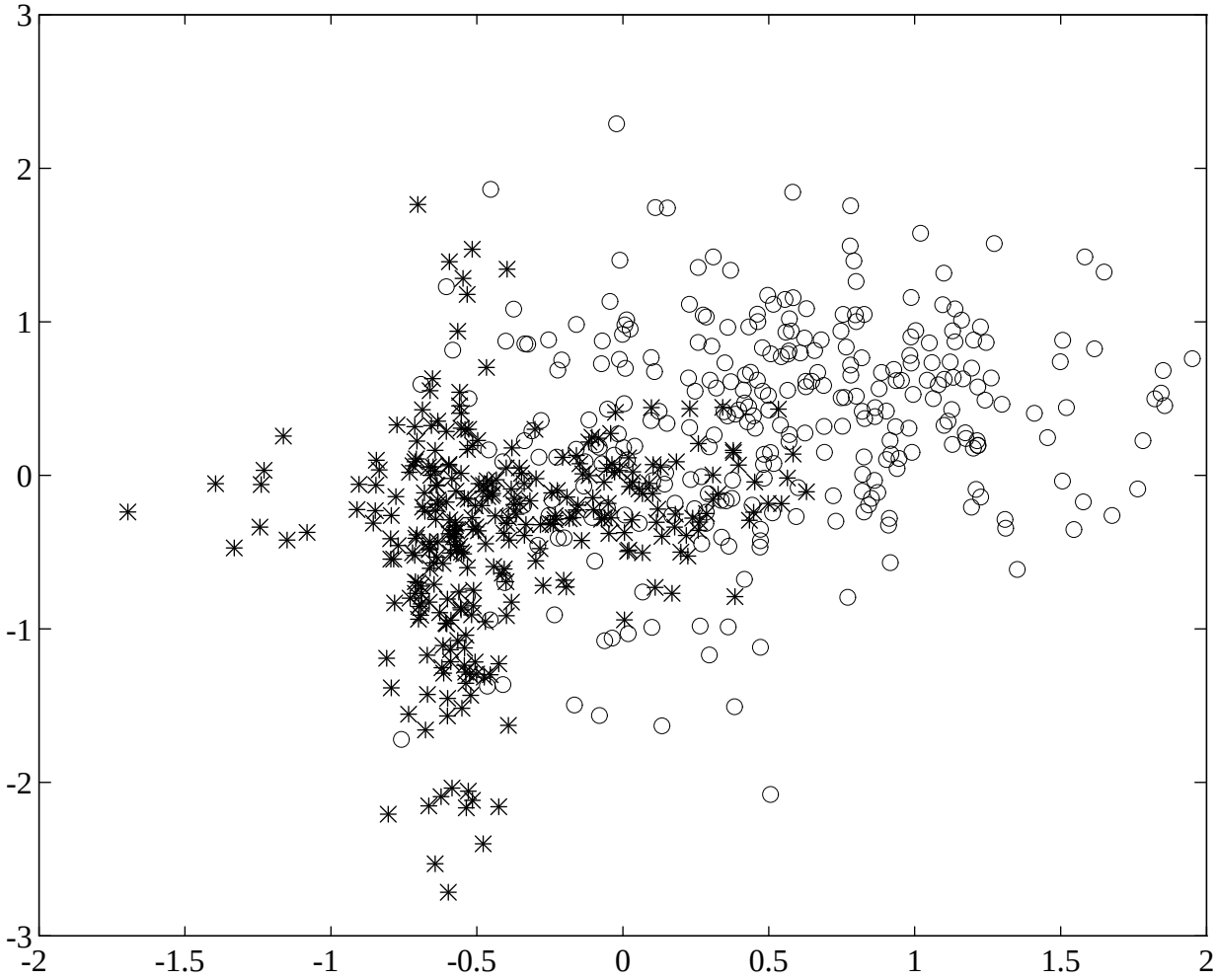
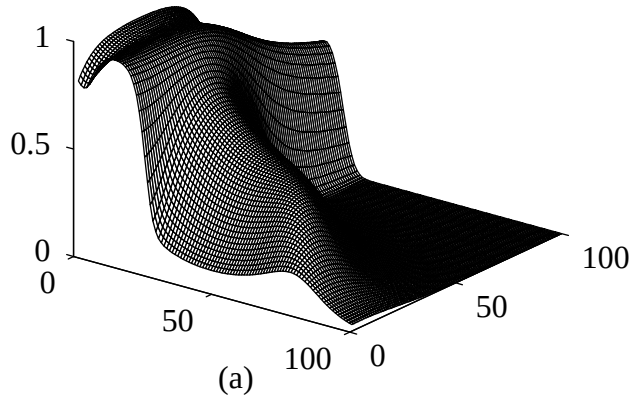


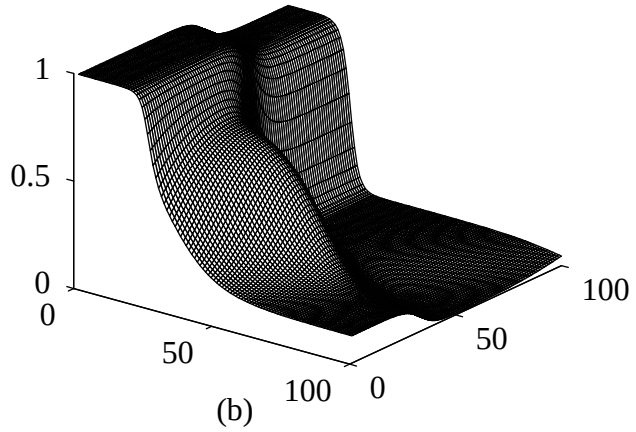
Figure 1: A subset of randomly generated training data of Gaussian mi xtures from two different classes.

Figure 2: (a) The true posterior probability of class 1.



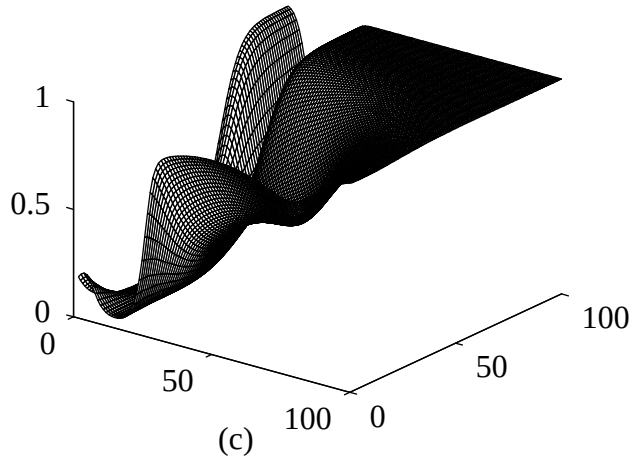
(a)

Figure 2: The estimated posterior probability of class 1.



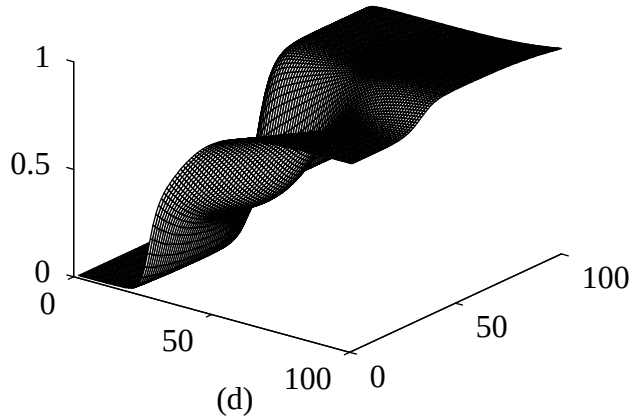
(b)

Figure 2: (c) The true posterior probability of class 2.



(c)

Figure 2: (d) The estimated posterior probability of class 2.



(d)

5	4	3	4	5
4	2	1	2	4
3	1	x	1	3
4	2	1	2	4
5	4	3	4	5

Fig 3 : A neighborhood system can be extended to any order hierarchically.

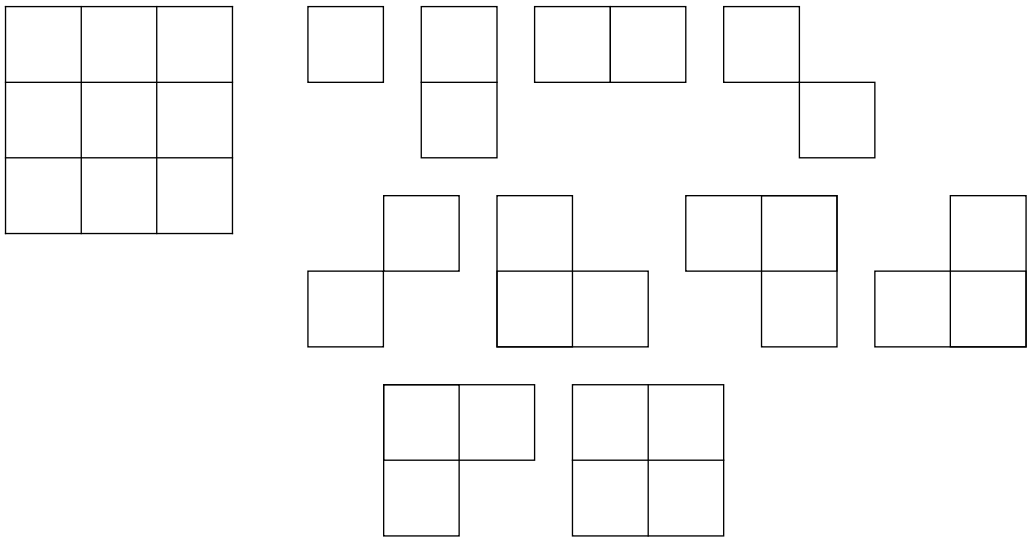


Fig 4: (a) The second-ordor neighborhood system $\eta 2$. (b) The associated cliques of $\eta 2$.

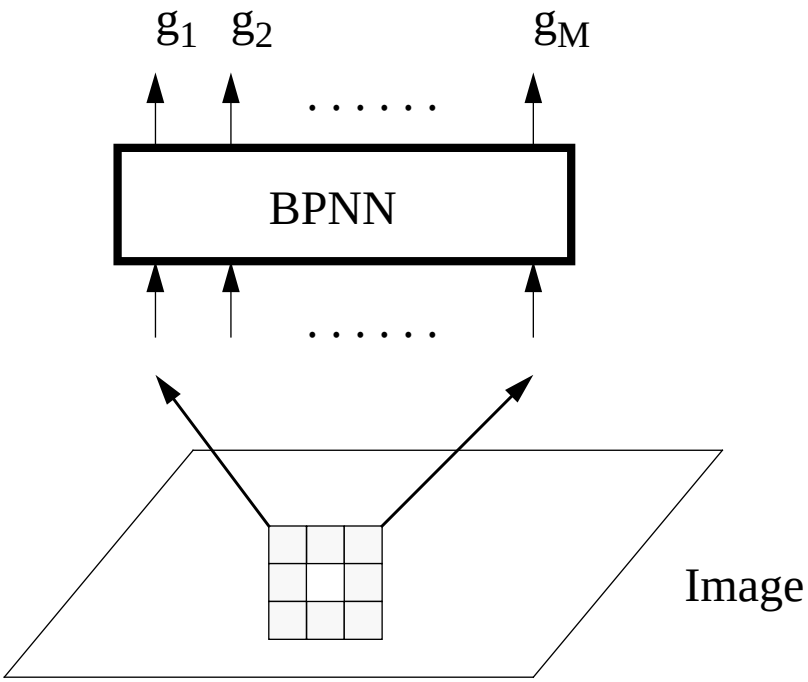
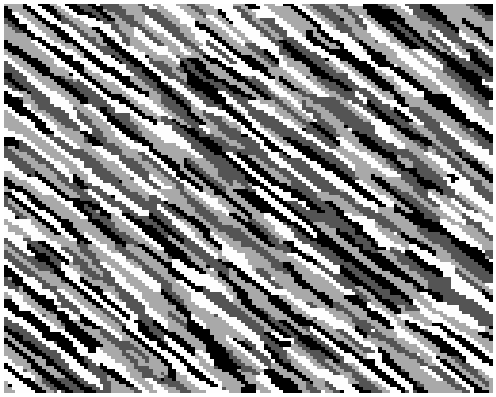


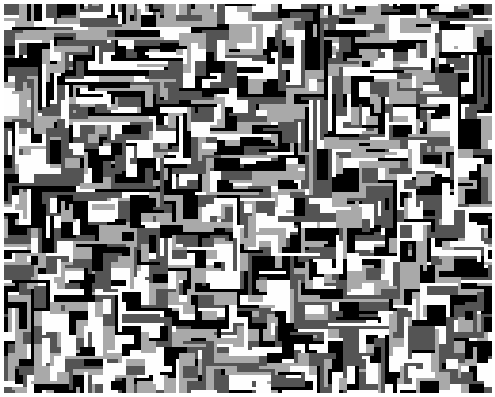
Fig 5: A BPNN is trained to provide local distribution of an image of M grey levels under a 2nd-order neighborhood system.

Fig 6: (a)



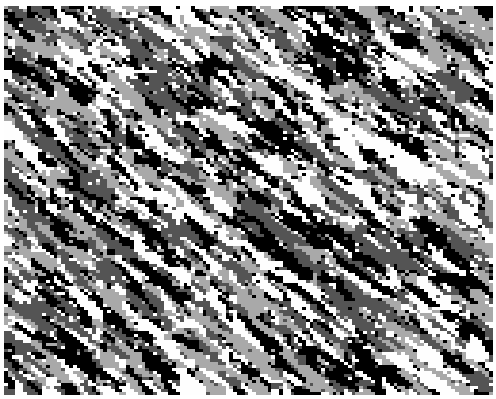
original artificial type 1

Fig 6: (b)



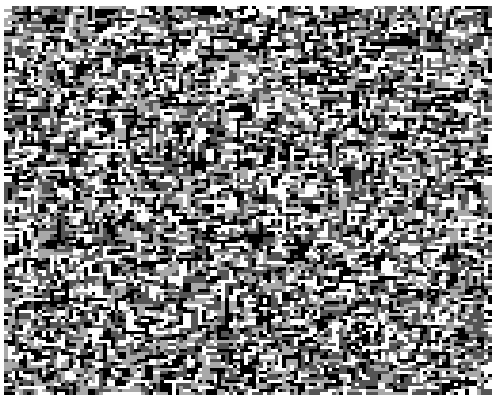
original artificial type 2

Fig 7: (a)



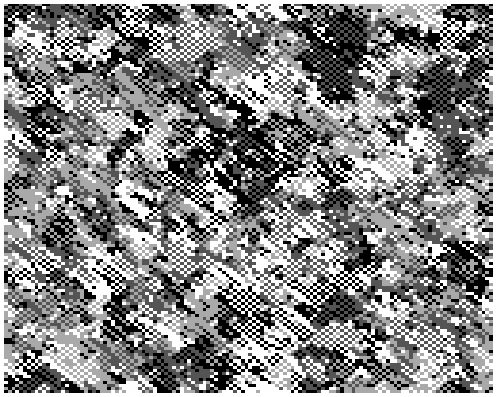
resynthesized : E4

Fig 7: (b)



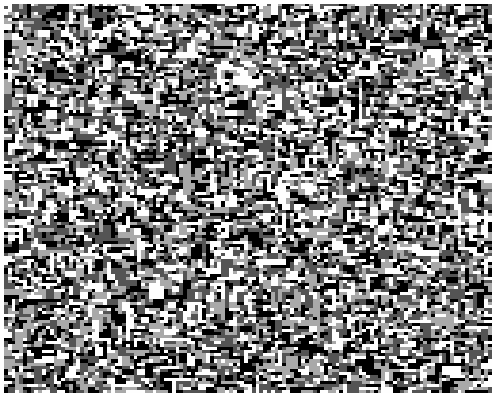
resynthesized : E4

Fig 8: (a)



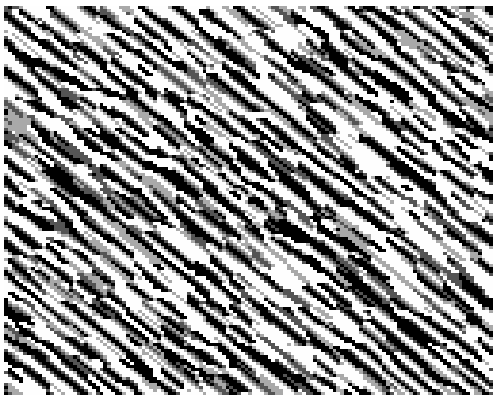
resynthesized : E9

Fig 8: (b)



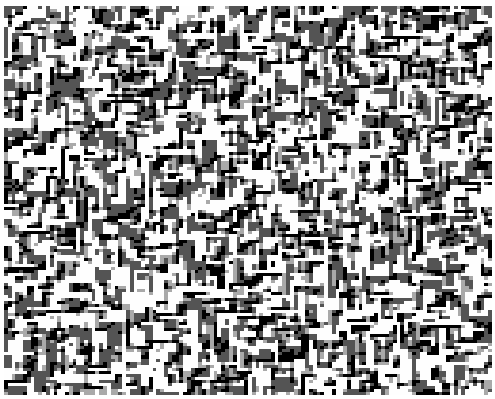
resynthesized : E9

Fig 9: (a)



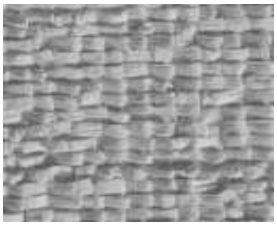
resynthesized : BPNN

Fig 9: (b)

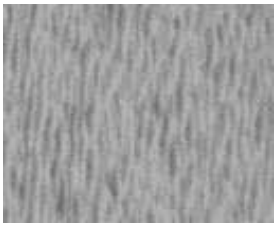


resynthesized : BPNN

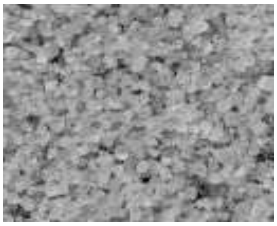
Fig 10: Original textures



(a) brick wall

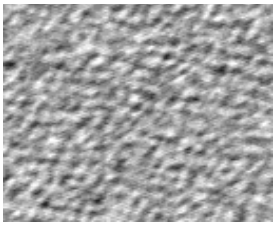


(b) wood grain

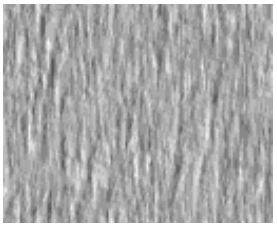


(c) beach sand

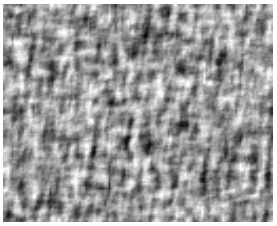
Fig 11: GMRF resynthesized



(a) brick wall

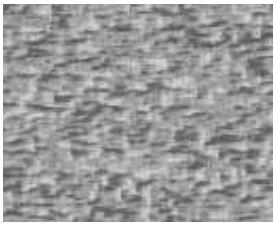


(b) wood grain

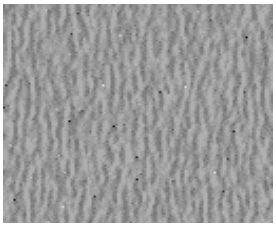


(c) beach sand

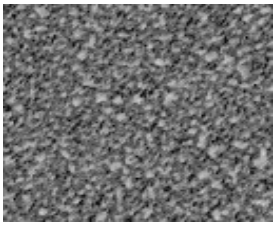
Fig 12: BPNN resynthesized



(a) brick wall



(b) wood grain



(c) beach sand

Fig 13: (a) artificial texture type 1

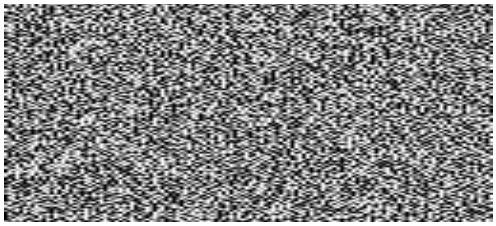


Fig 13: (b) artificial texture type 2

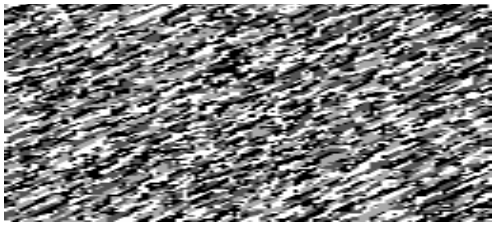
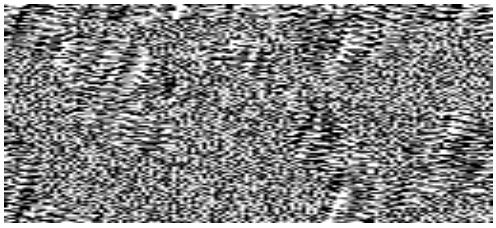


Fig 14: (a)



textured image

Fig 14: (b)

mask

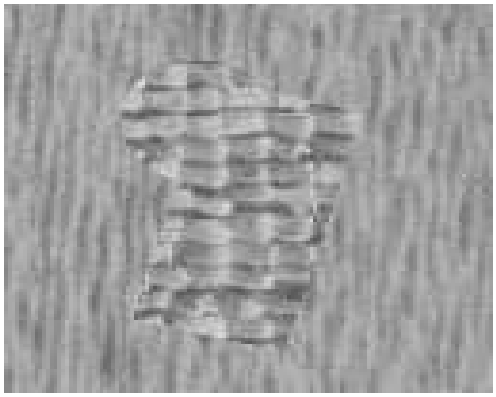
Fig 14: (c)

150 iterations

Fig 14: (d)

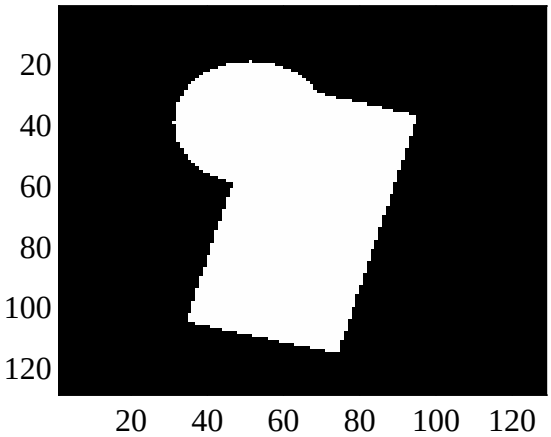
smoothed with 3x3 median filter

Fig 15: (a)



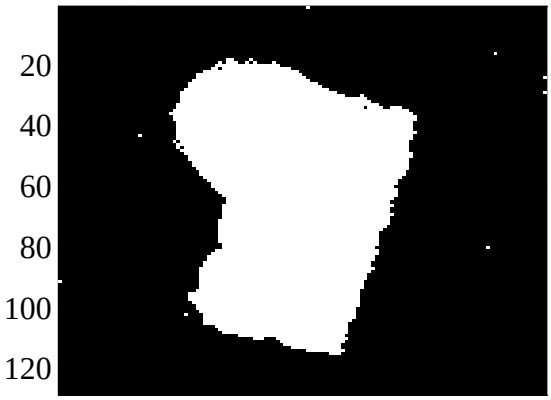
textured image with 2 16-gray-level Brodatz textures.

Fig 15: (b)



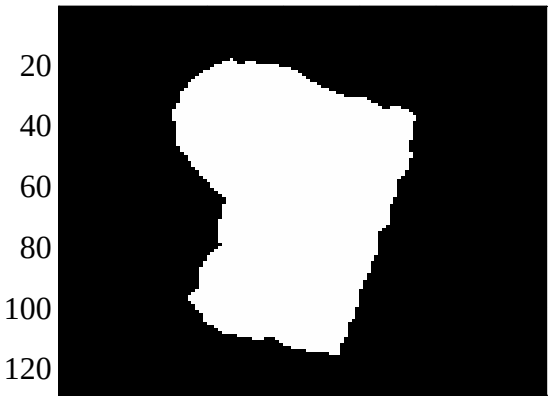
mask

Fig 15: (c)



150 iterations

Fig 15: (d)



smoothed iamge using 3x3 median filter