Partially ordered Markov models for texture synthesis and classification

by

Ashit Talukder

A Thesis Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE

Department: Electrical Engineering and Computer Engineering
Major: Electrical Engineering

Iowa State University
Ames, Iowa

1994

DEDICATION

This thesis is dedicated to my parents for their infinite patience and their constant support through all the times when I needed help, and to my brother, Ashim, for his guidance at the indecisive moments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Two-dimensional time series and random field models are among the most common classes of statistical image models. Textural patterns in images are important image descriptors used for classification of image data, and for synthesis of realistic looking images in computer simulations. In this thesis, the use of a new subclass of Markov random fields (MRFs), called *partially ordered Markov models* (POMMs), as models of texture is explored. A new fourth order POMM is created for texture analysis, and used for texture synthesis and classification. POMMs offer advantages over general MRFs, one of which is the POMMs capability to express exactly the joint distribution. This property, among others, is exploited in the efforts presented here. Texture classification is achieved by using the POMM model parameters as features and adopting a statistical MRF model for the region process. A template identification problem using morphology neural networks (MNN) is also described.

A method for generating samples from the POMM using an independent, identically distributed array of random variables is described that simulates textures in reasonable computation time. A wide variety of binary and gray scale textures are generated using the POMM, including homogenous and heterogeneous textures. The problem of parameter estimation is examined for noise-free image data and real, natural scenes, and a hypothesis test proposed to evaluate the accuracy of the estimates for the fourth order POMM. It is observed that the POMM model can perform well when fitting homogenous textures.

A supervised segmentation algorithm employing a *maximum a posteriori* (MAP) criterion using simulated annealing (SA) is proposed to segment textured image data assuming an underlying POMM.

Simulated annealing is used to train the train the weights of a morphology neural network (MNN), a nonlinear type of artificial neural network (ANN). The problem of morphology template identification using MNNs is addressed, and the application of SA to train MNNs for template identification is presented.

The POMM proposed in this thesis has potential applications in modeling of noisy images, image restoration, nonhomogeneous texture modeling, as well as in other nonimaging related fields.

PROGRAM OF STUDY COMMITTEE

Major Professor

J. Davidson

Committee Members

L. Udpa

N. Cressie

1. INTRODUCTION

Image processing is the branch of science that deals with the analysis and processing of images for the purposes of image enhancement, restoration, segmentation, and classification. Processing of digital images involves procedures that are usually expressed in algorithmic form. In most cases, the end result desired after the processing is the segmented image where each section or object in the picture is "recognized." Thus, the ultimate aim in a large number of image processing applications is to extract important features from image data, from which a description, interpretation, or understanding of the scene can be provided by the image processing system. These classified images are used in geographical mapping, tracking of earth's resources, stellar object recognition in images from deep space-probe missions, detection of tumors and other diseases in medical images, and automated target recognition in military warfare, to name a few applications.

Common techniques for image segmentation and classification include thresholding, filtering, and pattern recognition classification techniques utilizing structural and transform features. Regions in the image can also be described by their textural characteristics. One approach to texture classification assumes an underlying model for the regions in the image and then the image is classified based on the model. The work in this thesis is concerned with texture models. Stochastic models of texture assume the texture to have an underlying statistical distribution [1]. ARMA models, time series models, random mosaic models [2], and Markov models [3] are commonly used stochastic models.

Markov random field models (MRFs) have been used as probabilistic models of digital image textures where a textured region is viewed as a finite sample of a two-dimensional random process describable by its statistical parameters. MRFs can be considered to be multidimensional generalizations of Markov chains, and are defined in terms of conditional probabilities associated with spatial neighborhoods. In this thesis, a newly-developed statistical model called *partially ordered Markov models* (POMMs) [4], is used for texture analysis. POMMs are a generalization of the well-known

Markov mesh models (MMMs) [5]. The POMMs are based on the concept of partial orders. The feasibility of Markov models and POMMs as models of texture, as well as the theoretical underpinnings of partially ordered Markov models are discussed in the thesis.

Choice of the number of parameters in the POMM to model the images and a specific form of expression of the neighborhood conditional probabilities are crucial factors in the texture analysis problem. A fourth order POMM using the auto-binomial distribution for the conditional probabilities is proposed.(An advantage of Markov models and POMMs for texture analysis as compared to other structural and statistical techniques is that they also have an associated generative procedure. Textures may be synthesized that correspond uniquely to a set of model parameters.)Flinn's algorithm, an iterative procedure, has been used for synthesis of textures corresponding to a set of parameters.

The idea of generating similar looking unique textures given a set of parameters may be used for the inverse problem, that is, the problem of texture classification. This involves estimation of the parameters of each texture followed by texture classification using the texture parameters features. A maximum likelihood estimation (MLE) assuming the proposed POMM model for the textures is discussed. Owing to the mathematical tractability of expressing the joint distribution for the POMM model, the MLE problem is simplified. The accuracy of the method of parameter estimation is tested on synthesized textures whose parameters are already known. Also, a hypothesis test is developed for testing the accuracy of the parameter estimates.

Another important question is which method is to be used for the classification of an image having a variety of textures. Here, a Bayesian approach is adapted to compute the maximum a posteriori (MAP) estimate of the segmented (labelled) image given the original, unknown image. The essence of the approach to classification is simulated annealing, a stochastic relaxation algorithm which generates a sequence of images that converges in an appropriate sense to the MAP estimate. This sequence evolves by local changes in pixel gray levels in the labelled image. The algorithm is highly parallel and exploits the equivalence between Gibbs distributions and Markov random fields. The theory behind simulated annealing and the "cooling schedule" that guarantees convergence is discussed, and the segmentation results on real images are presented. Also, the application of simulated annealing as a learning rule for morphology neural networks is investigated, and results of this method for image-template identification problems are presented.

This thesis is organized as follows. Chapter 2 gives an overview of classical texture analysis techniques. In Chapter 3, Markov models and the more specific subclasses of POMMs and MMMs for texture analysis are discussed. The theory of POMMs and partial orders are given in Chapter 4. In Chapter 5, a new fourth order POMM model is presented and texture synthesis and parameter estimation techniques are detailed, followed by results of texture synthesis, and parameter estimation of natural images using the POMM. The texture segmentation problem is solved using simulated annealing. In Chapter 6, the theory of simulated annealing and the application of this algorithm to the problem of texture segmentation and training of a morphological neural network is described.

2. TEXTURES: AN OVERVIEW

Image classification maps different regions or segments into one of several objects, each identified by a label. This is achieved through image feature extraction, followed by classification. Important image features include edges and object boundaries, shape features, moments, and textures. Image features are discussed in more detail below. Also, an introduction to texture is followed by a discussion of a selected number of approaches to texture description and segmentation.

2.1. Image Features for Classification

Common edge detectors include the Sobel, Roberts, and Laplacian operators [6]. The application of the edge operator is followed by boundary linking, using, for example, such techniques as dynamic programming, graph search, and the Hough transform [7]. It is known that gradient masks perform poorly in the presence of noise [8], but averaging, low-pass filtering, or least squares edge filtering techniques can yield some reduction in noise effects. A better alternative is a stochastic gradient operator whose filter weights are estimated using the best linear mean square method [8]. Object boundaries can also be represented by chain-codes, Fourier descriptors, and autoregressive models [9]. The objects may then be classified using geometrical features such as perimeter, area, etc. or other moment-based features such as center of mass and eccentricity. Quad-trees [10] and run-length codes are other types of approaches to region representation in binary images. Template matching and morphological transforms of thinning, opening, closing, skeletonizing, maxmin, and medial-axis transforms [11] are useful in extracting predefined shapes in images. These methods are useful for analysis of images with objects that have sharp, well defined edges, and often fail when the region or object boundaries are not prominent.

2.2. Texture: Definition

Many portions of images of natural scenes are devoid of sharp intensity differences over large areas. In these cases, the scene can often be characterized as exhibiting a consistent textural structure.

Texture is observed in the structural patterns of surfaces of objects such as wood, grain, sand, grass and cloth. Image texture measurements can thus be used to segment images where the scenes are described by regional properties rather than edge properties. Texture is an important characteristic for the analysis of many types of images, from multispectral scanner images obtained from aircraft or satellite platforms, to microscopic images of cell cultures or tissue samples. Despite its importance and prevalence in image data, a formal approach or precise definition of texture does not exist. Textures are used to describe regions in images based on the fineness, coarseness and regularity of structure, or its randomness, or lack of structure. Hawkins [12] has provided a more detailed description of texture by emphasizing three main aspects of texture patterns, namely: (1) A "local order" that is repeated over a region which is large in comparison to the size of the order; (2) the "order" consists of nonrandom or partially random arrangement of the elementary parts that make up the texture; and (3) the parts are roughly uniform entities of nearly the same size having approximately the same dimensions everywhere within the textured region.

① A texture may be described by two basic characteristics. The first characteristic is concerned with the gray level primitives or local properties constituting the image texture, and the second characteristic is concerned with the spatial organization of the gray level primitives. The basic texture element which is repeated to form the texture is called a *texel*, for texture element. Gray level primitives are regions with gray level properties. The gray level primitive can be described in terms of gray level properties, such as the average value, or the minimum and maximum values of its regions. A region is a maximally connected set of pixels having a specific gray level property. The gray level region can be evaluated in terms of its area and shape. The gray level primitive includes both its gray level properties and gray level region properties.

An image texture is described by the number and types of its primitives and their spatial organization or layout. The spatial organization may be random, or it may have a pairwise dependence of one primitive on a neighboring primitive, or it may have a dependence of n primitives at a time. The dependence may be structural, probabilistic, or functional, such as a linear dependence. Image texture may be qualitatively evaluated as having one or more of the properties of fineness, coarseness, smoothness, granulation, lineation, or as being mottled, irregular, or hummocky. Each of these qualities translates into a property of the gray level primitives and the spatial interaction between them.

When the gray level primitives are small in size and the spatial interaction between gray level primitives is constrained to be very local, the resulting texture is a microtexture. When the gray level primitives have their own distinct shape and regular organization on a more global scale, the texture is called a macrotexture. This illustrates the point that any texture cannot be analyzed without a frame of reference in which a gray level primitive is stated or implied. For any textural surface, examination at a large scale gives a smooth and textureless effect to the surface. Then, as resolution increases, the surface appears as a fine texture and then a coarse one. For multiple-scale textural surfaces, the cycle of smooth, fine, and coarse may repeat.

Gray level and textural pattern elements are related to each other. When a small area of an image has little variation of gray level primitives, the dominant property of that area is gray level. When the small area has a wide variation of gray level primitives, the dominant property is texture. Crucial in this distinction are the size of the small area patch, the relative sizes and types of gray level primitives, and the number and placement or arrangement of the distinguishable primitives. As the number of distinguishable gray level primitives increases within an area, the texture property will predominate. When the spatial pattern in the gray primitives is random and the gray level variation is wide, a fine texture results. As the spatial pattern becomes more definite and the gray level regions involve more number of pixels, a coarser texture results.

Texture is therefore characterized by the gray level primitive properties as well as the spatial interrelationships between them. This implies that texture is actually a two-layered structure, where the first layer characterizes the local properties of gray level primitives, and the second layer specifies the organization among the gray level primitives.

There are two main approaches to the problem of texture classification:

1. FEATURE BASED APPROACH: features unique to each texture class are extracted to classify regions in the image;
2. MODEL BASED APPROACH: the texture is described by a mathematical model which is then used for texture synthesis and classification according to model parameters. This approach is in general both descriptive and generative.

2.3. Feature Based Approach to Texture

An image feature is a distinguishing attribute of an image. Natural features are defined by the visual appearance of an image, while other so-called artificial features result from mathematical manipulations of images. These features are useful to isolate regions with a common property or similar texture in an image. Features used for texture classification may be subdivided into three types: (1) structural, (2) statistical, and (3) spectral or transform methods.

2.3.1. Structural Techniques

In this model, we view the texture to be composed of primitive, regular structures such as circles, squares, blocks, or other geometrical entities called texels. The texture is then described by the arrangement of these primitives relative to each other [1], [13]. This method is based on the concept of relational descriptors.

Textural edgeness can also be used as a feature for classification. In [14], the authors have used edgeness per unit area as a feature. The authors use the Robert's gradient operator to compute the average gradient in a subimage. An extension of this idea was used in [15] as a textural measure for pulmonary disease identification, where the gradient is a function of the inter-pixel distance.

The structural approach to texture analysis works well on textures with well defined edges and prominent primitive structures.

2.3.2. Transform Techniques

Digital transform coefficients can indicate the degree of correspondence between two gray level patterns or two textures in an image. If a basis pattern is used as a "template" to detect a similar region in an image, then image detection can be performed simply by examining the values of the transform coefficients. For example, analysis of the Fourier spectrum of a region gives information about the spatial structure of a texture, since the degree of texture coarseness is proportional to its spatial period. Regions with fine textures have greater energies at higher frequencies, and vice versa. Hence energies at different frequencies may be used as features. The spectral intensity can be examined in specific regions using a horizontal or vertical slit, a ring or a sector to isolate certain image features [16].

Transforms other than the Fourier have been used for texture analysis. Kirvida [17] compared the fast Fourier, Hadamard and Slant transforms for identifying textural features on aerial images. However, experiments have shown [12] that there is a considerable overlap of regions of distinctly different natural textures such as urban, rural, and woodland regions extracted from aerial images, and hence spectral techniques fail in this case. In general, features based on Fourier power spectra have been shown to perform more poorly than features based on first order statistics of spatial gray level differences [18] or on second order co-occurence statistics [19].

The shortcomings of this approach are due to the fact that these procedures are not invariant, even under a monotonic transformation of gray level or spatial changes in the period and phase of texture pattern repetitions.

2.3.3. Statistical Techniques

Most images of natural scenes are generally random and lack specific structure or order. Textures that are random in nature are well suited for statistical characterization. Statistical texture analysis computes features at each point in a texture image using local information, and derives a set of statistics from the distributions of the local features. The local feature is defined as a combination of gray values at specified positions (pixels) relative to each point in the image. According to the number of pixels which define the local feature, statistics are classified into first-order, second-order, and higher-order statistics. Statistics give various measures of texture properties and are not limited to binary images. Statistics of such local geometrical features as edge, peak and valley, and spots or blobs give measures of specific texture properties.

2.3.3.1. First Order Statistics

The most basic of all texture features is a measure of the image gray level distribution, commonly called the gray level histogram. The measures formulated as quantitative shape description of a first order histogram are: mean, standard deviation, skewness, kurtosis, energy, and entropy. When the problem is to classify texture images, the images are usually normalized to have the same mean and standard deviation because these statistics are affected by the image input conditions. Entropy measures the spread in gray values in a texture. In [20], the authors have used the Kolmogorov-Smirnov (KS) test as a similarity measure for comparison of the cumulative distribution functions of observed images and the reference data to classify aerial images.

First order features are not always sufficient for classification of real data. This is due to the fact that visually differentiable textures may have a similar gray level histogram. Histograms thus do not carry any information about the spatial interrelationships between pixels.

2.3.3.2. Second Order Statistics

Second order histogram features can be incorporated into a *gray level co-occurence matrix*, which is based on the definition of the joint probability distribution of pairs of pixels separated by a fixed distance and angle. This technique of using gray level spatial dependencies is called the *spatial gray level dependence method* (SGLDM). If an image region contains fine texture, the two-dimensional histogram of pixel pairs will tend to be uniform, and for coarse textures, the histogram values will be skewed towards the diagonal of the histogram. Haralick [21] obtained 84% correct classification on terrain classes using 14 features from the gray level co-occurence matrix that included angular second moment, contrast, and correlation. Since the properties used in [21] are redundant, the authors in [22] proposed the use of the Karhunen-Loeve expansion to extract optimal properties from the full property set, and hence reduce the dimension of the feature space. Dyer et. al [23] computed co-occurence features for local properties, such as edge strength maxima and edge direction relationships.

In [18] and [24], the authors used a gray level variant of the co-occurence matrix, called the *gray level difference method* (GLDM), that uses the probability of contrast or difference of gray values occurring in a given spatial relationship, as a textural measure. The probability of a smaller contrast for a coarse texture will be much higher than for a fine texture. Galloway [25] used gray level run length statistics to measure texture. This measure, called the *gray level run length method* (GLRLM), is computable from the co-occurence matrix.

A thorough comparison of the SGLDM, GLRLM, GLDM, and a power spectral approach (PSM) [18] using transform features was carried out by Harlow et al., and they concluded that the SGLDM performed most efficiently and that the PSM was the least powerful of the four techniques. The power of the gray level co-occurence approach is that it characterizes the spatial interrelationships of the gray levels in a textural pattern and is invariant under monotonic gray level transformations. Its weakness is that it does not capture the shape aspects of the gray level primitives. Hence, it is not likely to work well for textures composed of large area primitives. Also, it cannot capture the spatial relationships between primitives that are greater in area than one pixel.

A texture is related to the spatial size of the gray level primitives in an image. Gray level primitives of larger size are indicative of coarse textures, while gray level primitives of smaller size indicate finer textures. The *autocorrelation function* is a feature that can describe the size of the gray level primitives. Since the gray level primitives are spatially periodic, the autocorrelation function also drops and rises accordingly. A region of coarse texture exhibits a higher correlation for a fixed shift than a finer texture. Thus texture coarseness is proportional to the spread of the autocorrelation function. First and second order autocorrelation spread measures have been proposed in [26].

Ashjari [27] has proposed a measure of texture based upon the *singular value decomposition* of a texture sample. After extraction of the singular values of the images, the lower order singular values were used as features for classification. This approach is also a transform method, since it involves mapping of the texture features to a new feature space.

In the *textural energy* approach [28], the image is first convolved with a series of kernels called the *Laws operators*. Then each convolved image is processed with a nonlinear operator to determine the total textural energy in the neighborhood of each pixel. The biggest drawback of this approach is the possibility of significant errors along texture boundaries. Unser [29] and Ade [30] suggest the use of discrete orthogonal transforms such as the discrete sine or discrete cosine transform applied to each pixels' neighborhood as an alternative to the Laws operator.

The co-occurence matrix model, power spectral method, and the autocorrelation matrix are all, theoretically, the same subset of second-order statistics. Each of these can be derived mathematically from the other. Higher order statistics do not necessarily give better results in texture discrimination than lower order statistics. It has been conjectured by Julesz [31] that textures having the same first and second order distributions but different third-order, and presumably higher order distributions, cannot be visually discriminated. Even though Pratt [32] has provided two examples to the contrary, Julesz's theory has been found to hold in most cases, and hence most of the current research on texture analysis does not use higher order statistics than second order statistics.

2.4. Model Based Approach

All of the methods discussed so far employ feature extraction from geometrical or statistical information contained within the texture, followed by classification. Another, more recent, approach is to assume an underlying model for the texture. The major problems to be considered in the statistical model based approach for texture analysis are:

1. Choice of the model depending on the class of textures to be analyzed.
2. Order of the chosen model in terms of the number of parameters.
3. Estimation of the model parameters.

4. Synthetic generation of an image obeying the fitted model.
5. Method of comparison of the properties of the generated textures with natural textures.
6. A segmentation technique to classify an image consisting of various textures.

Mosaic models, fractals, autoregressive models, and Markov models are the most common approaches to texture classification. Mosaic models [2] have been used to model textures with a patchy appearance. This model has a two-level mechanism: one level divides the plane into a set of non-overlapping regions, and the other level colors the individual regions. However, little work has been done on estimation of the model parameters, and hence no algorithm exists to synthesize real textures using the mosaic models. Another texture synthesis technique using second order spatial averages of the texture field [33] has been developed. This algorithm increases in complexity with an increase in the number of gray levels. Mandelbrot [34] has used fractals, a set of functions characterized by non-integer fractal dimensions, to describe patterns in the natural world. The fractal pattern is a self-similar pattern, even in a statistical sense, which has an infinitely recursive texture. The two more important models, autoregressive models and Markov models, are discussed separately in the next two sections.

2.4.1. Autoregression Model Approach

In the field of signal processing, time series analysis is a well known method for prediction of a one-dimensional signal from the past signal values. This method has been extended to image processing by extending the concept of one-dimensional space to two-dimensional space [35]. The linear dependence that one pixel of an image has on another can be stated clearly and can be represented by the autocorrelation function. This linear dependence is exploited by the autoregression model for textures.

The autoregressive (AR) model provides a way to use linear estimates of a pixel's gray value, given the gray values in a neighborhood containing it, in order to characterize texture. For coarse textures, all the coefficients will be similar. For fine textures, the coefficients will vary widely. A one dimensional AR model was used in [36] to synthesize streaky vertical textures. Two-dimensional noncausal autoregressive (NCAR) [37] models have been used for texture synthesis. Given a natural texture, the parameters of the best fitting NCAR models, chosen using the consistent decision rule, are determined using a maximum likelihood method. More recently, autoregressive moving average (ARMA) models for images using a neural network approach [38] has been developed, and a recursive least squares (RLS) method used to extract model coefficients.

The power of the autoregressive linear estimator approach is that it is easy to use the estimator in a mode that synthesizes textures from any initial linear estimator. In this sense, the autoregressive approach is sufficient to capture everything about a texture. Its weakness is that it is poor at representing textures consisting of anything but microtextures.

2.4.2. Markov Model Approach

A more recent and successful addition to the list of texture classification techniques are Markov models and Gibbs distributions. A Markov random field (MRF) model for texture assumes that the texture field is stochastic, and satisfies a certain conditional independence assumption.

Julesz [1], [31], and later Yokoyoma [39] utilized Markov chain state methods to create stochastic texture arrays independently and dependently along rows of the image array. MRFs, a two-dimensional analogue of Markov chains, and Gibbs distributions [40] use clique potentials to describe spatial interrelationships [41]. They have been extensively used, not only for texture synthesis [3], [42], but also for texture classification [3], [43]. The advantage of this approach is that a degree of randomness can be described very easily, while still being able to characterize structural properties in textures. Higher orders are used to describe relationships between pixels that are not spatially close to each other.

In the following chapters, we discuss a MRF called a partially ordered Markov model, and its application to texture analysis.

3. MRFs, POMMs AND MMMs

Markov random fields (MRFs), partially ordered Markov models (POMMs), and Markov mesh models (MMMs) are all stochastic models. An introduction to stochastic models and methods for texture analysis is given below. The use of Markov chains in time series analysis is important. Markov chains and the basic notations used in this thesis are discussed in this section, followed by a discussion of MRFs and Gibbs distributions (GD) in the next section. The chapter ends with the theory of MMMs, its relation to the MRF, and a discussion of the POMM models developed by other authors.

3.1. Stochastic Models

A discrete stochastic field is an array of random variables that are randomly distributed in amplitude, and governed by some underlying joint probability density [44]. By controlling the generating probability density, a particular realization can approximate natural textures surprisingly well.

A good model for texture generation is to assume an array of independent, identically distributed random variables *W(j,k)* that is passed through a linear or nonlinear spatial operator *O()* to produce a stochastic texture array *F(j,k)* as shown in Fig. 1. Texture fields with specified statistical properties can be obtained by controlling the form of the generating probability *P(W)* and the spatial operator. Faugeras [26] tried a technique to invert this model to estimate its parameters. The texture field is first decorrelated by a whitening filter, designed from the knowledge of the autocorrelation function. The whitened field is used as an estimate of the generating process.

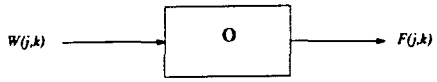


Figure 1. Stochastic texture modeling

MRFs can be used as another approach to two-dimensional analysis of data characterized by conditional probabilities. In order to understand MRFs and MMMs, a brief introduction to Markov chains is given below.

3.2. Markov Chains

Markov chains are used to model a sequence of random variables (r.v.s) where the r.v.s are correlated to each other. Given a set of possible outcomes $\{A_1, A_2, A_3, A_4, ...\}$, which are finite or infinite in number, each r.v. A_n has an associated probability $p_k \equiv P(A_n = k)$. If the events occur in a sequence, and they are independent, their joint distribution can be written as

$$P(\{A_0 = j_0, A_1 = j_1, A_2 = j_2, A_3 = j_3, ...\}) = p_{j_0} p_{j_1} p_{j_3} \cdots$$

If the r.v.s are correlated, however, expression of the joint distribution becomes very difficult. Markov chains assume that the outcome of a r.v. depend on certain preceding r.v.s. In a one-dimensional *first order* Markov chain, the generalization is that the outcome of any trial depends only on the outcome of the directly preceding trial. The outcome of the r.v. $A_n = k$ is no longer associated with a fixed probability p_k, but to every pair $(A_m = j, A_n = k)$ there corresponds a *conditional probability* p_{jk} of a transition from state $A_m = j$ to state $A_n = k$. The conditional probability for a r.v. A_n given the preceding *(n-1)* r.v.s is given as

$$P(A_n = k | A_{n-1} = j, A_{n-2}, ..., A_0) = P(A_n = k | A_{n-1} = j)$$

for a first-order Markov chain, and the joint distribution of the r.v.s is

$$P(all\ A_i) = \prod_i P(A_i | A_{i-1}).$$

For an m-th order Markov chain, the conditional probability of a r.v. is given as

$$P(A_n | A_{n-1}, A_{n-2}, .., A_{n-m}, .., A_0) = P(A_n | A_{n-1}, A_{n-2}, .., A_{n-m}) \text{ leads to}$$

$$P(A_n | A_k, k \neq n) = P(A_n | A_{n\pm1}, A_{n\pm2}, ..., A_{n\pm m}),$$

and the joint distribution of the r.v.s can be written as

$$P(all\ A_i) = \prod_i P(A_i|A_{i-1}, A_{i-2}, .., A_{i-m}).$$

These Markov chains are called stationary or time-homogenous Markov chains since the transition probabilities are independent of the location of the r.v.s in the sequence.

Notation. Throughout the rest of the thesis, capital letters are used to denote r.v.s and the corresponding lower case represents a realization of the r.v. The notation $P(a_n)$ is used to represent the probability of the r.v. A_n taking the value a_n, or $P(A_n = a_n)$.

3.3. MRFs and GD

The use of MRF methods and processing techniques based on these models has increased considerably over the past several years. The pioneering work on MRFs done by Dobrushin [45], Wong [46], and Woods [44] involve extending the Markovian property in one dimension to higher dimensions. MRFs can be used as probabilistic models of digital image texture, where a textured region is viewed as a finite sample of a two-dimensional random process describable by its statistical parameters. MRFs have been used in a variety of applications: stochastic relaxation [42], models for texture [47], and others [3]. In some cases, the models used are adequate for the special processing desired; in others, simplifying approximations are done on the models to make them suitable for the desired processing. However, it was only after the MRF-GD equivalence was discovered [48] that the full power of the MRF began to be exploited and used in image processing applications. Due to this reason, the theory of MRFs and GDs is discussed below.

3.3.1. MRFs: Definition and Theory

The basic definitions pertaining to MRFs and its limitations as models of textures and images in image processing are discussed here. An image is viewed as random field on a finite two-dimensional lattice array $\mathbf{X}$ of random variables (r.v.s) on $\mathbf{Z} \times \mathbf{Z}$, where $\mathbf{Z}$ is the set of integers. Typically $\mathbf{X}$

is an $M \times N$ array and hence, we can write, $\mathbf{X} = \{(i,j) : 1 \leqslant i \leqslant M, 1 \leqslant j \leqslant N\}$ where *(i,j)* is a lattice point or a pixel location.

The MRF assumes a neighborhood system η for $\mathbf{X}$ consisting of a collection of neighborhoods $\eta_{ij} \subset \mathbf{X}$ for each pixel in the lattice $\mathbf{X}$:

$$\eta = \{\eta_{ij} : (i,j) \in \mathbf{X}, \eta_{ij} \subset \mathbf{X}\}.$$

The neighborhood system η must satisfy

1. $(i,j) \not\in \eta_{ij}$, for all $(i,j) \in \mathbf{X}$.
2. If $(k,l) \in \eta_{ij}$, then $(i,j) \in \eta_{kl}$, for all $(i,j) \in \mathbf{X}$

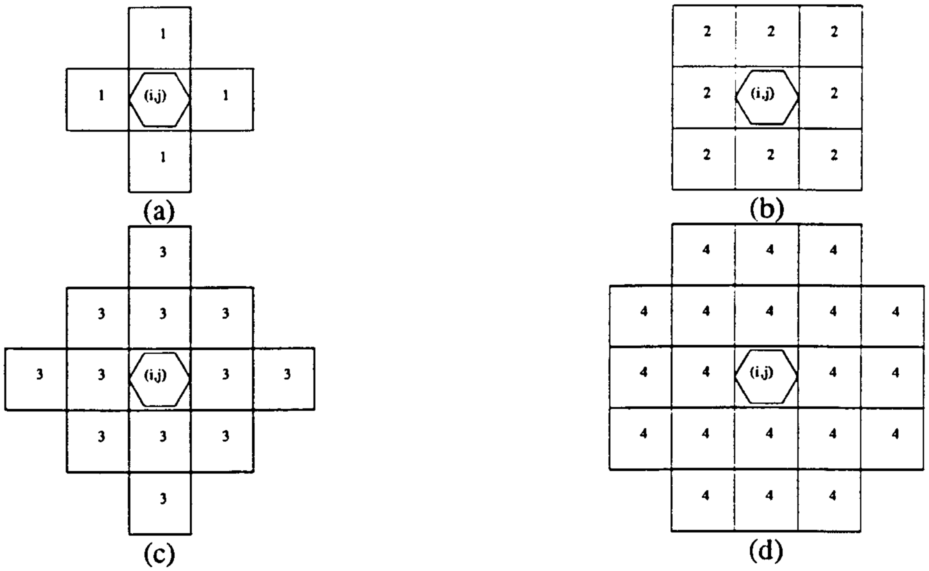
Then, an MRF with respect to the neighborhood system η is defined in terms of a random field $A = \{A_{ij}\}_{(i,j) \in \mathbf{X}}$, which is a MRF with respect to η iff:

$$P(A_{ij} = a_{ij} | A \setminus \{A_{ij}\}) = P(A_{ij} | A_{kl} = a_{kl}, (k,l) \in \eta_{ij}) \;\; for\ all \;\; (i,j) \in \mathbf{X}. \qquad (1)$$

A crucial choice in image modeling is the choice of a neighborhood system. Commonly used ones are $\eta^1, \eta^2, \eta^3, ...$ where

1. $\eta^1 = \left\{\eta^1_{ij}\right\}$ is the set of four neighborhood pixels around *(i,j)* for every $(i,j) \in \mathbf{X}$;
2. $\eta^2 = \left\{\eta^2_{ij}\right\}$ is the set of 8 neighboring pixels around *(i,j)* for every $(i,j) \in \mathbf{X}$;
3. and so forth for all $\eta^3, \eta^4, \eta^5, ...$

In general, η^m is called the *m-th order* neighborhood system. The set of neighborhood pixels for the first, second, third and fourth order neighborhoods are shown in Fig. 2.



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

Figure 2. Neighbors of A_{ij}: (a) first order; (b) second order; (c) third order; and (d) fourth order Markov random field.

A MRF is characterized by the conditional probabilities or the local characteristics of the random field. This concept is intuitively appealing, because in most image processing applications it is natural to expect an image value at a pixel to be correlated to, or be dependent on, spatially close neighbors. Also, for highly correlated data, a higher order neighborhood model may be chosen. Despite the natural fit of the Markov model for natural images, some difficulties encountered while describing the conditional probabilities are:

1. Unavailability of the joint distribution from the local characteristics
2. Consistency difficulties due to lack of definition of a random field or equivalently, the joint distribution
3. A noncausal nature that shows a lack of direction in the model
4. Processing complications due to lack of causality and unavailability of the joint distribution.

Most of these drawbacks are overcome when the MRF-GD compatibility is considered.

3.3.2. Gibbs Distribution

The Gibbs distribution (GD) was first used to model interactions between molecules in ferromagnetic materials by Ising. In addition, GDs have been used in lattice gas models, agricultural studies by Besag [40], neural modeling and artificial intelligence [49], social and economic modeling, optimal VLSI design, and image processing applications [42].

A GD is defined in terms of cliques associated with the lattice-neighborhood system pair $(\mathbf{X}, \eta)$. A *clique* is denoted by c and is defined as:

1. $c \subseteqq \mathbf{X}$
2. For $(i,j) \neq (k,l)$, $(i,j) \in c$ and $(k,l) \in c$ implies that $(i,j) \in \eta_{kl}$.

The clique types of a η^2 neighborhood system is shown in Fig. 3. Then the Gibbs distribution

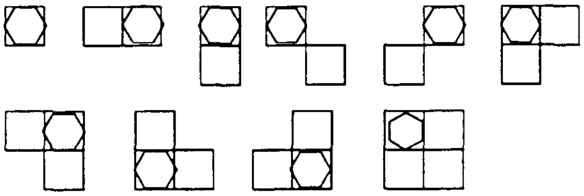


Figure 3. Second order cliques of a Gibbs distribution.

or the Gibbs random field (GRF) can be defined as: If there exists a random field $A = \{A_{ij}\}_{(i,j)\in\mathbf{X}}$ with $a = \{a_{ij}\}_{(i,j)\in\mathbf{X}}$, a single realization of A, and Ω_A, the set of all possible realizations of A, then A is a GRF with respect to the neighborhood system η iff its joint distribution is of the form:

$$P(A = a) = \frac{1}{Z} e^{-U(a)}, \tag{2a}$$

$$U(a) = \sum_{c\in C} V_c(a), \tag{2b}$$

$$Z = \sum_{a\in\Omega_A} e^{-U(a)}. \tag{2c}$$

Here, C is the set of all cliques, $V_c(a)$ is the potential associated with clique c, $U(a)$ is the energy function, and Z is the partition function or a normalizing constant. This terminology appears to have come from statistical physics wherein such measures are "equilibrium states" for physical systems such as ferromagnets. The $V_c(a)$ functions represent contributions to the total energy from external fields (singleton cliques), pair interactions (doubletons), and so on for higher order neighborhood systems.

The particular form of expressions for the GD has the physical interpretation that the smaller the energy of the realization $U(a)$, the more likely is the realization or $P(A = a)$. Frequently, a "temperature" term, T, divides the energy term in equation 2(a), that controls the "peaking" in the distribution. A low energy state can be attained when T is low. Geman and Geman [42] use an "annealing schedule" for image segmentation and restoration whereby T is decreased slowly to allow convergence to a low energy state.

The GD is essentially an exponential distribution. However, depending on the choice of the clique potential functions $V_c(a)$, a variety of random fields, both discrete and continuous, can be expressed as a GD.

The main interest in GD stems from the one-one correspondence between MRFs and GDs that was provided by [40], [50]. The Hammersley-Clifford theorem [51], [50] states that a probability distribution defines a MRF with respect to a neighborhood system η iff it is a GD with cliques associated with η. Then the local conditional distribution of an MRF can be readily obtained from the corresponding GD as:

$$P(A_{ij} = a_{ij} | A \setminus A_{ij}) = \frac{e^{-\sum_{c \in C} V_c(a_{ij})}}{\sum_{a_{ij}} e^{-\sum_{c \in C} V_c(a_{ij})}} = P(A_{ij} | A_{kl} = a_{kl}, (k, l) \in \eta_{ij}) \qquad (3)$$

This equality between MRFs and GD makes the MRF models accessible in two or higher dimensional applications. Most of the drawbacks of the MRF detailed earlier are overcome by this equality. The main advantages resulting from the equivalence are:

1. The joint distribution can be expressed in terms of local characteristics
2. The joint distribution can be written in terms of the model parameters
3. Due to the derivation of the GD from physics, physical interpretation is possible.

3.3.2.1. Commonly used Gibbs Distribution.

The class of GDs presented in the previous section contains a wide class of exponential distributions. A class of GDs called *auto-models* are commonly used that have only single pixel and pair cliques, with the potential functions of the higher order cliques assumed to be zero. Here the form of the energy function U is

$$U(a) = \sum_{x \in \mathbf{X}} G_x a_x + \sum_{x_1, x_2 \in \mathbf{X}, x_2 \in \eta_{x_1}} G_{x_1 x_2}(a_{x_1}, a_{x_2})$$

where G_x and $G_{x_1 x_2}$ are arbitrary functions. The commonly used auto-models are:

1. Auto-logistic model which has been used in binary random variables (r.v.s) and binary images [52], [47], [53]. The energy function is of the form

$$U(a) = \sum_{x \in \mathbf{X}} \alpha_x a_x + \sum_{x_1, x_2 \in \mathbf{X}, x_2 \in \eta_{x_1}} \beta_{x_1 x_2} a_{x_1} a_{x_2}.$$

2. Auto-normal model where the distributions have the form of the joint Gaussian distribution, i.e. the set of r.v.s, $\{a\} \in A$ are jointly gaussian distributed. This model was used in [53] for texture modeling.
3. Auto-binomial model which has been used by Cross and Jain [3] as texture models. The assumption is that each r.v. is conditionally binomially distributed with the binomial parameters determined by the values of the neighboring pixels.
4. Auto-Poisson and auto-exponential models introduced by Besag [40] where each r.v. is respectively Poisson and exponentially distributed, with the parameter values specified by the neighboring r.v.s.

5. Multi-level logistic model that uses a gray level difference count to define the clique potentials as follows:

$$G_x(a_x) = \alpha_m \quad if \quad a_x = m, \forall x \in \mathrm{X}$$

$$G_{x_1x_2}(a_{x_1}, a_{x_2}) = \beta_{x_1x_2}|a_{x_1} - a_{x_2}|^r \quad if \quad a_{x_1} = m, a_{x_2} = n.$$

This model is an extension of the auto-logistic model to the gray level case. The parameters $\beta_{x_1x_2}$ are spatial interaction parameters that affect the energy function based on the gray value difference. A simplifying assumption is that the random field is homogenous. In such a case, the parameters $\beta_{x_1x_2}$ are independent of the pixel locations x_1 and x_2 on the lattice X. Thus, for a η^1 system, the parameters are specified as $\{\beta_1, \beta_2\}$ and for a η^2 system, the parameters are $\{\beta_1, \beta_2, \beta_3, \beta_4\}$. This model is one of the more popular ones and has been used for texture segmentation [43], and texture synthesis using the Gibbs sampler [42].

3.4. Markov Mesh Models

Markov mesh models (MMMs) [5] are a subclass of MRFs that introduce the concept of causality in two-dimensions while retaining the Markovian property of neighborhood dependence. Although the definition of a neighborhood set of a MMM is more constrained than that of a MRF, it allows the expression of the joint probability of a set of r.v.s in terms of conditional probabilities in a closed form expression. MMMs are based on the concept of partial orders that is discussed in the next chapter.

MMMs can be considered to be a 2–D generalization of Markov chains. As given in section 2 in this chapter, for an n-th order Markov chain, the assumption is that

$$P(a_i|a_{i-1}, a_{i-2}, ...) = P(a_i|a_{i-1}, a_{i-2}, ..., a_{i-n}) \qquad \text{leads to} \qquad (4a)$$

$$P(a_i|a_k, k \neq i) = P(a_i|a_{i\pm1}, a_{i\pm2}, ..., a_{i\pm n}) \text{ and} \qquad (4b)$$

$$P(\text{all } a_i) = \prod_k P(a_k|a_{k-1}, a_{k-2}, ..., a_{k-n}). \qquad (4c)$$

We shall see that the corresponding equations for the MMM can be expressed in a similar form.

The definition of the Markov mesh model proposed by Abend to classify binary patterns [5] is as follows:

1. Let A be the set of r.v.s in the $M \times N$ lattice $\mathbf{X}$, and let a_{ij} denote the r.v. at location (i, j).
2. Let $\mathbf{X}_{ij}$ be the $i \times j$ array of r.v.s satisfying $\mathbf{X}_{ij} = \{a_{hk} : 1 \leqslant h \leqslant i \;\; and \;\; 1 \leqslant k \leqslant j\}$
3. Let Z_{ij} be the array satisfying $Z_{ij} = \{a_{hk} : h < i \;\; or \;\; k < j\}$ as shown in Fig. 4(a) below.

The set Z_{ij} corresponds to the set of r.v.s that "precede" or "occur before" A_{ij}. The MMM given by Abend assumes that the probability of A_{ij} conditional on Z_{ij} is equal to the probability of A_{ij} conditional on a set of neighboring pixels close to A_{ij} and contained in the array Z_{ij}, or,

$$P(a_{ij}|Z_{ij}) = P(a_{ij}|N_{ij}), \tag{5}$$

where $N_{ij} \subset \mathbf{X}_{ij} \setminus \{a_{ij}\}$.

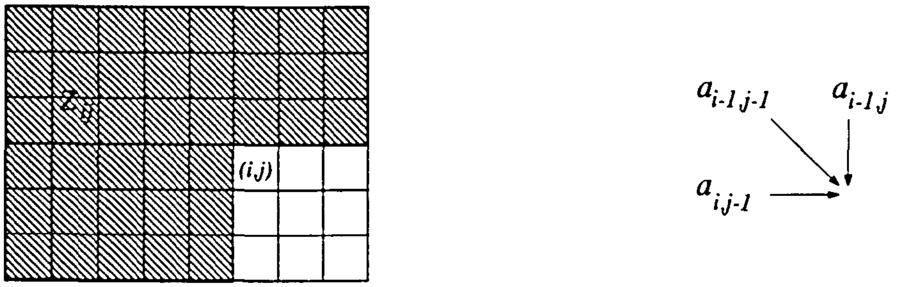
For a third order Markov mesh as given by Abend [5], the assumption that

$$P(a_{ij}|Z_{ij}) = P(a_{ij}|a_{i-1,j}, a_{i-1,j-1}, a_{i,j-1}) \qquad \text{leads to} \tag{6a}$$

$$P(a_{ij}|\mathrm{a} \setminus \{a_{ij}\}) = P\left(a_{ij}| \begin{array}{ccc} a_{i-1,j-1} & a_{i-1,j} & a_{i-1,j+1} \\ a_{i,j-1} & & a_{i,j+1} \\ a_{i+1,j-1} & a_{i+1,j} & a_{i+1,j+1} \end{array} \right) \text{ and} \tag{6b}$$

$$P(\mathrm{a}) = \prod_{(i,j)} P(a_{ij} \mid a_{i-1,j}, a_{i-1,j-1}, a_{i,j-1}), \tag{6c}$$

with appropriate modifications on the boundaries. Here, Z_{ij} is the set of r.v.s in the shaded region in Fig. 4(a), and the set of adjacent neighbors $N_{ij} = \{a_{i-1,j}, a_{i-1,j-1}, a_{i,j-1}\}$ is depicted in Fig. 4(b). Thus we see that the neighborhood assumptions Eq. 4(a) and Eq. 6(a) in the respective models



(a) (b)

Figure 4. (a) The set Z_{ij}. (b) The adjacent neighbors of a_{ij}.

lead to results that are similar in form: compare Eq. 4(b) with Eq. 6(b), and Eq.4(c) with Eq. 6(c). Also, a more important observation is that the neighborhood assumption of Eq. 6(a) leading to the dependence as shown in Eq. 6(b) shows the equivalence of the MMM to the MRF; this is not obvious from the expression of the conditional probabilities. The model exhibits an indirect dependence of a pixel on all 8 neighboring pixels around it, that are both "causal" and "non-causal," and thus matches the MRF in terms of spatial dependence. Also, the joint distribution can be expressed as a multiplicative product, thus making the computation of the joint probability more tractable as compared to a MRF. Other choices of the neighborhood system have been given in [5].

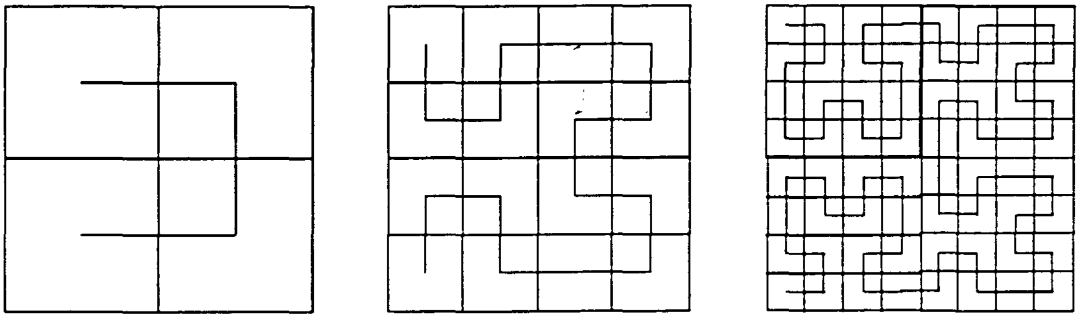
Several other researchers have used other types of MRFs that are similar in idea or concept to the MMMs. Some of these methods fall under the more general category of *partially ordered Markov models* (POMMs.) Devijver [54] has used the third order MMM developed by Abend for image segmentation and restoration by using a deterministic relaxation technique for noise removal. In [55], Lacroix has addressed the problem of pixel labeling under the assumption that the image can be represented by a second order MMM, and presented results for noise removal in binary images. Qian and Titterington [56] have used multi-dimensional Markov chains as texture models in which each row is regarded as a random vector. The multi-dimensional Markov chain model assumes that the distribution of the current row conditionally on all previous rows depends only on a number of past, neighboring rows, assuming the topmost row to be the first row. Goutsias [57] has studied the equivalence between mutually compatible Gibbs random field (MCGRF) and Markov meshes, and used the MCGRF for texture synthesis. In order to obtain a recursive form for the Gauss-Markov random fields (GMRFs), Moura [58] has used a canonical representation for the GMRFs based on the inverse of the covariance matrix. Similar work has been done by others [59]. The theoretical concepts underlying partial orders and its relation to Markov models is still being investigated by several researchers [60], [61]. The next chapter introduces the theory of partial orders and its relationship to Markov mesh models.

4. PARTIAL ORDERS AND PARTIALLY ORDERED MARKOV MODELS

Partially ordered Markov models (POMMs) and Markov mesh models (MMMs) are based on the concept of partial orders. As has been shown [60], MMMs are a special type of POMMs, and thus MMMs have an underlying *partial order* as well. The concept of partial orders is introduced in Section 4.1, along with the definitions and terminology. Section 4.2 describes the relationship between POMMs and partial orders.

4.1. Partial Orders

The Markov mesh model [5] was formulated to incorporate the concept of ordering in 2–dimensions. Abend et al. suggested the use of space-filling curves as shown in Fig. 5 in an attempt to use Markov chains in 2–dimensions. However, it is easy to see that sites that are spatially close on the grid may not be sequentially close along the path. This sequential ordering is a *total order* on the set of pixel locations.



(a) (b) (c)

Figure 5. A space filling curve. (a) First iteration. (b) Second iteration. (c) Third iteration.

Total order. Let X be a set and let $\mathcal{R}$ be a subset of $X \times X$, that is, $\mathcal{R}$ describes a binary relation on X. If $(x, y) \in \mathcal{R}$ we write $x\mathcal{R}y$. The relation $\mathcal{R}$ is called a *total order* for X if

1. $x\mathcal{R}y, y\mathcal{R}z \Rightarrow x\mathcal{R}z$ (transitivity)

2. for any pair of elements $x, y \in X$ such that $x \neq y$, then $x\mathcal{R}y$ or $y\mathcal{R}x$, or if $x = y$, then $x\mathcal{R}x$, that is, $(x, x) \in \mathcal{R}$ (law of trichotomy)

Partial order. A set of elements X with a binary relation $\prec$ is said to have a *partial order* with respect to $\prec$ if the following hold:

1. $x \prec x \quad \forall\, x \in X$ (reflexivity)
2. $x \prec y, y \prec z \Rightarrow x \prec z$ (transitivity)
3. If $x \prec y$ and $y \prec x$ then $x = y$ (anti-symmetry).

In the second definition, $(X, \prec)$ is called a *partially ordered set*, or a *poset.* Note that the main difference between a total and a partial order is that in a poset, it is not necessary that every two elements be related, that is, the law of trichotomy is not necessarily satisfied. A *finite poset* is a poset $(X, \prec)$ where the cardinality of the set X is finite, that is, $| X | < \infty$. The poset used by Abend et al. (unknowingly) in their definition of MMM was the following: Let

$$X = \{(i,j) : 1 \leqslant i \leqslant M, 1 \leqslant j \leqslant N\}, \tag{7}$$

and let $(i,j), (h,k)$ be two sites. Define the binary relation $\prec$ on X by

$$(i,j) \prec (h,k) \Leftrightarrow i \leqslant h \text{ and } \overset{j}{\cancel{h}} \leqslant k. \tag{8}$$

That is, $(i,j) \prec (h,k)$ if and only if $i \leqslant h$ and $j \leqslant k$, provided that $(i,j) \neq (h,k)$, or if and only if $(i,j) = (h,k)$. This relation satisfies the condition for a partial order of the set X. In this poset the set of sites unrelated to (*i,j*) is the set Y_{ij} consisting of the two shaded blocks as shown in Fig. 6.

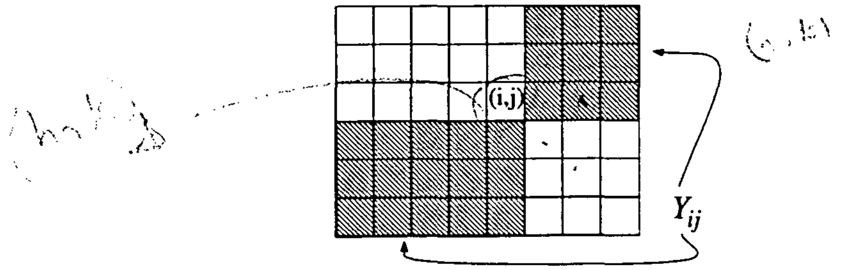


Figure 6. Set of sites Y_{ij} unrelated to (*i,j*) under the partial order in Eq. 8.

In image processing, an image may be viewed as an array of r.v.s $A = \{a_{x_k}\}$ with a finite indexing set $X = \{x_k : 1 \leqslant k \leqslant N\}$. This may be expressed as $A_k \equiv A_{x_k}$, and is used for the rest of the thesis for notational convenience. We abuse notation and say that r.v. A_k is related to *random variable* A_i under the binary relation $\prec$ if and only if index x_k is related to index x_i under the binary relation $\prec$. Then the partially ordered set of r.v.s $(A, \prec)$ implies that if $A_k \prec A_i$, then index x_k is related to index x_i under $\prec$, that is, $x_k \prec x_i$.

Since partial orders are closely related to directed graphs, a brief introduction to graphs is given here. A graph [62] is a pair (V, E), where $V = \{v_i\}_{i=1}^N$ denotes a finite set of vertices or nodes, and $E = \{\{v_i, v_j\}\}$ is a (possibly empty) set of unordered pairs of distinct vertices of *V*; *E* is called the set of *edges.* An edge $\{v_i, v_j\}$ (or $\{v_j, v_i\}$) connects the two vertices v_i and v_j with a line segment. A directed graph, or digraph, is a graph whose edge set consists of ordered pairs (v_i, v_j). Thus, the edge (v_i, v_j) is distinct from the edge (v_j, v_i), one, both, or neither of which may be in the edge set *E*. The direction of the edge (v_i, v_j) is from vertex v_i to vertex v_j, and is represented by an arrow with its tail on v_i and its arrow head on v_j. A *cycle* in a directed graph is a sequence of distinct vertices, $v_1, v_2, \ldots, v_k = v_1$, except for v_1 and v_k which are the same $(v_k = v_1)$, and where (v_i, v_{i+1}) is a directed edge in the digraph (V, E), *i=1,...,k-1*. An acyclic digraph is a digraph that has no cycles in it. The correspondence between a finite poset $(X, \prec)$ and its unique digraph is given after introducing several definitions.

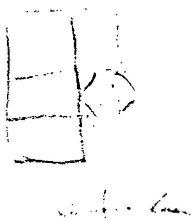
$B \in X$

$\text{cone } B = \{C \in X : C \prec B\}$

Let $(X, \prec)$ be a finite poset.

Definition 1. For any $B \in X$ the *cone* of *B* is the set

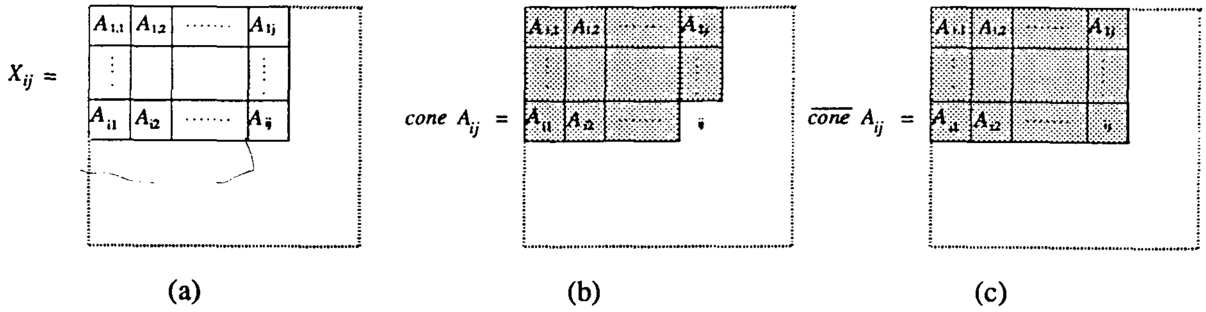
$$\mathit{cone}\, B = \{C \in X : C \prec B\}.$$



Definition 2. The *closure* of *cone B* is the set of r.v.s

$$\overline{\mathit{cone}}\, B = \{C \in X : C \preceq B\} = \{B\} \cup \mathit{cone}\, B.$$

As in Abend et al.'s model, if we define $X_{ij} \equiv \{(h, k) : 1 \leqslant h \leqslant i, 1 \leqslant k \leqslant j\}$, then we have $\mathit{cone}\, A_{ij} = X_{ij} \backslash \{A_{ij}\}$ and $\overline{\mathit{cone}}\, A_{ij} = X_{ij}$. This is portrayed in Fig. 7.



(a) (b) (c)

Figure 7. (a) The subarray X_{ij}. (b) The set *cone* A_{ij}. (c) The set $\overline{cone}\ A_{ij}$.

Definition 3. Elements B and C in X, with $B \neq C$, are said to be *adjacent* if either

1. $C \succ B$ and there is no other element D such that $C \succ D \succ B$; or
2. $C \prec B$ and there is no other element E such that $C \prec E \prec B$

For example, in Abend et al.'s model, A_{ij} is adjacent to $A_{i,j-1}$ but not to $A_{i,j-2}$, as shown in Fig. 8.

Definition 4. For any $B \in X$, the *adjacent lower neighbors of B* are the elements adjacent to B

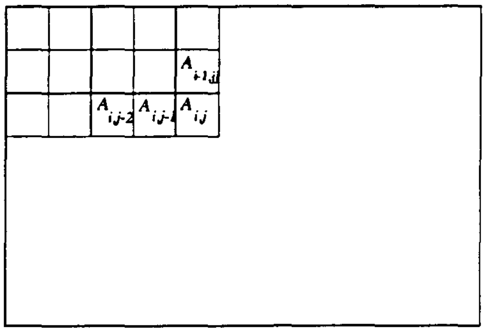


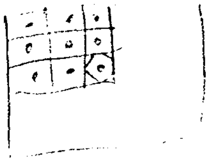
Figure 8. Adjacency relationships in Abend's third order model.

and less than B. Formally,

$$adj_{\prec}\, B = \{C : C \prec B \text{ and } B \text{ is adjacent to } C\}.$$

In Abend et al.'s third order model,

$$adj_{\prec}\, A_{ij} = \{A_{i,j-1}, A_{i-1,j-1}, A_{i-1,j}\}.$$



Definition 5. The *closure* of $adj_{\prec}B$ is the set $\overline{adj}_{\prec}B \equiv \{B\} \cup adj_{\prec}B$.

Definition 6. For any element $B \in X$ the *dilation of B with respect to the partial order* is the set

$$dil\ \ B = \bigcup_{\{Z:B\in\overline{adj}_{\prec}Z\}} \overline{adj}_{\prec}Z. \tag{9}$$

In Abend et al.'s third order example, we have

$$dil\ \ A_{ij} = \bigcup_{\{Z:A_{ij}\in\overline{adj}_{\prec}Z\}} \overline{adj}_{\prec}Z = \begin{Bmatrix} A_{i-1,j-1} & A_{i-1,j} & A_{i-1,j+1} \\ A_{i,j-1} & A_{ij} & A_{i,j+1} \\ A_{i+1,j-1} & A_{i+1,j} & A_{i+1,j+1} \end{Bmatrix},$$

which is exactly the eight nearest neighbors of A_{ij} plus itself.

Definition 7. The set $dil^{*}B$ is the set

$$dil^{*}B\ \ = [dil\ \ B]\backslash\{B\}.$$

Definition 8. An element *B* is called a *minimal element* if there is no other element *C* satisfying $C \prec B$, and a *maximal element* if there is no other element *D* satisfying $D \succ B$. The set of minimal elements is denoted by X_{min}, while X_{max} is the set of maximal elements in *X*. In Abend et al.'s model, the set of minimal elements is $A_{min} = \{A_{11}\}$, while $A_{max} = \{A_{MN}\}$.

Definition 9. For a subset $R \subset X$ the *cover* of *R* is the set of elements of *X* not in *R* whose adjacent lower neighborhood lies entirely in *R*:

$$cover\ R = \{Z \notin R : adj_{\prec}Z \subset R\}.$$

For example, in Abend et al.'s third order model,

$$cover\ A_{min} = cover\ \{A_{11}\} = \{A_{12}, A_{21}\}.$$

The element A_{22} is not in the set $cover\ \{A_{11}\}$, as $adj_{\prec}A_{22} = \{A_{11}, A_{21}, A_{12}\} \not\subset \{A_{11}\}$.

The unique relation between a poset $(X, \prec)$ and a digraph can then be described as follows. Let $X = \{x_1, \ldots, x_n\}$ and let $adj_{\prec}x$ denote the set of lower adjacent neighbors for each $x \in X$. The set of vertices of the digraph is the set X, and the edges $D = \{d_{ij}\}$ of the digraph are constructed as follows: d_{ij} is a directed edge from x_i to x_j if and only if $x_i \in adj_{\prec} x_j$.

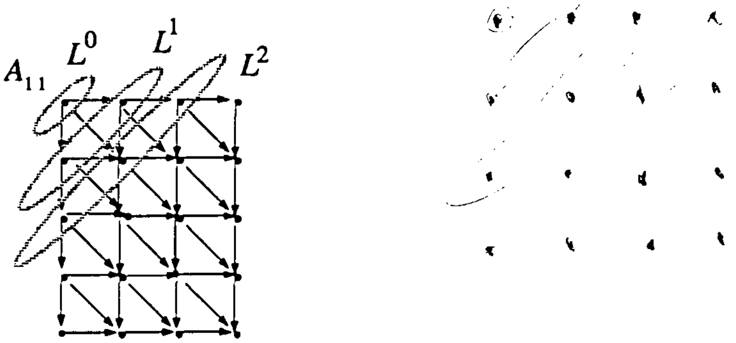
Since acyclic digraphs and posets are one and the same, the notation and definitions are used interchangeably.

Another way of defining a partial order on a set is to state all the relationships $adj_{\prec} x$ for all elements x in the set X, and determine if the induced digraph is acyclic. The third order example of Abend et al.'s specifies a partial order by defining $adj_{\prec} A_{ij} = \{A_{i-1,j}, A_{i-1,j-1}, A_{i,j-1}\}$ for all (i, j) sites in the array (except on the boundary). The resulting digraph from such an induced poset must be checked for cycles; in Abend's case such a valid partially ordered set of sites exists.

Definition 10. A unique sequence of nonempty cover sets can be described in the following way. Let $L^0 = X_{min}$, and recursively define $L^i = cover\left(\bigcup_{k=0}^{i-1} L^k\right)$. These sets are the *level* sets for X. The level sets for Abend et al.'s third order model are given in Fig. 9.

$L^1 = cover(L^0)$

$L^2 = cov.(L^0 \cup L^1)$



(a)

Figure 9. The level sets for Abend et al.'s third order model.

The following properties of level sets will be needed later. The notation X is used to define a finite poset, and L^i is to denote the *i-th* level set, as in Definition 10.

4.2. Partially Ordered Markov Models

The definition of partial orders in the previous section has laid the theoretical foundations for the development of POMMs. The formal definition of POMMs and the theoretical probabilistic results that describe POMMs are presented in this section.

As in the previous section, the poset of r.v.s $(A, \prec)$ implies that if $A_k \prec A_i$, then index x_k is related to index x_i under $\prec$, that is, $x_k \prec x_i$. It is assumed that the poset A does not have any *singleton* points, that is, points corresponding to independent r.v.s in a statistical sense.

Definition. A *partially ordered Markov model* (POMM) is defined as follows. Let $B \in A$ where $(A, \prec)$ is a finite poset of r.v.s. The set of r.v.s not related to B is denoted by

$$Y_B = \{C : B \text{ and } C \text{ are not related}\}.$$

Then the random field A is called a *partially ordered Markov model* (POMM) if for any subset $U_B \subset Y_B$,

$$\checkmark \; \boxed{P(B|cone\, B, U_B) = P(B|adj_{\prec} B).} \tag{10}$$

This must hold true for any $B \in A$ satisfying $B \notin L_A^0$. If $B \in L_A^0$, then $cone\ B = \emptyset$, and $P(B|\emptyset) = P(B)$.

The properties of POMMs can be summarized in the two theorems presented below. The proofs of these theorems are given in [61].

Theorem 1. *Let $(A, \prec)$ be a POMM, and Z be a subset of unrelated elements in A, that is, $Z = \{Z_1, Z_2, ..., Z_k\}$ where the Z_is are pairwise unrelated under $\prec$. Then*

$$P(\overline{cone}Z) = P(L_Z^0) \cdot \prod_{C \in [\overline{cone}Z] \setminus L_Z^0} P(C|adj_{\prec} C). \tag{11}$$

This theorem states that the marginal probability of a subset of r.v.s that are not related can be expressed as a product of the local conditional probabilities. The corollary that follows from this theorem that is of interest to us is:

Corollary 1. *Let $(A, \prec)$ be a POMM, and Z be the set of maximal elements in A, that is, $Z = A_{max}$. Then, the joint probability of all the r.v.s in A is*

$$P(A) = P(L^0) \cdot \left[\prod_{\{C : C \in A \setminus L^0\}} P(C|adj_{\prec} C) \right]. \tag{12}$$

Corollary 1 gives us a closed-form expression for the joint probabilities of all r.v.s in terms of its conditional probabilities.

Theorem 2 states that the conditional probability of one r.v. B conditioned on the remaining r.v.s in the image depends only on the r.v.s in the set $dil^{*}B$:

Theorem 2. *Let* $(A, \prec)$ *be a POMM,* $B \in A$*, and let* $Q_B = \{C \in A : B \in \overline{adj}_{\prec}C\}$*. Then the probability of B conditioned on the rest of the image is*

$$P(B|A \setminus \{B\}) = P(B|dil^{*}B) = \frac{\prod\limits_{C \in Q_B} P(C|adj_{\prec}C)}{\sum\limits_{All\ possible\ values\ of\ B} \left[\prod\limits_{C \in Q_B} P(C|adj_{\prec}C) \right]}. \qquad (13)$$

These theorems can be used to derive the conditional probabilities and joint distributions of a set of r.v.s satisfying a given partial order, and have been used to derive the partials and joint distributions of the fourth order POMM developed in the next chapter for texture modeling.

5. POMM FOR TEXTURE MODELING

In the previous two chapters, the theory of MMMs and the more general class of POMMs was discussed. The feasibility of using the POMMs to model images, in comparison with a MRF, is summed up below:

1. In a POMM, the joint distribution can be expressed as a product of the conditionals due to the algebraic structure of the POMMs induced by the partial order.
2. In POMMs, the lack of the partition function makes the computation of the joint distribution much easier.
3. Due to 1. and 2., the POMM is better suited to computer implementation of stochastic optimization procedures.

In this chapter, a fourth order POMM is introduced for modeling of textures. The specific order of the POMM and the model for the distribution of the conditionals of the r.v.s is presented in Section 1. A texture synthesis process is described in Section 2. The texture synthesis results for binary, four and sixteen gray level images are presented in Section 3. Section 4 deals with the inverse problem of estimating the parameters of unknown images. Results of parameter estimation on synthetic and natural images are given in Section 5, as well as the results of a statistical test to test the validity of some of the estimates.

5.1. A Fourth-Order POMM for Textures

In choosing a POMM, we would like one that can model a relatively wide variety of textures. Since directionality is an important characteristic of textures, a fairly good model would be one that could represent a dependence or pixel interactions in the four principal directions of north, northwest, west, and southwest. Also, equally important, the chosen model should have a dependence that exhibits a partial order so that the full power of the POMM properties can be applied. Lacroix

[55] and Devijver [54] have outlined labeling methods for second and third-order MMMs similar to the one proposed by Abend [5]. They each used a third-order MMM, for image restoration and segmentation respectively.

The third-order MMM used in [5] and [54] has a direct dependence on the north, northwest, and west neighboring pixels of a pixel. This, however, does not have the additional dependence in the southwest direction, and hence it is possible that the parameters in this third order-model could not capture the orientation in the texture in the southwest direction. A fourth-order POMM is suggested here that perhaps can intuitively capture the directionality in textures in a broader context than the third-order MMM model. If the value at a pixel conditional on "previous values" (relative to the partial order $\prec$) is conditioned on only the pixels values at the north, northwest, west, and southwest locations near the pixel,

$$P(a_{ij}|Z_{ij}) = P(a_{ij}|N_{ij}) = P(a_{ij}|a_{i-1,j}, a_{i-1,j-1}, a_{i,j-1}, a_{i+1,j-1}), \tag{14}$$

then the value of the r.v. at location *(i,j)* conditioned on the rest of the image is

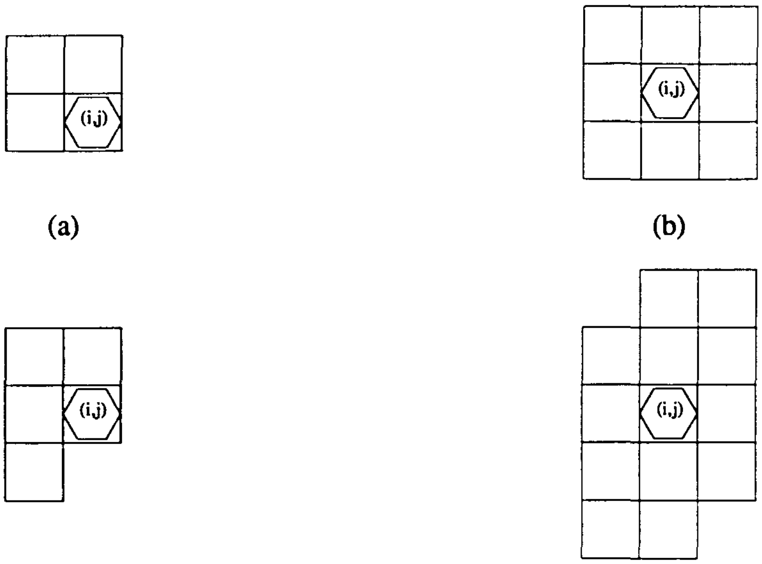
$$P(a_{ij}|\mathbf{X} \setminus \{a_{ij}\}) = P(a_{ij}|\eta_{ij}) = P\left(\begin{array}{cccc} & & a_{i-2,j} & a_{i-2,j+1} \\ & a_{i-1,j-1} & a_{i-1,j} & a_{i-1,j+1} \\ a_{ij}| & a_{i,j-1} & & a_{i,j+1} \\ & a_{i+1,j-1} & a_{i+1,j} & a_{i+1,j+1} \\ & a_{i+2,j-1} & a_{i+2,j} & \end{array} \right). \tag{15}$$

The configurations of the sets N_{ij} and η_{ij} are shown below in Fig. 10.

Note that the general assumption for a MMM given by Abend et al. [5] is

$$P(a_{ij}|Z_{ij}) = P(a_{ij}|U_{ij}), \tag{16}$$

where Z_{ij} is the set of r.v.s as described in Eq. 4(a), and U_{ij} is any subset of $\mathbf{X}_{ij}$, where $\mathbf{X}_{ij}$ is the $i \times j$ subarray of r.v.s as given in Section 3.4. It is important, therefore, to note that the fourth order POMM as given in Fig. 10(c) and (d) includes the pixel $A_{i+1,j-1}$, which is not in the subarray $\mathbf{X}_{ij}$, and therefore this POMM is <u>not</u> a MMM.



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

Figure 10. Configurations of (a) N_{ij} and (b) η_{ij} for Abend's model, and (c) N_{ij} and (d) η_{ij} for fourth-order POMM model.

5.1.1. Auto-binomial Model for POMM

A number of distributions for the MRF models were discussed in Chapter 3. They include: the auto-logistic model, exponential model, Poisson model, multilevel logistic model, and the auto-binomial model. The multilevel logistic model has been used in a number of MRF applications [43], [42].

The auto-binomial model, suggested by [40], was successfully used by Cross and Jain [3] for texture analysis. The auto-binomial model assumes that the value at a pixel conditioned on its neighbors is binomially distributed, with the model parameter value q_{ij} depending on the neighboring pixel values. For the fourth order POMM proposed here, the spatial interaction between pixels in terms of conditional probabilities is given by

$$P(a_{ij}|N_{ij}) = {}^{G-1}C_{a_{ij}} q_{ij}^{a_{ij}} (1 - q_{ij})^{G-1-a_{ij}}, \text{ where} \tag{17a}$$

$$q_{ij} = \frac{e^{T_{ij}}}{1 + e^{T_{ij}}} \text{ and} \tag{17b}$$

$$T_{ij} = \alpha + \beta a_{i-1,j} + \gamma a_{i-1,j-1} + \delta a_{i,j-1} + \epsilon a_{i+1,j-1}. \tag{17c}$$

Through the value of T in Eq. 17(c), it is proposed that this distribution can describe the nonlinear interrelationships between neighboring pixels effectively as compared to the multilevel logistic model, which uses the clique potentials to measure the gray value difference count in the neighborhood of a pixel to compute the conditional probabilities. See Fig. 11 for a pictorial representation of the locations of the parameters in Eq. 17.

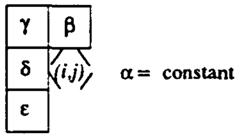


Figure 11. Location of parameters for the fourth order POMM model

The proposed fourth-order POMM model is characterized by its parameters, each of which corresponds to one of the four particular directions of north, northeast, west, and southwest. Each parameter indicates the degree of orientation in the direction to which it corresponds. The larger a parameter value, the greater the orientation in that direction for a certain texture. With the addition of a constant parameter, α, this set of five parameters together controls the size and directionality of the clusters in adjacent pixels, qualities which are basic to texture description. Due to the recursive form of the model, the effects in each *neighborhood* are carried over to the surrounding neighborhoods, resulting in a global effect on the whole image.

5.2. Simulation of a POMM Random Field

The generation of realistic texture is an important area from both theoretical and practical viewpoints. Several techniques for the generation of realistic looking images have been proposed in the literature [63], [64], [65]. In order to generate textures that are the visual representation of the POMM random fields, a procedure is needed that yields a sample from a POMM random field with a given set of parameters. A number of stochastic and deterministic procedures have been used for generation of MRF samples. Since POMMs are a subclass of MRFs, all techniques used for MRF simulation should also apply to POMM models. Hence, several commonly used techniques for MRF simulations are discussed below, and a particular method chosen for the POMM generation problem.

One possible approach to generating a desired joint distribution would be to investigate all possible configurations. For an $M \times N$ size image having L gray levels, there would be $L^{M \times N}$ possibilities, a large number for even a small image, and hence such a direct search is ruled out. One method used in [66] for texture synthesis is based on simulation methods that investigated properties of two and three-dimensional Ising lattice systems [67]. Two other well-known methods exist for generation of realizations from MRFs or GDs, namely the "spin-flip algorithm" and the "exchange algorithm." Both of these algorithms are based on the Metropolis algorithm [68]. Since the Metropolis algorithm is fundamental to these two approaches, the theory of the Metropolis algorithm is discussed next.

The Metropolis algorithm, a probabilistic method, was invented to study the equilibrium properties of very large systems of identical, interacting components in physical systems. Let ω represent a particular site configuration in a physical system, and let Ω denote all possible configurations so that $\omega \in \Omega$. If the system is in equilibrium with its surroundings, then the probability of ω is given by

$$\pi(\omega) = \frac{e^{-\beta V_c(\omega)}}{\sum\limits_{\omega} e^{-\beta V_c(\omega)}},$$

where $V_c(\omega)$ is the potential energy of ω, and $\beta = 1/kT$, k is the Boltzmann's constant, and T is the absolute temperature. This expression for the equilibrium is similar in form to the GD. A sampling theorem has been proved in [68] which ensures that the equilibrium state can be attained from any initial state. Briefly, the Metropolis algorithm is

1. Given the state of the system at "time" t, say $X(t)$, choose another configuration $\widehat{X}(t)$ by changing one variable in the system
2. Compute the energy change $\Delta V = V\left(\widehat{X}(t) - X(t)\right)$ and $q = \frac{\pi\left(\widehat{X}(t)\right)}{\pi(X(t))} = e^{-\beta \Delta V}$.

3. If $q>1$, then next state of the system is $\widehat{X}(t)$, i.e., $X(t+1)=\widehat{X}(t)$; otherwise, $X(t+1)=\widehat{X}(t)$ with probability q.

This algorithm involves Markov chains with state space Ω [42]. The goal of the algorithm is convergence to an irreducible, aperiodic chain with equilibrium measure π. This approach is very well suited to MRFs and its subclasses, since it involves ratios of probabilities of random fields that differ in value at a maximum of two or fewer r.v.s in the field. For a MRF, this reduces to calculations of neighborhood joint probabilities, and hence the computational burden is decreased enormously, excepting the determination of the normalizing constant.

The problem of MRF synthesis and the synthesis of MRF subsets can be posed as a probabilistic relaxation problem, where the probability $P(A)$ of an image A is cast as the equilibrium distribution of an irreducible, aperiodic Markov chain $\{A(t), t=1,2,3,4,...\}$. A realization A is obtained by visiting each state of the chain infinitely often. Let $A(t-1)$ be the state of the chain at time $(t-1)$, and $A(t)$ be the state at time t. The image $A(t)$ is the same as $A(t-1)$ except for the gray values at two locations. Then, the evolution of the chain to the equilibrium state is independent of the initial state, provided an infinite number of transitions have been carried out [42], and so that $P(\mathbf{A})=\lim_{t\to\infty} P(A(t)=A|A(0)=A_0)$ for an arbitrary A_0. The two commonly used techniques for MRF simulation, the "spin-flip algorithm" and the "exchange algorithm," are variations of the above method, and differ from each other in the way in which the transitions are carried out.

The "spin-flip" technique, also known as the Gibbs sampler, was developed by Geman and Geman [42] to obtain samples from the GD. In this method, a pixel is chosen at random or in a deterministic manner. The value of the pixel A_{ij} is changed to another random gray value, $\widehat{A}_{ij}$. The new value $\widehat{A}_{ij}$ is chosen based on the conditional distribution at location (i,j) given its neighbors. The pixel visiting mechanisms can be random or deterministic, such as a raster scan, the important point being that each pixel should be visited infinitely often. The major drawback of this method in implementation is the long convergence time.

Another algorithm for texture synthesis is the "exchange algorithm" or Flinn's algorithm [52], also used by Cross and Jain [3]. In this algorithm, a distribution of gray values is chosen for the image, and the pixel values in the image are initialized randomly according to the distribution. This distribution remains fixed for the duration of the algorithm. Two pixels are chosen at random and their gray values are exchanged if they are different. If the resulting new configuration $\widehat{A}$ is more probable than the old one A, the new configuration, $\widehat{A}$, is retained; otherwise, $\widehat{A}$ is retained with a probability $r = \frac{P(\widehat{A})}{P(A)}$. This algorithm is guaranteed to converge to a configuration that maximizes the joint probability [69]. A major drawback of this algorithm is that the distribution of gray values in the image remains fixed, due to the fact that new configurations are generated simply by exchanging pixel values. Thus, care must be taken to choose a "good" gray level distribution. But, it is due to this reason that the convergence to the joint distribution may be faster, since the search space for the algorithm then becomes smaller. Flinn's algorithm was used here for texture synthesis, employing the fourth-order POMM model, and is outlined below.

5.2.1. Flinn's Algorithm for Fourth-Order POMM Model Synthesis

1. Set the iteration number *t* to zero, *t=0*, and generate the initial image $A(0)$ using a predefined distribution for the gray values in the image.
2. Generate a new image $\widehat{A}(t)$ by interchanging two randomly chosen pixel values a_{ij} and a_{hk}, where $a_{ij} = a_{hk}$ in $A(t)$.
3. Compute $r = \frac{P(\widehat{A}(t))}{P(A(t))}$.
4. The new image $A(t+1)$ is chosen as

$$A(t+1) = \begin{cases} \widehat{A}(t) & if \quad r > 1 \\ \widehat{A}(t) & if \quad r > p \in U[0,1] \\ A(t) & otherwise \end{cases}.$$

   Set *t=t+1*.

5. Go back to step 2 and repeat until a "convergence" criterion is satisfied.

In the above algorithm, the ratio of the probabilities $r = \frac{P(\widehat{A}(t))}{P(A(t))}$ for the POMM model reduces to the ratio of only a few terms, due to Corollary 1 in Chapter 4. The ratio *r* for the fourth-order POMM model introduced earlier can be expressed as

$$r = \frac{P(\widehat{A})}{P(A)} = \prod_j \prod_i \frac{P(\widehat{a}_{ij} | \widehat{a}_{i-1,j}, \widehat{a}_{i-1,j-1}, \widehat{a}_{i,j-1}, \widehat{a}_{i+1,j-1})}{P(a_{ij} | a_{i-1,j}, a_{i-1,j-1}, a_{i,j-1}, a_{i+1,j-1})}.$$

It can also be written in more compact notation in the following form

$$r = \frac{P(\widehat{A})}{P(A)} = \frac{\prod_j \prod_i P(\widehat{a}_{ij} | N_{ij})}{\prod_j \prod_i P(a_{ij} | N_{ij})},$$

where the configuration N_{ij} is shown in Fig. 10(c). Since the image $\widehat{A}$ is the same as the image A with the gray values interchanged at locations *(i,j)* and *(h,k)*, the ratio of the joint distributions reduces to a product of the terms that involve the pixel values at locations *(i,j)* and *(h,k)*. Thus, if it is assumed that $a_{ij} = m$ and $a_{hk} = n$, then $\widehat{a}_{ij} = n$ and $\widehat{a}_{hk} = m$, and *r* reduces to

$$r = \frac{\left[P_c(\widehat{a}_{i-1,j+1}) \prod_{s=j}^{j+1} \prod_{p=i}^{i+1} P_c(\widehat{a}_{p,s}) \right] \left[P_c(\widehat{a}_{h-1,k+1}) \prod_{s=k}^{k+1} \prod_{p=h}^{h+1} P_c(\widehat{a}_{p,s}) \right]}{\left[P_c(a_{i-1,j+1}) \prod_{s=j}^{j+1} \prod_{p=i}^{i+1} P_c(a_{p,s}) \right] \left[P_c(a_{h-1,k+1}) \prod_{s=k}^{k+1} \prod_{p=h}^{h+1} P_c(a_{p,s}) \right]} \tag{18}$$

where $P_c(a_{ij}) \equiv P(a_{ij} | N_{ij})$. Thus, the ratio involves only at most ten terms each in the numerator and denominator, and hence huge numerical calculations are avoided.

5.3. Texture Synthesis Results

This section presents results of using POMM models for texture synthesis. The textures that have been generated use the auto-binomial distribution and include both binary and gray-valued textures generated with Flinn's spin-exchange algorithm [52]. The images obtained are very similar in appearance to those appearing in [56], [57], and [3], and are presented below.

The binomial model for the conditional probability of G gray levels 0, ..., G-1, is given by

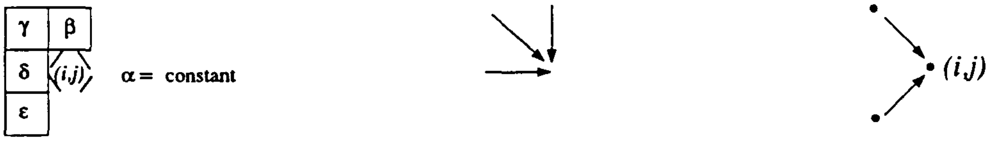
$$P(a_{ij} \mid N_{ij}) = \binom{G-1}{a_{ij}} q^{a_{ij}} (1-q)^{G-1-a_{ij}}, \quad q = \frac{e^T}{1+e^T} \tag{19}$$

where

$$T = \alpha + \beta a_{i-1,j} + \gamma a_{i-1,j-1} + \delta a_{i,j-1} + \epsilon a_{i+1,j-1} \tag{20}$$

is a combination of the values in the neighborhood of a_{ij} for the fourth order POMM model. The spatial arrangement of parameters is shown in Fig. 12. When G = 2 as for binary images, Eq. (19) reduces to

$$P(a_{ij} \mid N_{ij}) = \frac{e^{a_{ij}T}}{1 + e^{T}}.$$



(a) (b) (c)

Figure 12. POMM models used for texture synthesis. (a) Fourth-order POMM model with parameter locations; (b) Abend's model; (c) second-order model.

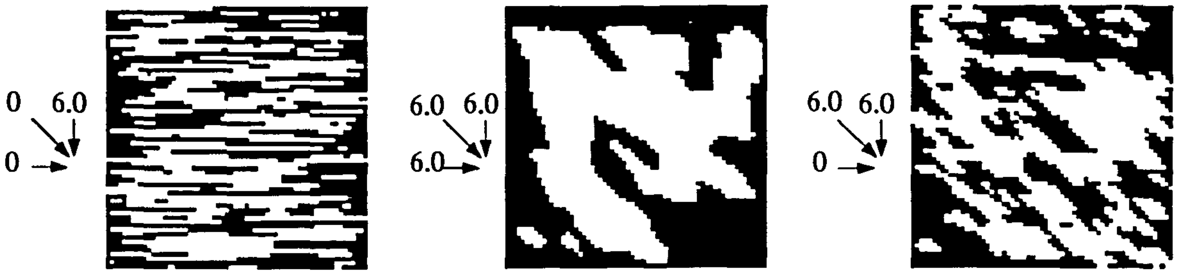
Using Flinn's algorithm, 400,000 to 800,000 iterations were performed to generate texture data, where one iteration is one calculation of a possible exchange. No stopping criterion exists for the Flinn's algorithm, as it requires, in theory, infinite iterations. Thus, after observing the exchange rates at periodic intervals, the algorithm was terminated when the rate of exchanges stabilized to a low value. For 64x64 binary images, the exchanges decreased after 400,000 to 800,000 iterations; so all the binary textures were generated using 400,000 iterations. In calculating the ratio

$$P(\{a_i\}^{new})/P(\{a_i\}^{curr}), \tag{21}$$

the expression for the joint probability of the POMM model as given in Chapter 4, Section 4.2 was used. Here, $L^0 = \{a_{11}\}$, and N_{ij} is given in Fig. 12(a). As only two values which are different are exchanged, most of the terms in Eq. (21) cancel except those containing either of the two different r.v.s considered for the exchange. Depending on the spatial relation of the two r.v.s being exchanged, Eq. 21 reduces to a ratio of 6 to 10 terms for the model in Fig. 12(a), 4 to 8 terms for Abend's model in Fig. 12(b), and 3 to 6 terms for the model in Fig. 12(c). For each texture simulated in Figs. 13 through 20, there were an equal number of pixels assigned to each gray value. Thus, for example, half the pixels for the binary case had value 0 and the remaining half had value 1.

In the results that follow, all textures were generated using one of the three partial orders given in Fig. 12. The gray values were uniformly distributed, and Flinn's algorithm was used to generate the textures. The values for the parameters accompany each texture alongside. Unless otherwise notated, the value of the constant α is 0.2 and is omitted from the diagram. After further investigation, it was found that the model in Fig. 12(c) can be subsumed by the model in Fig. 12(a) by setting $\beta = \delta = 0$. This allowed easier implementation of more textures, and this is why only a few of the textures are generated with the partial order of Fig. 12(c).

The results for the binary texture synthesis are presented next. Abend's third-order model has been used for the images in Fig. 13, the second-order POMM model as shown in Fig. 12(c) is used for the images in Fig. 14, and the fourth-order POMM for all the remaining images. The values for the parameters are given in the diagram to the left of each texture.



(a) (b) (c)

Figure 13. Binary textures generated using Abend's model.

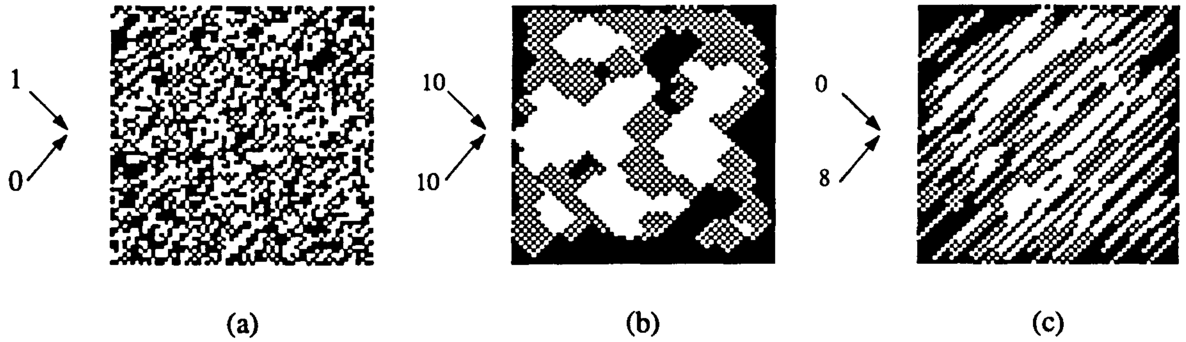


Figure 14. Binary images generated using POMM model shown in Fig. 12(c).

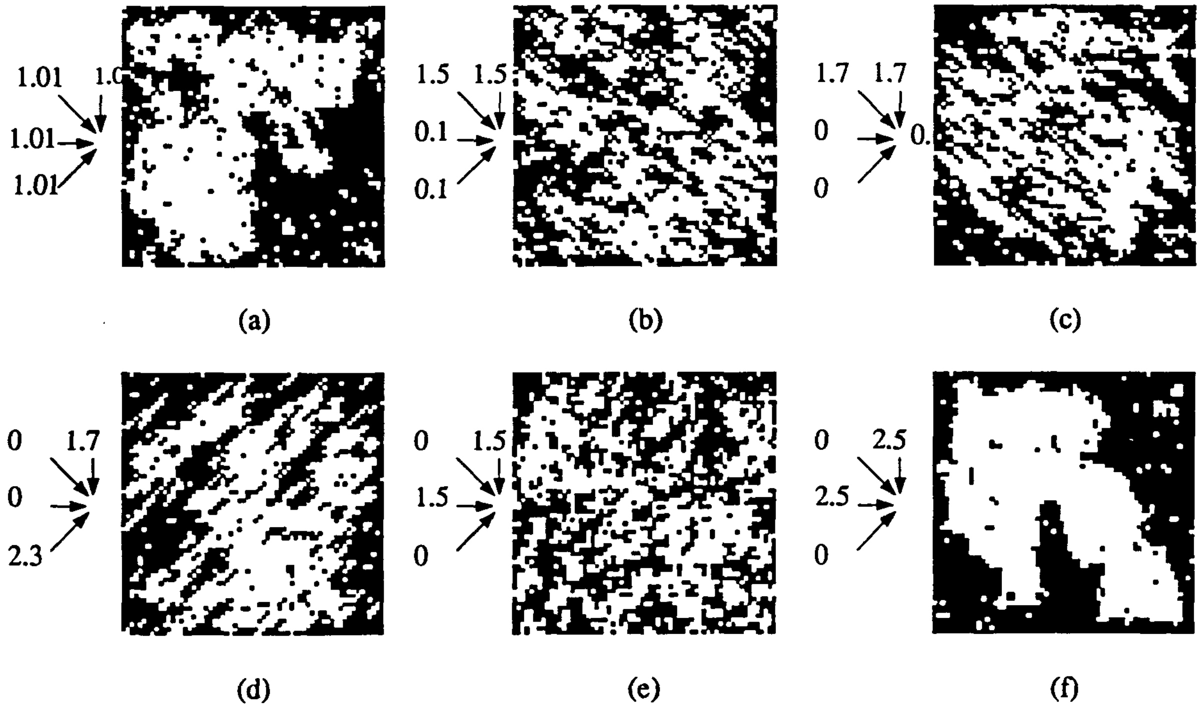


Figure 15. Binary images generated using fourth order POMM model.

In all the texture synthesis results shown in Figs. 13, 14, and 15, the images were assumed to have bordering pixels with gray values zero. This resulted in the formation of black regions at the borders. Another reasonable assumption would be to wrap the values in a toroidal manner around the boundaries. Using this different boundary condition, more textures were synthesized, and these results are shown in Fig. 16.

It is also possible to generate textures using parameters that vary spatially. For example, replacing the value T in Eq. (20) by

$$T = \alpha + \beta\frac{(i+j)}{128}a_{i-1,j} + \gamma\frac{(i+j)}{128}a_{i-1,j-1} + \delta\frac{(i+j)}{128}a_{i,j-1} + \epsilon\frac{(i+j)}{128}a_{i+1,j-1} \qquad (22)$$

introduces a dependency on location of the pixel as the values i,j are different for each pixel in the image. The POMM models using spatially varying parameters are called *heterogeneous* models. Otherwise, the models are called *homogeneous*. For example, the models that generated the images in Figs. 13–16 are homogeneous models.

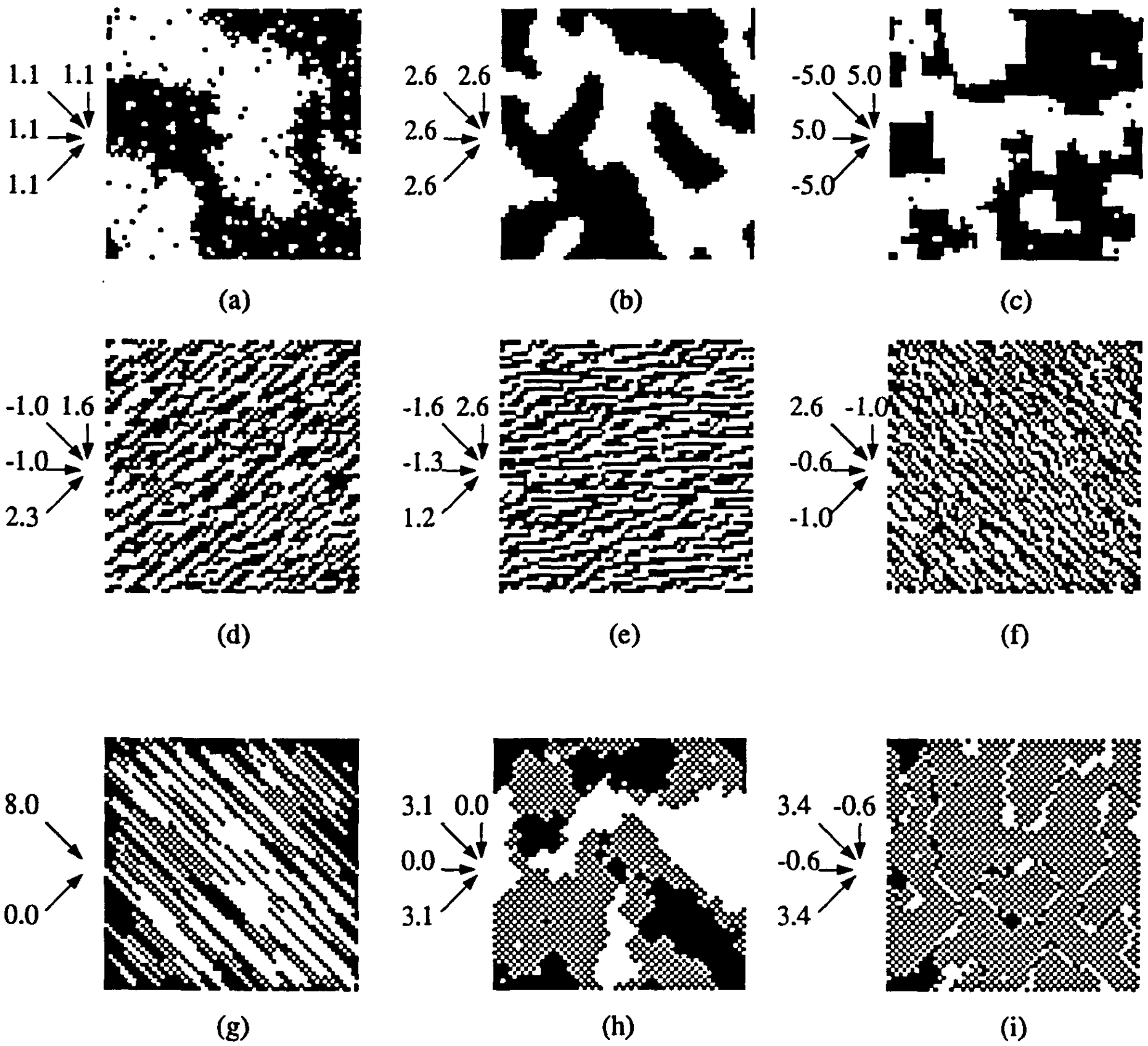


Figure 16. Binary textures generated using the previously described procedure.

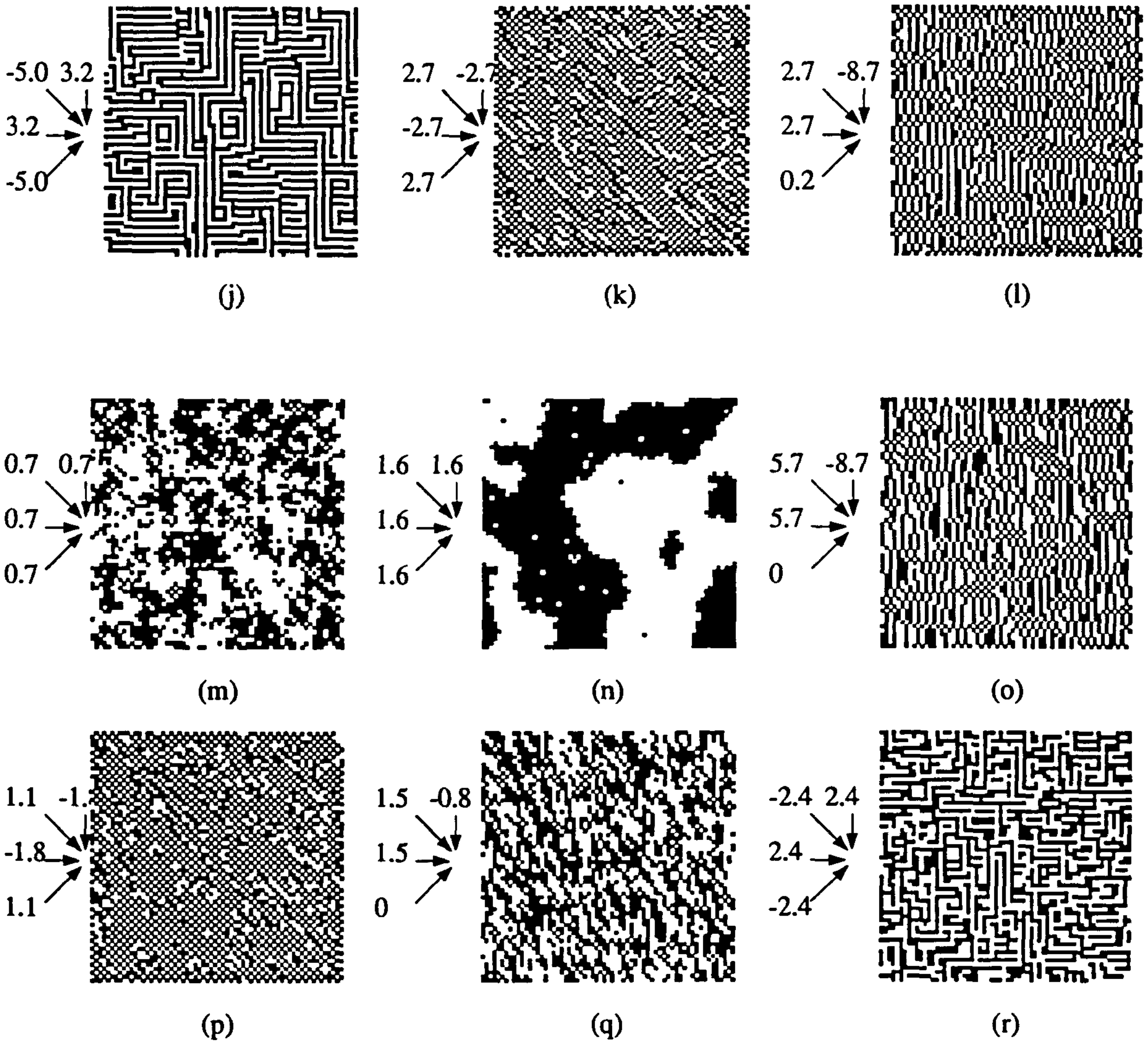
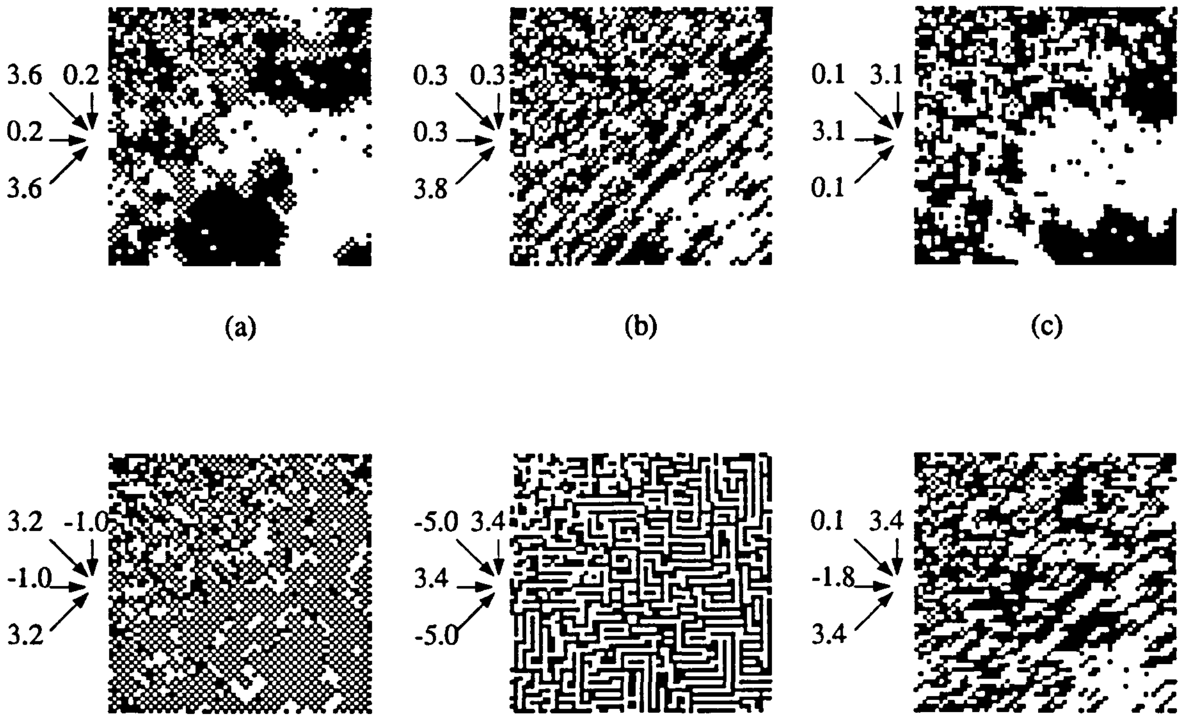


Figure 16. (continued)

In Fig. 17, binary textures were generated using the POMM model of 12(a) but with *T* as in Eq. (22). This is a heterogeneous POMM model. Note how the characteristic of the texture changes from the upper left-hand corner to the lower right-hand corner, as suggested by the specific spatial dependence in Eq. (22). Eq. (22) was used in all the heterogeneous textures shown in Figs. 17, 19, and 20, for different values of G.

In Fig. 18, textures with the four gray values 0, 1, 2, 3, were generated, using the homogeneous model in Fig. 12(a), with a binomial distribution, G = 4, equally distributed gray values, and Flinn's



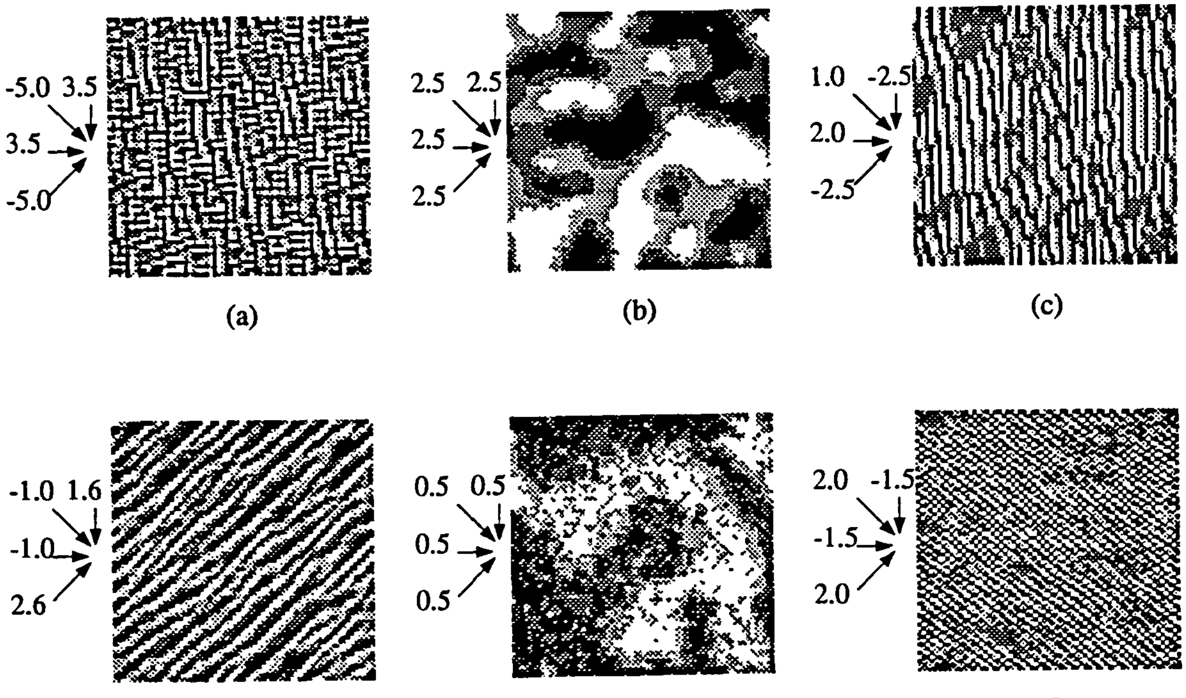
(a) (b) (c)

(d) (e) (f)

Figure 17. Binary textures generated using the heterogeneous POMM model.

iterative algorithm. In Fig. 19, textures with four gray levels and spatially varying neighborhood parameters were generated. The formula for T was the same as in Eq. (22), using Fig. 12(a).

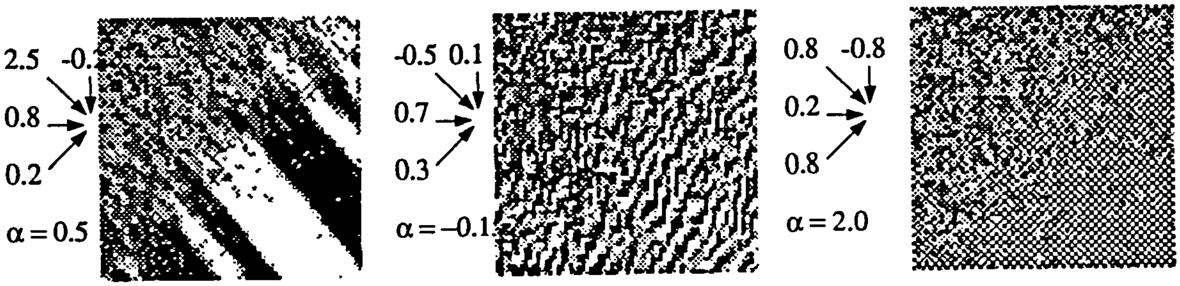
In Figs. 20, 21, and 22 textures with the sixteen gray values 0 through 15 were generated. In Fig. 20 are textures generated using a heterogeneous POMM model and Eq. (22). Fig. 21 displays textures generated with an homogeneous POMM model; the parameters are given to the left of the respective texture. Texture synthesis of sixteen gray level textures having a gray level distribution different from a uniform distribution was tried. The textures displayed in Fig. 22 were generated using the probability distribution for the gray levels as given in Table 1. Note also the effect that simply changing the parameter α can have on the resulting texture. In Fig. 22(a) α = —0.2, while in Fig. 22(b), α = —2.8. In both cases, the model in Fig. 12(a) was used as the underlying partial order.



(a) (b) (c)

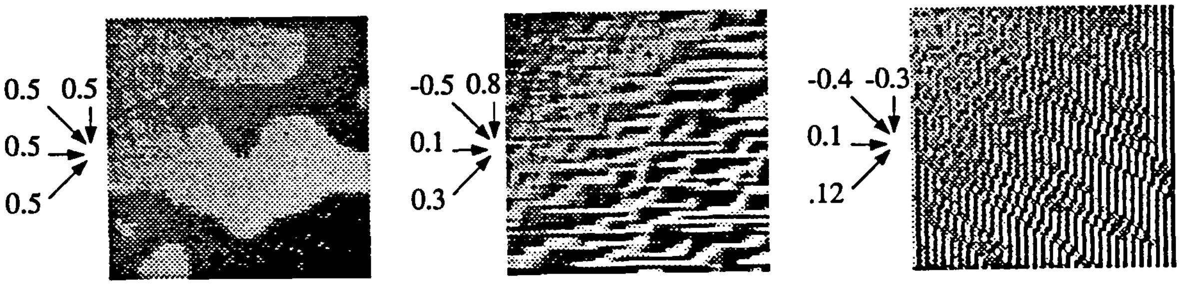
(d) (e) (f)

Figure 18. Textures generated using the homogeneous model for G = 4.



(a) (b) (c)

Figure 19. Textures generated using the heterogeneous POMM model for G = 4.



(a) (b) (c)

Figure 20. Textures generated using a heterogeneous POMM model for G = 16.

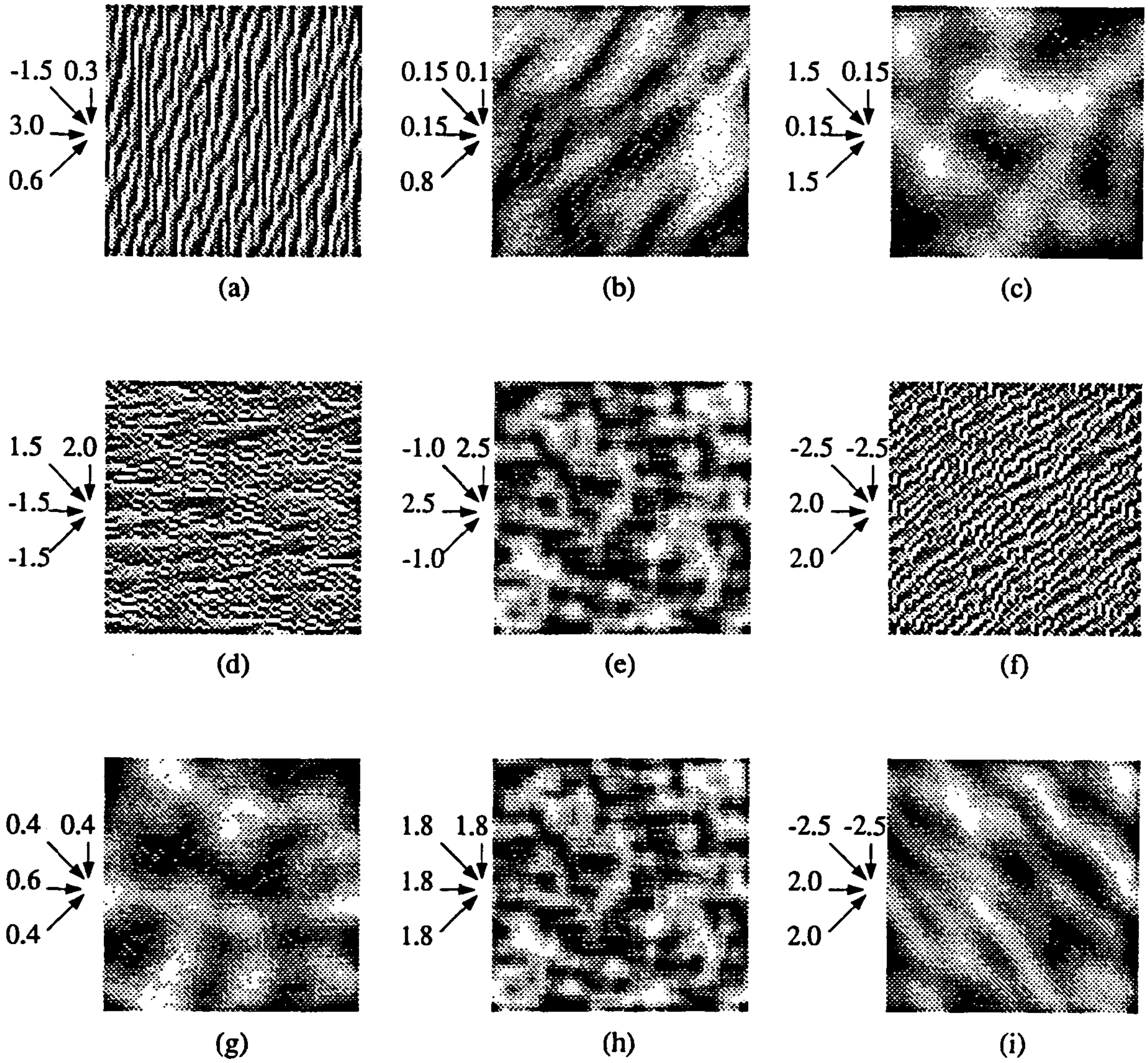


Figure 21. Textures generated using an homogeneous POMM model for G = 16.

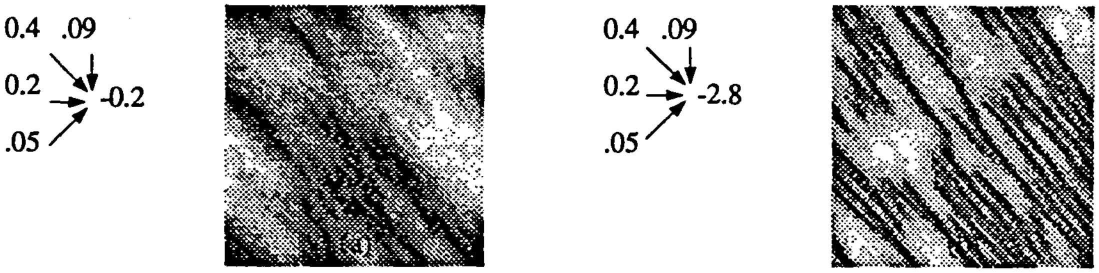


Figure 22. Textures generated using 16 gray levels having distributions as in Table 1.

Table 1. Probability of occurrence for the 16 gray levels in Fig. 22.

Gray Level	Probability	Gray Level	Probability
0	0.0258789	8	0.0924683
1	0.0689087	9	0.0525513
2	0.0551758	10	0.0653687
3	0.0704956	11	0.0789795
4	0.0605469	12	0.0588989
5	0.0574341	13	0.0770264
6	0.0613403	14	0.0600586
7	0.0703735	15	0.0444946

As can be seen, using even a simple POMM model as in Fig. 12(a), many different types of textures are possible to synthesize. The results obtained include most images obtained by other authors [56], [57], and [3] using a MRF model, and some other images *not* generated in the surveyed literature. This does not imply that other models could not generate these other textures, however. These results show that this particular fourth order model is sufficient for simulation of a variety of textures. In the next section, the inverse problem of parameter estimation from sample textures is discussed, followed by results of parameter estimation of synthetic and natural textures.

5.4. Parameter Estimation of Textures

A crucial factor in the choice of a POMM texture model is the order of the neighborhood. Once the order of the neighborhood for the POMM model is specified, the texture field is characterized by a set of parameters under stationarity assumptions. In order to use the POMM model for texture classification, it is necessary that the model parameters be estimated, if they are already not known. In applications to real data, a reasonable assumption is that only a single instance of a texture class is available. Hence, a good, efficient parameter estimation procedure is desired that will yield accurate estimates from perhaps only one sample.

A general parameter estimation problem may be expressed in the following form: Given a set of samples $A = \{A_1, A_2, A_3, ..., A_n\}$ from a distribution that is characterized by a set of parameters ψ, the goal is to find an estimate for the parameters $\widehat{\psi}$, such that it is the "best" of all possible values. The likelihood function for the set of samples is $L(\psi) = f(A_1, A_2, ..., A_n|\psi)$. "Best" in this case means an estimate that is based on some optimality criterion, and that can be implemented reliably and within reasonable computation time. Three of the most commonly used parameter estimation techniques are: Moment-based parameter estimation, Bayesian estimation, and maximum-likelihood (ML) estimation. These three estimation techniques are discussed briefly below.

Moment-based parameter estimation estimates the parameters by computing the moments of the likelihood function in terms of the model parameters. The set of equations may then be solved to obtain the estimates. A drawback of the MRF is that the moments usually cannot be expressed in closed form. At present, no work has been done on determining the moments for the POMM models. Thus, this method cannot currently be applied for parameter estimation of POMM models.

Bayesian parameter estimation assumes a prior distribution $P(\psi)$ for the set of model parameters. Then, given a set of samples, A, a "best" estimate in the Bayesian sense (for a 0–1 loss function) would be a value $\widehat{\psi}$ such that

$$P\left(\widehat{\psi}|A\right) = \max_{\psi} P(\psi|A), \text{ or, equivalently,}$$

$$P\left(A|\widehat{\psi}\right)P\left(\widehat{\psi}\right) = \max_{\psi} P(A|\psi)P(\psi).$$

In most cases, a prior knowledge of the parameters is unavailable. The Bayesian technique also fails to be suitable for our purposes.

Maximum likelihood (ML) estimation chooses the estimate $\widehat{\psi}$ such that the likelihood for the set of observations is maximized,

$$L\left(\widehat{\psi}\right) = \max_{\psi} L(\psi).$$

The maximum likelihood estimator (MLE) of any set of samples has three "desirable" statistical properties, due to which this method is widely used in most estimation problems. The properties of the ML estimator are:

1. *Consistent estimator:* $E\left(\widehat{\psi}\right) \rightarrow \psi$.
2. *Minimum variance estimator:* The MLE has the minimum variance of all possible estimates as given by the Cramer-Rao Inequality.

In many cases, the likelihood function may be unwieldy due to exponential terms. Then the log likelihood, $\bar{L}(\psi) = \ln L(\psi)$, may be used instead, since one is a monotonically increasing function of the other.

In the case of the POMMs, the parameter estimation procedure becomes greatly simplified as compared to the MRF due to the tractability of the likelihood function of the POMMs. A simple maximum likelihood estimation that would yield estimates with "desired" properties would suffice for any POMM model. For the sake of completeness, however, a brief overview of the estimation techniques for MRF models is given below.

Some of the parameter estimation procedures used for MRFs are: maximum-likelihood estimation [70], the "coding method" [41], [3], and the maximum pseudo-likelihood (MPL) estimation [71]. For the multi-level logistic (MLL) field, the estimate is not readily accessible, because of the unmanageable normalizing constant which makes the expression for the joint distribution very unwieldy. The Younes' stochastic gradient algorithm (SGA) [72] was used in [73] to yield the maximum likelihood (ML) estimate. Won and Derin [73], however, concluded that this technique is not accurate for region-type realizations. For the Gaussian Markov random fields (GMRF), the parameter space is limited due to the restriction of positive-definiteness on the covariance matrix. In this case, the normal equations can be solved to yield the maximum likelihood. However, since a GMRF exists on an infinite lattice, it is not suited for discrete images. The maximum pseudo-likelihood (MPL) estimate has been used as an alternative to the ML estimate for various classes of

GRFs. The pseudo-likelihood (PL) is the product of the local conditional probabilities over all sites in the image, i.e. if the set of model parameters are ψ, then the PL is

$$Q(A = a|\psi) = \prod_{s \in \mathbf{X}} P(a_s|\eta_s, \psi),$$

where η_s is the neighborhood of a_s in context of the MRF. The MPL estimate has been observed to be numerically close to the maximum-likelihood (ML) estimates for the "simple" Auto-normal case [74].

Another method is a coding or "coloring" of pixels in the image such that similar "colored" sites are not in the neighborhood of each other, thus partitioning the lattice into a disjoint set of points. Then a conditional ML estimation of similar colored sites can be computed, where the conditional likelihood is a product of the conditionals. A major drawback of this method is that the estimation is carried out on fewer samples for a given order of the MRF, as the larger the order, the smaller the number of samples [40], [3]. This reduction in sample size may result in poorer estimates.

5.4.1. ML Estimation for Fourth-Order POMM

The ML problem for this case can be expressed as a simple optimization problem where a $\widehat{\psi} = [\widehat{\alpha} \quad \widehat{\beta} \quad \widehat{\gamma} \quad \widehat{\delta} \quad \widehat{\epsilon}]'$ is needed such that $\tilde{L}\left(\widehat{\psi}\right) = \max_{\psi} \tilde{L}(\psi)$, where $L(\psi) = \prod_j \prod_i P(a_{ij}|N_{ij}, \psi)$, and a is a single image, or a realization of the texture. Since the conditional probabilities for the fourth-order POMM are given as

$$\begin{aligned} P(a_{ij}|N_{ij}, \psi) &= {}^{G-1}C_{a_{ij}} q_{ij}^{a_{ij}} (1 - q_{ij})^{G-1-a_{ij}} \\ q_{ij} &= \frac{e^{T_{ij}}}{1 + e^{T_{ij}}} \\ T_{ij} &= \alpha + \beta a_{i-1,j} + \gamma a_{i-1,j-1} + \delta a_{i,j-1} + \epsilon a_{i+1,j-1}, \end{aligned}$$

the log-likelihood is $\tilde{L}(\psi) = \sum_j \sum_i \ln P(a_{ij}|N_{ij}, \psi)$. Then

$$\tilde{L}(\psi) = \sum_j \sum_i \left[\ln {}^{G-1}C_{a_{ij}} + a_{ij} \ln e^{T_{ij}} - a_{ij} \ln\left(1 + e^{T_{ij}}\right) - (G - 1 - a_{ij}) \ln\left(1 + e^{T_{ij}}\right)\right],$$

which reduces to

$$\tilde{L}(\psi) = \sum_j \sum_i \left[\ln {}^{G-1}C_{a_{ij}} + a_{ij} T_{ij} - (G - 1 - a_{ij}) \ln\left(1 + e^{T_{ij}}\right)\right]. \tag{24}$$

The log-likelihood function in Eq. 24 can be optimized by applying any reasonable optimization routine. Simulated annealing has been proven to give good results when used for optimization. However, good convergence may be obtained only if a proper "cooling" schedule is used. Since this problem is not very complex, we instead investigated a **gradient descent method**, choosing initial estimates uniformly and randomly in a "reasonable" parameter space, and updating the values by

$$\psi^{(l+1)} = \psi^{(l)} - \lambda \partial \tilde{L} / \partial \psi_{|\psi=\psi^{(l)}}. \tag{25}$$

A crucial factor is the choice of the step size λ in Eq. 25, that may be scalar, a vector or a variable. A poor choice of λ results in the algorithm getting stuck at a local optimum.

Due to the tendency of the gradient descent method to converge to a local minimum, we used a variation of this algorithm called the **Newton-Raphson method** [75]. The derivation of the Newton-Raphson is as follows: The maximization problem can also be written as solving the equation $\frac{\partial \tilde{L}^*}{\partial \psi} = 0 = g(\psi)$, where $\tilde{L}^*$ is the negative of the log likelihood. By Taylor's expansion theorem,

$$0 = g(\psi_1) = g(\psi_0) + (\psi_1 - \psi_0) g'(\psi_0), \tag{26}$$

neglecting the higher order terms. By solving Eq. 26 for the desired set of parameters ψ_1 given the initial set of parameters ψ_0, convergence can be obtained by iterating in the following manner

$$\psi_1 = \psi_0 - \frac{g(\psi_0)}{g'(\psi_0)} \text{ , or}$$

$$\psi_1 = \psi_0 - \frac{1}{\left[\frac{\partial^2 \tilde{L}^*}{\partial \psi^2}\right]_{|\psi=\psi_0}} \left[\frac{\partial \tilde{L}^*}{\partial \psi}\right]_{|\psi=\psi_0} \text{, or}$$

$$\psi_1 = \psi_0 - \left[B^{-1}\right] \left[\frac{\partial \tilde{L}^*}{\partial \psi}\right]_{|\psi=\psi_0} .$$

In a more general form, the change in parameters at iteration l can be written as

$$\psi^{(l+1)} = \psi^{(l)} - \left[B^{(l)}\right]^{-1} \left[\frac{\partial \tilde{L}^*}{\partial \psi}\right]_{|\psi=\psi^{(l)}}, \tag{27}$$

where $B^{(l)}$ is the matrix whose (i,j)th element is $\left(\partial^2 \tilde{L}^* / \partial \psi_i \partial \psi_j\right)_{|\psi=\psi^{(l)}}$, and ψ_i and ψ_j are, respectively, the ith and jth elements of ψ.

5.4.2. Hypothesis Testing of Estimates

Problems of statistical inference are divided into problems of *estimation* and *tests of hypothesis.* The main difference between the two kinds of problems is that, in problems of estimation, we must determine the value of a parameter, or the values of several parameters, from a possible continuum of alternatives, whereas in tests of hypothesis, we must decide whether to accept or reject a specific value, or a set of specific values of a parameter, or those of several parameters.

Common values used as estimates are the sample mean, $\overline{Y}$, and the sample variance, S^2,

$$\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i, \text{ and}$$

$$S^2 = \frac{1}{(n-1)}\sum_{i=1}^{n} (Y_i - \overline{Y}).$$

These estimates are referred to as *point estimates.* The statistic $\overline{Y}$ is commonly used as an estimate of the mean, μ, of the population, or as a measure of a parameter estimate when several estimates are available.

Since estimators are random variables, one of the key problems of estimation is to study their sampling distributions. Various statistical properties of estimators can thus be used to decide which estimator is most appropriate in a given situation, and whether an estimator is within "acceptable" limits. In *interval estimation,* possible intervals or ranges of the estimates $\widehat{\psi}_1 < \psi < \widehat{\psi}_2$ are used to appraise the size of the error by associating a coverage probability with the interval.

These methods of analysis of estimates can be translated into the language of *statistical tests of hypothesis.* Most tests of statistical hypothesis concern the parameters of a distribution. A statistical hypothesis is defined as an assertion or conjecture about the distribution of one or more random variables. If a statistical hypothesis asserts a single distribution, it is referred to as a *simple hypothesis*; if not, it is referred to as a *composite hypothesis.* A good example of a simple hypothesis is the application of a test to determine the simple hypothesis, H_0, that the parameter of a set of samples from a distribution has a value $\theta = \theta_0$, against the alternative hypothesis, H_1, $\theta \neq \theta_0$. The alternative hypothesis *could* also be $\theta = \theta_1$, $\theta > \theta_0$, or $\theta < \theta_0$. Given a set of data, a method may be formulated to compute the value of a *test statistic* based on which possible values of the test statistic may be divided into an *acceptance region* for H_0 and a *rejection region* for H_0.

In our case, we would like to test the null hypothesis H_0: $\psi = \psi_0$ against the alternative hypothesis H_1: $\psi \neq \psi_0$. Since it appears reasonable to accept the null hypothesis when the point estimate $\widehat{\psi}$ of ψ is close to ψ_0, and to reject it when $\widehat{\psi}$ is much larger or smaller than ψ_0, it would be logical to let the rejection region or the *critical region* consist of both tails of the sampling distribution of our test statistic $\widehat{\psi}$. A representative two-tailed test for an estimate of ψ is shown in Fig. 23. The probability distribution of the estimate $\widehat{\psi}$ needs to be known in order to apply the hypothesis test.

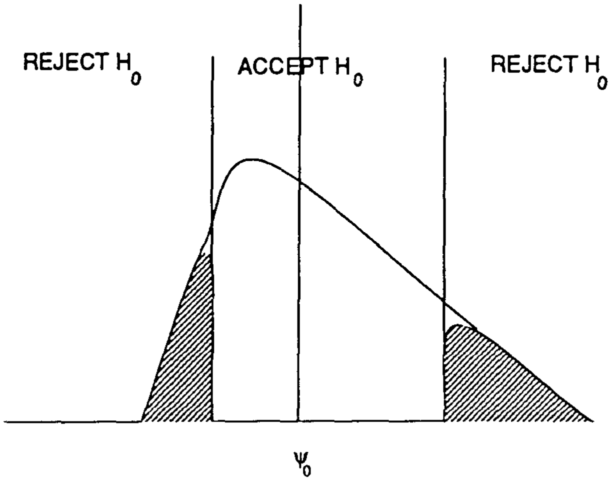


Figure 23. Two tailed test for an estimate.

The following property of the estimated mean of random, identically distributed samples has been used. If $Y_1, Y_2, .., Y_n$ are independent, identically distributed r.v.s from an infinite population with mean μ and variance σ^2, then $\overline{Y} = \frac{1}{n} \sum\limits_{i=1}^{n} Y_i$ is called the sample mean, and $S^2 = \frac{1}{(n-1)} \sum\limits_{i=1}^{n} (Y_i - \overline{Y})$ is called the sample variance. We then have $E(\overline{Y}) = \mu$ and $Var(\overline{Y}) = \frac{\sigma^2}{n}$. The *central limit theorem* states that the limiting distribution of

$$Z = \frac{\overline{Y} - \mu}{\sigma/\sqrt{n}} \; as \; n \rightarrow \infty \tag{28}$$

is the standard normal distribution. The central limit theorem also states that the convergence to a normal distribution holds true for random samples from nonnormal populations provided that the number of samples, *n*, is sufficiently large; that is, $n \geqslant 30$. Since σ is unknown, and the sample standard deviation S is a consistent estimator of σ, the value for S may be substituted for the value σ.

The knowledge of the distribution may be used for interval estimation and hypothesis testing. An interval estimate of ψ is of the form $\widehat{\psi}_1 < \psi < \widehat{\psi}_2$. Then, $P\left(\widehat{\psi}_1 < \psi < \widehat{\psi}_2\right) = 1 - \alpha$ for some specified probability $(1 - \alpha)$. For a specified value of $(1 - \alpha)$, the interval $\widehat{\psi}_1 < \psi < \widehat{\psi}_2$ is referred to as a $(1 - \alpha)100\%$ confidence interval for ψ. The quantity $(1 - \alpha)$ is called the *degree of confidence*, and the endpoints $\widehat{\psi}_1$ and $\widehat{\psi}_2$ are called the *lower* and *upper confidence limits*, respectively. The values of the estimates falling outside the confidence intervals fall in the critical region, and are the values of a null hypothesis that would be rejected.

For a normal distribution with mean μ and variance σ^2, the critical region of size α for the two-tailed test with the simple hypothesis, H_0: $\mu = \mu_0$ and the two sided alternative hypothesis H_1: $\mu \neq \mu_0$ can be written as

$$\left|\overline{Y} - \mu_0\right| \geqslant z_{\alpha/2}\frac{\sigma}{\sqrt{n}},$$

where $z_{\alpha/2}$ can be obtained from the standard normal tables. Then if

$$\mu_0 - z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \leqslant \overline{Y} \leqslant \mu_0 + z_{\alpha/2}\frac{\sigma}{\sqrt{n}},$$

the null hypothesis, H_0: $\mu = \mu_0$ is accepted; otherwise, it is rejected. The critical region is shown below in Fig. 24.

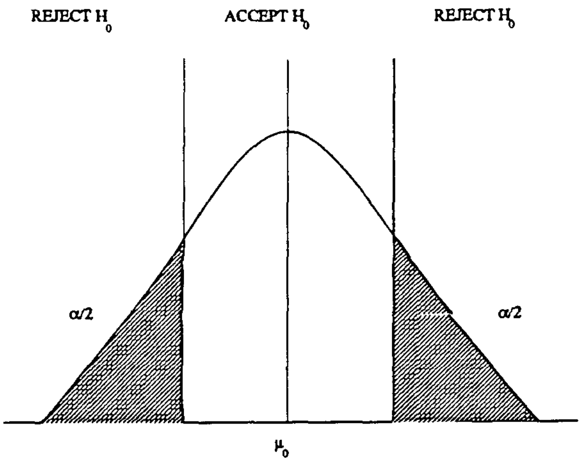


Figure 24. Confidence interval test for normal distribution.

By virtue of the central limit theorem, the method above may be used even for nonnormal samples although in that case the convergence can be quite a bit slower. Then, for any set of random samples, the null hypothesis, H_0: $\mu = \mu_0$ can be accepted if

$$\left|\overline{Y} - \mu_0\right| \leqslant z_{\alpha/2}\frac{\sigma}{\sqrt{n}}.$$

If the variance σ^2 of the distribution is not available, then the interval can be written as

$$\left|\overline{Y} - \mu_0\right| \leqslant z_{\alpha/2}\frac{S}{\sqrt{n}}. \tag{29}$$

A 95% confidence interval with a corresponding value of $\alpha = 0.05$ is commonly used. The corresponding value of $z_{\alpha/2}$ is 1.96.

For hypothesis testing of the feasibility of the estimates of the fourth-order POMM model, the problem can be formulated as follows: Given a set of images $A_1, A_2, .., A_n$ representative of a single texture with an ideal set of parameters, $\psi_a = [\alpha_a,\ \beta_a,\ \gamma_a, \delta_a,\ \epsilon_a]'$, suppose their corresponding estimates are $\psi_1, \psi_2, .., \psi_n$ and the mean of the estimates $\overline{\psi} = \sum_{i=1}^{n} \psi_i$. The null hypothesis H_0: $\psi = \psi_a$ is accepted if ψ_a is within a 95% confidence interval about $\overline{\psi}$, or

$$\left|\overline{\psi} - \psi_a\right| \leqslant z_{0.025}\frac{S}{\sqrt{n}},$$

where S is the standard deviation of the estimates and n is large (we chose $n = 100$ below.) The critical region is then

$$\begin{aligned} \psi_a &\leqslant \overline{\psi} - z_{0.025}\frac{S}{\sqrt{n}} \text{ and },\\ \psi_a &\geqslant \overline{\psi} + z_{0.025}\frac{S}{\sqrt{n}}, \end{aligned} \tag{30}$$

and is shown below in Fig. 25.

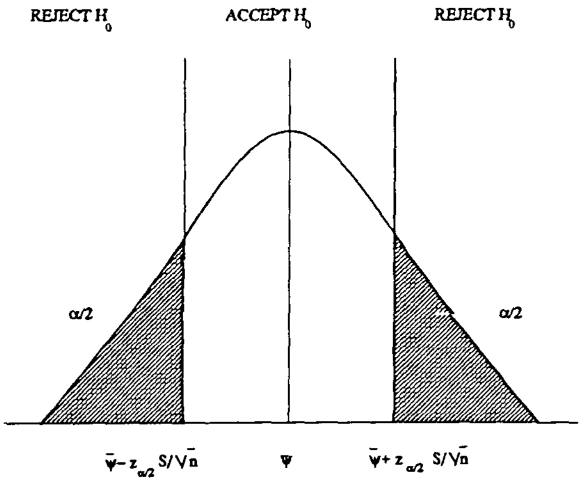


Figure 25. Confidence interval test for parameter estimates.

The hypothesis test of the estimates could be used for two purposes. First, one could evaluate whether the POMM is suitable for textured images; and secondly, one could give an idea of the suitability of specific POMM models to be used for different classes and types of textures. We use the hypothesis test for testing the validity of the estimates in the next section.

5.5. Parameter Estimation Results

In this section, the parameter estimation results on both synthetic and natural images are presented. The method for computing the estimates of the parameters was maximum likelihood estimation, explained in the previous section. To search the parameter space, two different methods were used: *gradient descent* and the *Newton-Raphson* algorithm. In all cases, convergence to the estimates via the Newton-Raphson method was faster than and as equally accurate as those found using the gradient descent method.

In order to test the validity of the results, the synthetic images generated in Section 5.3 were used, as their parameters are already known. The fourth-order POMM model shown in Fig. 26 was used. The results obtained on the estimation of synthetic binary images generated in Section 5.3 are shown below in Table 2. The actual parameter values are given first, followed by the corresponding estimated parameters.

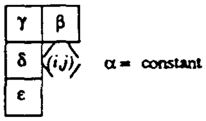


Figure 26. Parameters of the fourth-order POMM that describe the textures.

In order to evaluate the "goodness" of the estimates, the hypothesis test proposed in the previous section was tried on a set of binary textures with the same parameters. A series of 100 synthetic images were generated using the parameters shown in Fig. 16(j) with different initial images. The ML estimates, $\psi_1, \psi_2, ..., \psi_{100}$, of each of these images was calculated using the Newton-Raphson method. The plot of the estimates of each parameter α, β, γ, δ, and ϵ is shown below in Fig. 27. The sample mean $\overline{\psi}$ and the sample variance S of the estimates were computed as

$$\overline{\psi} = \frac{1}{100} \sum_{i=1}^{100} \psi_i \text{ , and}$$

$$S^2 = \frac{1}{99} \sum_{i=1}^{100} \left(\psi_i - \overline{\psi}\right)^2,$$

Table 2. Parameter estimation of synthetic binary images.

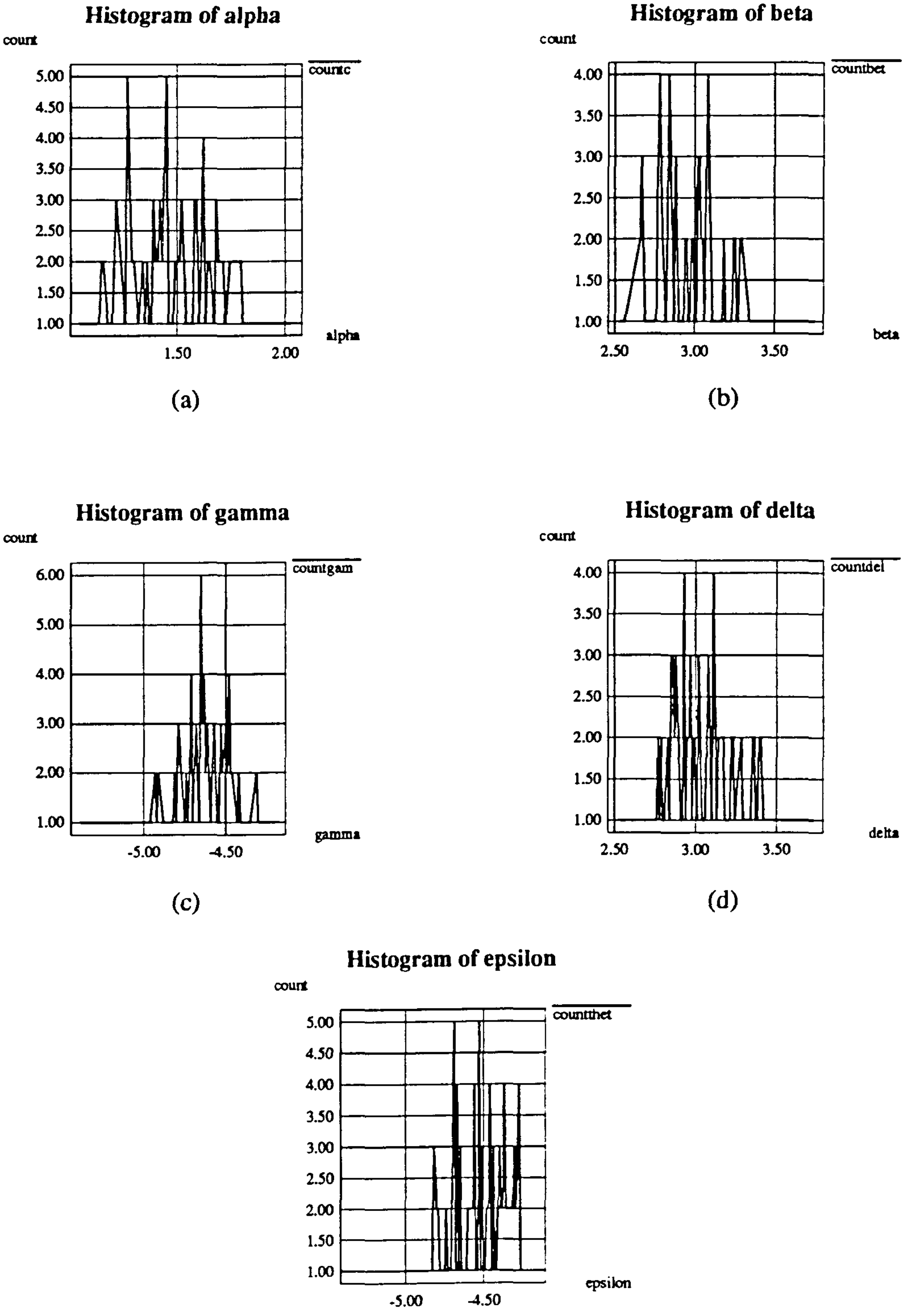
ACTUAL PARAMETERS					ESTIMATED PARAMETERS					Corresponding image
α	β	γ	δ	ϵ	$\widehat{\alpha}$	$\widehat{\beta}$	$\widehat{\gamma}$	$\widehat{\delta}$	$\widehat{\epsilon}$	
0.2	0	3.1	0	3.1	-3.14	-0.12	3.36	-0.07	3.26	Fig. 16(h)
0.2	2.6	2.6	2.6	2.6	-4.02	2.93	1.17	1.84	2.3	Fig. 16(b)
0.2	1.1	1.1	1.1	1.1	-2.81	1.61	1.33	1.4	1.46	Fig. 16(a)
0.2	0.7	0.7	0.7	0.7	-1.61	0.87	0.83	0.73	0.83	Fig. 16(m)
0.2	-2.7	2.7	-2.7	2.7	0.21	-1.8	2.52	-2.7	1.63	Fig. 16(k)
0.2	-0.8	1.5	1.5	0	-1.08	-0.67	1.48	1.43	-0.09	Fig. 16(q)
0.2	-8.7	5.7	5.7	0	0.55	-11.4	5.4	7.88	-0.49	Fig. 16(o)
0.2	2.4	-2.4	2.4	-2.4	0.35	2.3	-2.78	2.45	-2.5	Fig. 16(r)
0.2	-1.0	2.6	-0.6	-1.0	0.12	-1.07	2.45	-0.57	-0.99	Fig. 16(f)
0.2	1.6	1.0	1.0	2.3	-0.82	1.5	-1.0	-1.17	2.39	Fig. 16(d)
0.2	2.6	-1.6	-1.3	1.2	-0.37	2.69	-1.63	-1.26	0.97	Fig. 16(e)
0.2	-1.8	1.1	-1.8	1.1	0.46	-1.79	1.31	-1.71	0.98	Fig. 16(p)

and the results are shown below in Table 3. The synthetic images generated with the actual parameters and sample mean are shown in Fig. 28. The 95% confidence interval test with α= 0.05 was carried out on the sample mean $\overline{\psi}$. For this value of α, the confidence region is $\psi_a \leqslant \overline{\psi} + z_{0.025}\frac{S}{\sqrt{100}}$ and $\psi_a \geqslant \overline{\psi} - z_{0.025}\frac{S}{\sqrt{100}}$. Substituting $z_{0.025} = 1.96 \approx 2$, we get the confidence region for the 95% confidence interval test to be

$$\psi_a \leqslant \overline{\psi} + 2 \times \frac{S}{\sqrt{100}} \text{ and}$$

$$\psi_a \geqslant \overline{\psi} - 2 \times \frac{S}{\sqrt{100}}.$$

The actual parameter values and their confidence intervals are shown in Table 3. Note that none



(a)

(b)

(c)

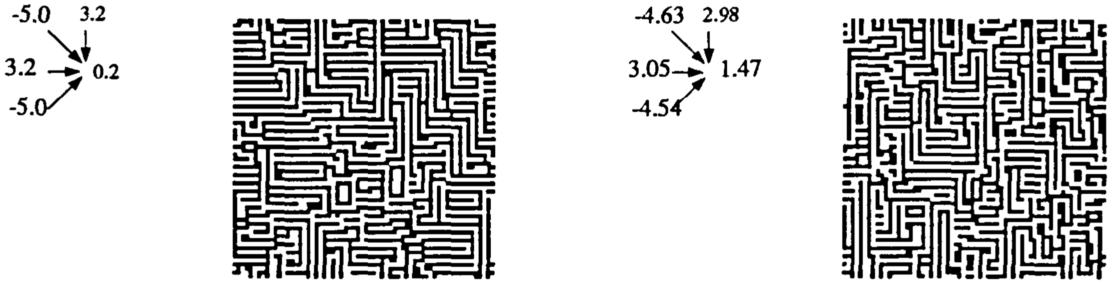
(d)

(e)

Figure 27. Histogram of the estimates of Fig. 28(a). (a) α (b) β (c) γ (d) δ (e) ϵ.

Table 3. Estimated parameters and confidence intervals.

Parameters	Actual parameter values	Estimated parameter values (mean)	Variance of estimated parameters	Confidence interval limits	
				Lower limit	Upper limit
α	-0.2	1.473	0.04	1.43	1.51
β	3.2	2.986	0.0475	2.94	3.00
γ	5.0	-4.636	0.0382	-4.68	-4.60
δ	3.2	3.06	0.052	3.01	3.11
ϵ	5.0	-4.536	0.0372	-4.58	-4.50



(a) (b)

Figure 28. (a) Original synthetic image (b) Image from mean of the estimates.

of the parameter estimates are within the acceptable limits. An image generated with the estimated sample mean shown in Fig. 28(b) is, however, observed to be visually similar to an image generated using the actual parameter values shown in 28(a).

A possible reason for the failure of the hypothesis test is the fact that the gray level distribution of the generated images is a *uniform distribution*, whereas the underlying conditional distribution of the conditional probability of each r.v. is a *binomial distribution* which would lead to a highly nonuniform marginal distribution of pixel values. This "fitting" of binomially distributed r.v.s to an uniformly distributed image *may* lead to unpredictable results. A possible solution would be use a binomial distribution for the gray levels in the image. Since in the expression for the conditional probability of a r.v.,

$$P(a_{ij}|N_{ij}) = \frac{e^{a_{ij}T_{ij}}}{1+e^{a_{ij}T_{ij}}}, \text{ where}$$

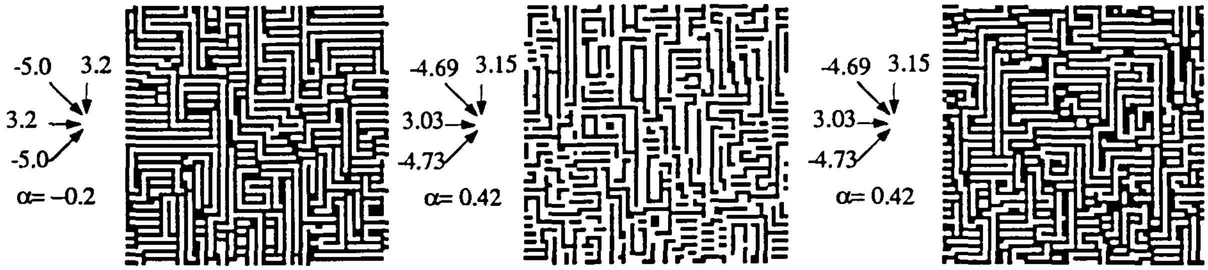
$$T_{ij} = \alpha + \beta a_{i-1,j} + \gamma a_{i-1,j-1} + \delta a_{i,j-1} + \epsilon a_{i+1,j-1},$$

only the parameter α does not represent a dependence on neighboring pixels, and hence could be used to determine an approximation to the gray level distribution in the image. Given a set of parameters, the gray level distribution in the image could be approximated by assuming $\beta = \gamma = \delta = \epsilon = 0$, and then the probability of each possible gray value for the specified value α, or $P(a_{ij} = m) = \frac{e^{m\alpha}}{1+e^{\alpha}}$, $m = 0, 1, .., G-1$, can be computed. Thus, the initial image may be generated by assuming the initial image to be composed of independent, identically distributed (i.i.d) r.v.s, each having a gray value with a probability $P(a_{ij} = m) = \frac{e^{m\alpha}}{1+e^{\alpha}}$, $m = 0, 1, .., G-1$.

In order to test this technique, a series of 100 binary images was synthesized with a value of $\alpha = -0.2$, with a corresponding gray level distribution of $P(a_{ij} = 0) = 0.55$, and $P(a_{ij} = 1) = 0.45$. The other parameter values chosen are shown below in Fig. 29(a) and Table 4. The ML estimates were computed and a hypothesis test carried out on these estimates similar to the technique discussed previously. The sample mean, variance, and the 95% confidence intervals are given in Table 4. The hypothesis test results were much better as compared to the results obtained when using a uniform distributed binary image as shown previously in Fig. 3. All the estimates, especially the parameters α, β, and δ were closer to the 95% confidence intervals as compared to the method in Table 3, even though they still were not within the 95% confidence limits. A synthetic image was generated with the mean of the estimates shown in Table 4, with a gray level distribution $P(a_{ij} = 0) = 0.4$, and $P(a_{ij} = 1) = 0.6$, corresponding to $\alpha = 0.42$. Due to the difference in gray level distributions between the texture generated using the actual parameters with $\alpha = -0.2$ and the texture generated using the estimated parameters with $\alpha = 0.42$, the two textures shown in Fig. 29(a) and (b) are visually different from each other. However the image generated with the sample mean

Table 4. Estimated parameters and confidence intervals.

Parameters	Actual parameter values	Estimated parameter values (mean)	Variance of estimated parameters	Confidence interval limits	
				Lower limit	Upper limit
α	-0.2	0.418	0.03	0.38	0.45
β	3.2	3.147	0.0331	3.1	3.18
γ	5.0	-4.686	0.0289	-4.72	-4.65
δ	3.2	3.033	0.032	2.99	3.07
ϵ	5.0	-4.734	0.0349	-4.77	-4.70

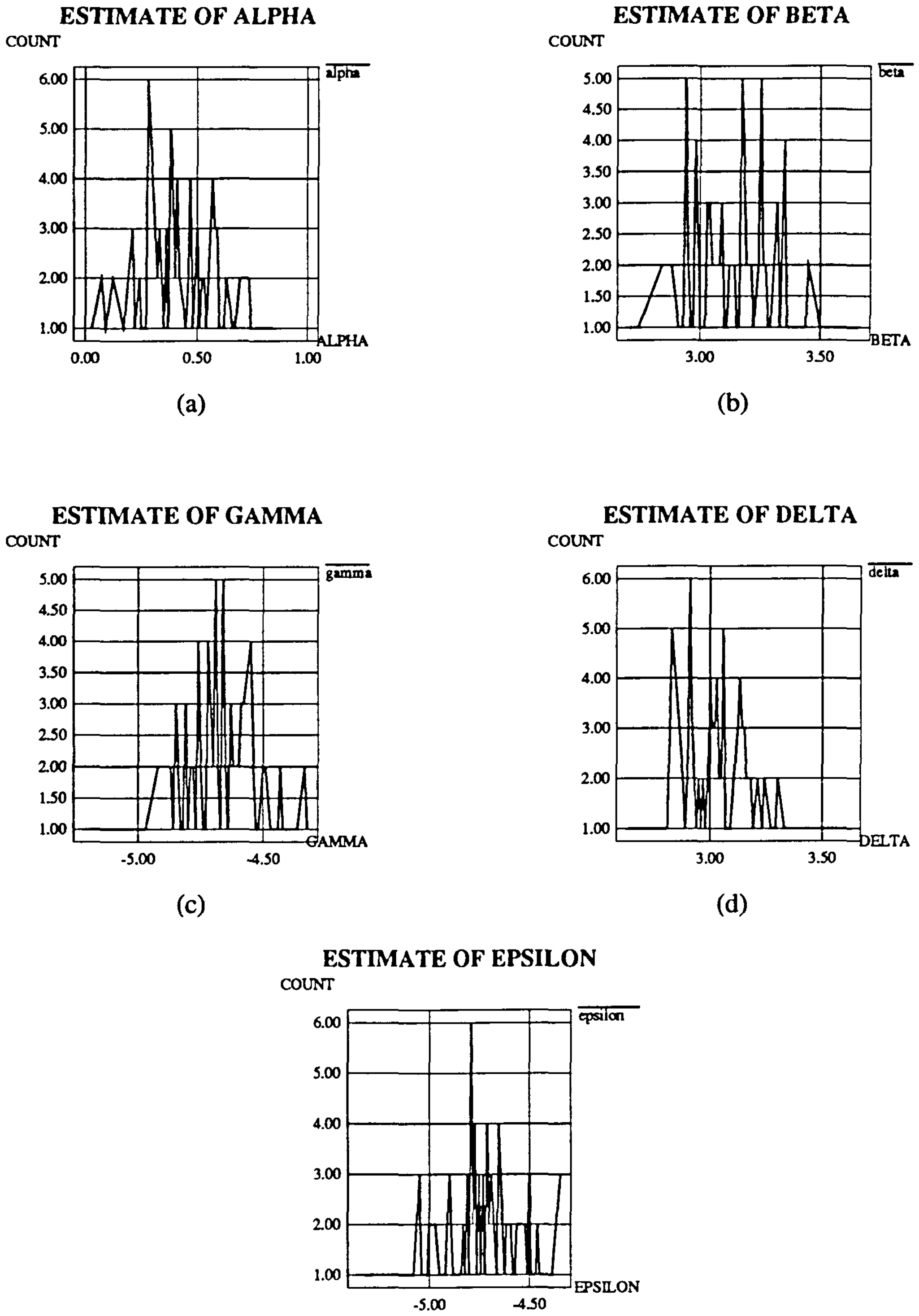


(a) (b) (c)

Figure 29. (a) Original synthetic image, (b) Image from sample mean with distribution for α= 0.42, (c) Image from sample mean with distribution for α= —0.2.

with distribution for α= —0.2 shown in Fig. 29(c) is visually close to the original image. The plot of the parameter estimates are shown in Fig. 30.

Since the method discussed previously yields synthetic estimated images that are not visually close to the original images due to the difference in their gray level distributions, an alternative solution to the problem of estimation is given next. Given an image, a value for α may be found that satisfies the gray level distribution of the image. After estimating α in this manner, the remaining parameters β, γ, δ, and ϵ may be estimated by the method of maximum likelihood. This technique yields a synthetic estimated image that has the same gray level distribution as the original image. We tried this approach on the same set of images used in the previous technique with $\alpha = -0.2$ corresponding to the distribution $P(a_{ij} = 0) = 0.55$, and $P(a_{ij} = 1) = 0.45$. In effect,



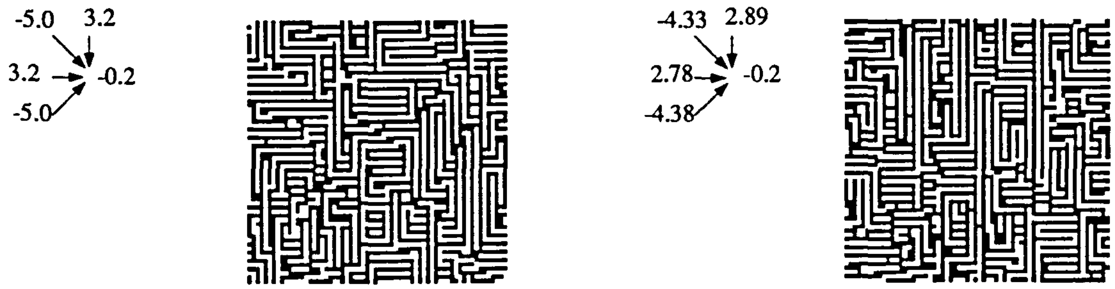
(a) (b) (c) (d) (e)

Figure 30. Histogram of the estimates of Fig. 29(a). (a) α (b) β (c) γ (d) δ (e) ϵ.

we *assigned* $\alpha = -0.2$ corresponding to the gray level distribution, and then estimated the other POMM parameters. However, it should be noted that the problem of a fixed gray level distribution still exists since the Flinn's algorithm is used to generate the synthetic images. The actual parameters, the sample mean and variance of the estimates, and the 95% confidence intervals are given in Table 5. The estimates are better than the estimates obtained from the uniform distributed images presented in Table 3, but not as good as the results obtained in Table 4. A sample image with the actual parameters and the synthetic image generated from the corresponding estimates is shown in Fig. 31. The plot of the parameter estimates are shown in Fig. 32.

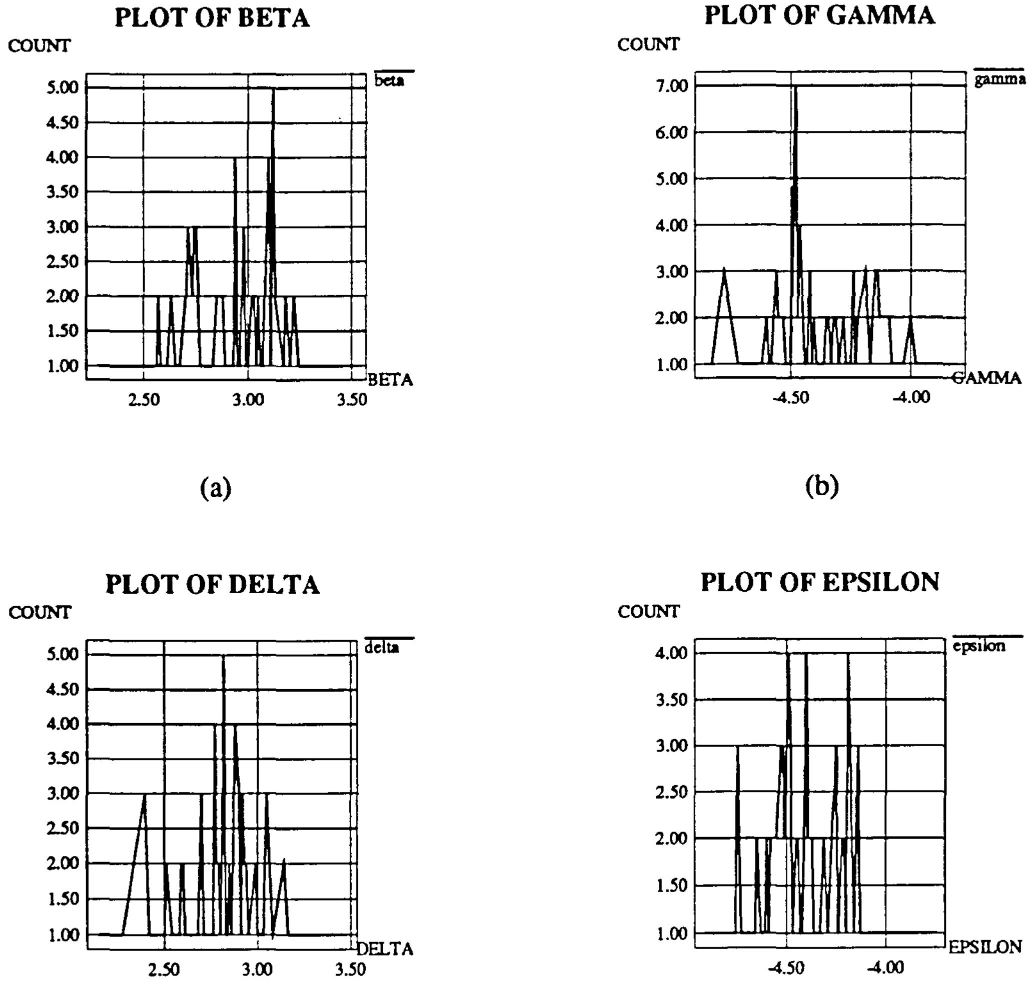
Table 5. Estimated parameters and confidence intervals.

Parameters	Actual parameter values	Estimated parameter values (mean)	Variance of estimated parameters	Confidence interval limits	
				Lower limit	Upper limit
α	-0.2	-0.2	0	-0.2	-0.2
β	3.2	2.887	0.0729	2.83	2.94
γ	5.0	-4.338	0.0625	-4.4	-4.28
δ	3.2	2.776	0.0676	2.73	2.83
ϵ	5.0	-4.386	0.0630	-4.44	-4.34



(a) (b)

Figure 31. (a) Original synthetic image (b) Image from mean of the estimates.



(a) (b)

(c) (d)

Figure 32. Histogram of the estimates of Fig. 31(a). (a) α (b) β (c) γ (d) δ (e) ϵ.

Further investigation needs to be done on the effects of imposing a fixed gray level distribution in the texture synthesis process. A better texture synthesis technique can be formulated that does not restrict the gray level distribution.

Next, the estimation algorithm was tested on sixteen gray level synthetic images from Section 5.5 with known parameters. The results are shown below in Fig. 34 and 33. For each set of images, the original images are on the left, and the estimated images on the right, with the corresponding parameters shown at the left of each image. Unless specified, the value of the constant $\alpha = 0.2$.

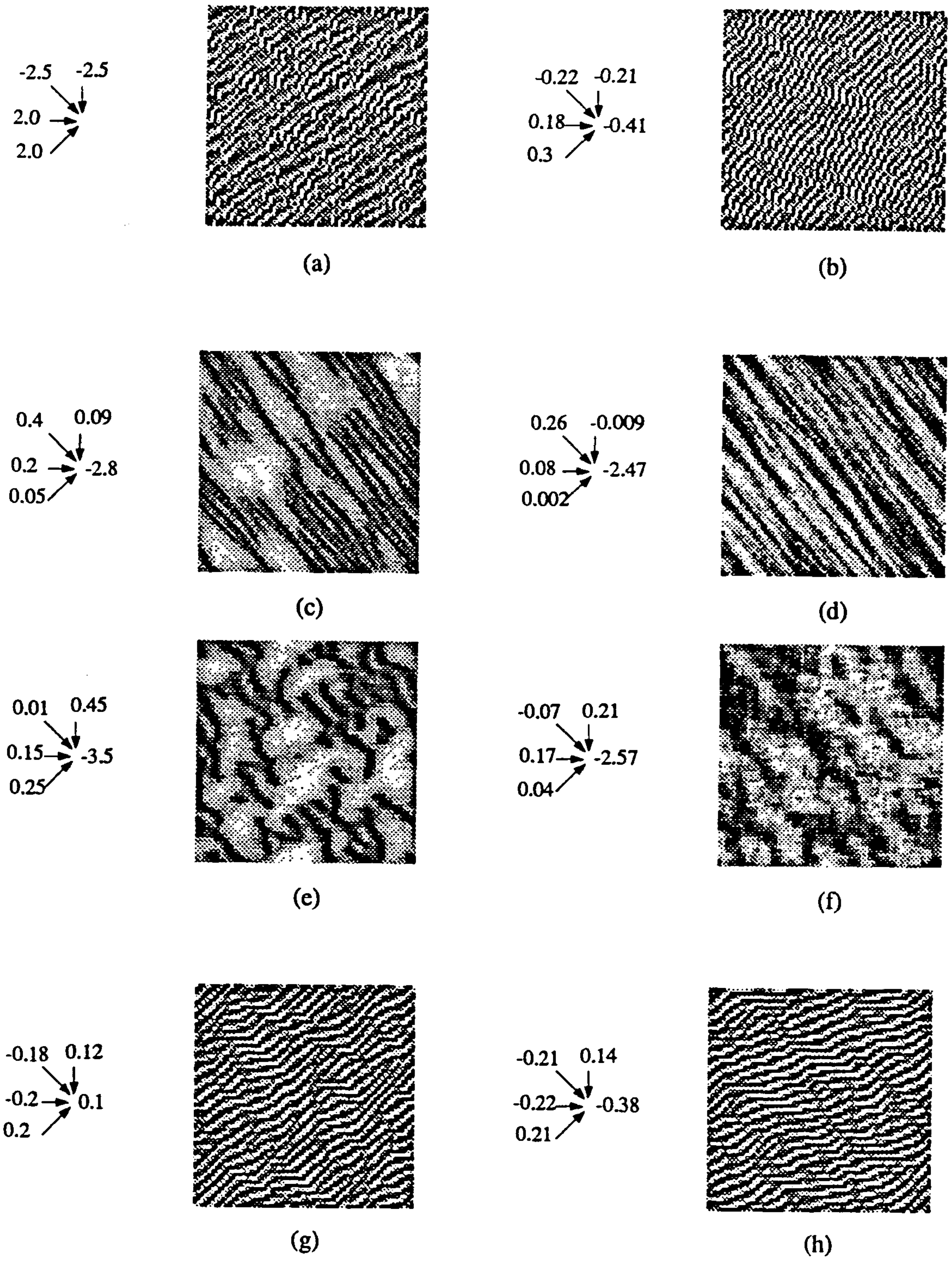


Figure 33. Synthetic 64x64 image and its corresponding regenerated image estimates on the right.

Parameter estimation was also performed on natural Brodatz textures [76]. It was observed that the MLE parameter estimation gave better results when the sample size was larger, or equivalently, when the image size was bigger. The estimation results on the Brodatz textures of a cloud, water, tree, grain, and cork are shown below. Fig. 35 displays results for image data of size 64x64, while Fig. 36 displays results for image data of size 128x128.

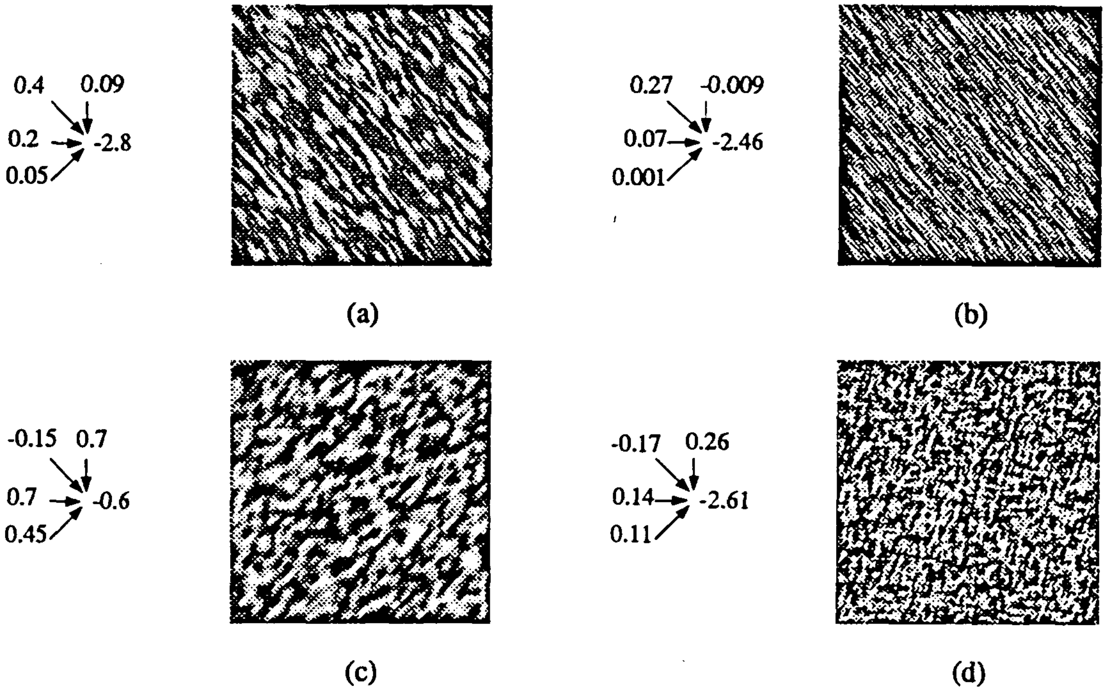
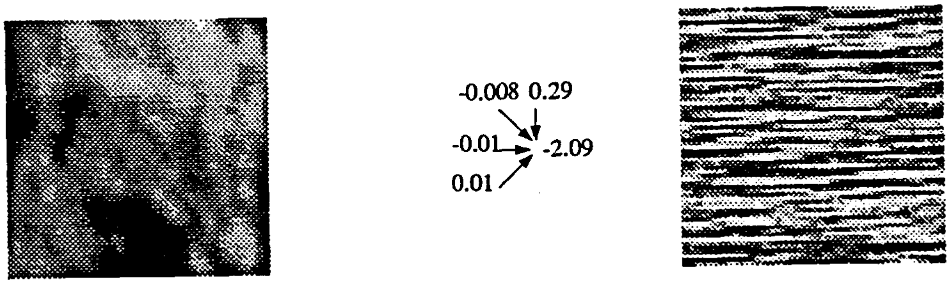
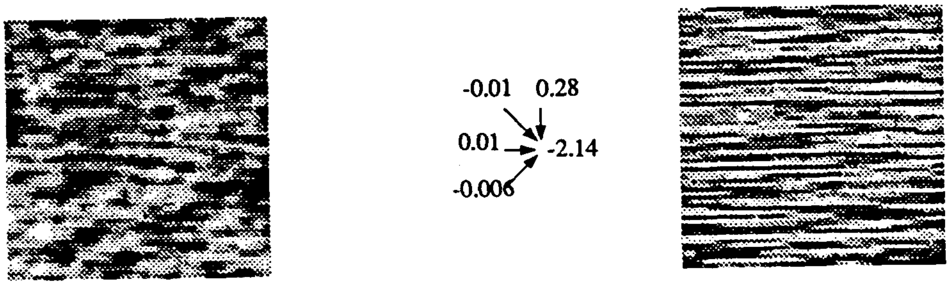


Figure 34. Synthetic 256x256 image and its corresponding regenerated image on the right.

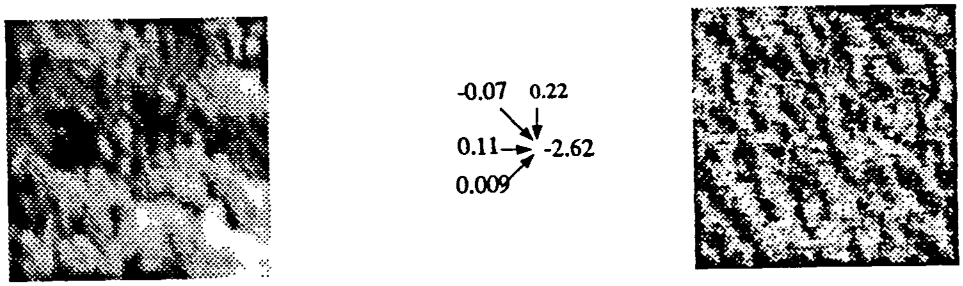


(a) Brodatz cloud image (b)

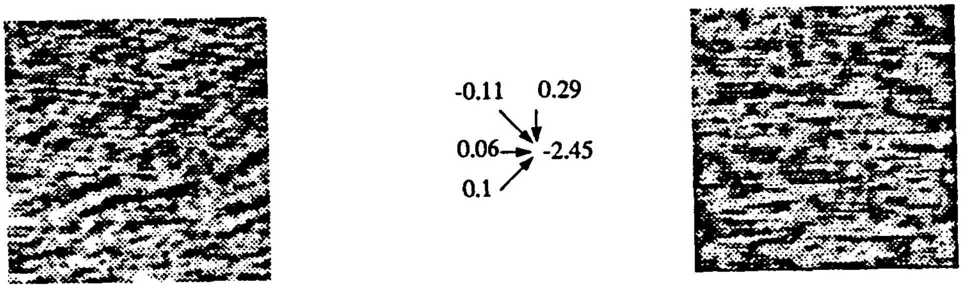


(c) Brodatz water image (d)

Figure 35. 64x64 Brodatz image, and the regenerated synthetic texture from estimated parameters.

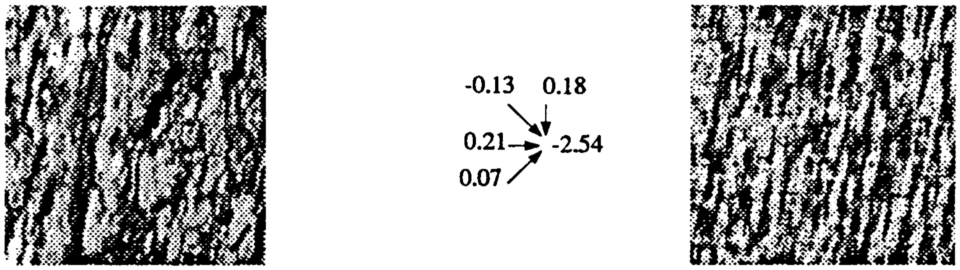


(a) Brodatz cloud image (b)

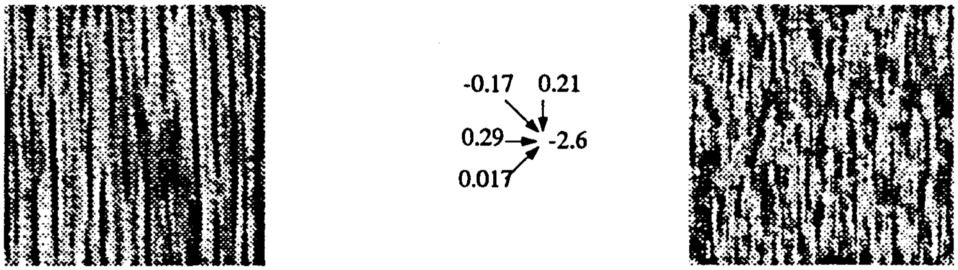


(c) Brodatz water image (d)

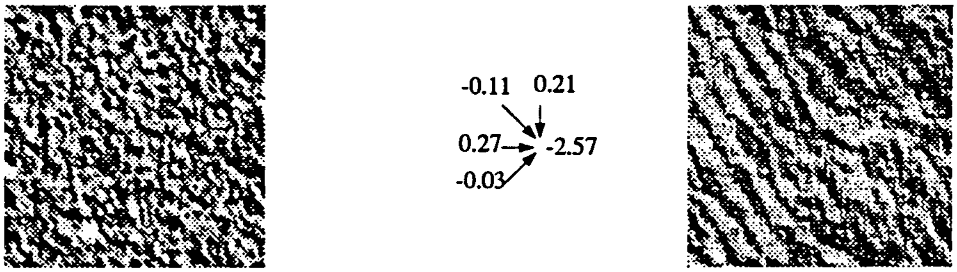
Figure 36. 128x128 Brodatz texture, and the regenerated synthetic texture from estimated parameters.



(e) Brodatz tree image (f)



(g) Brodatz grain image (h)



(i) Brodatz cork image (j)

Figure 36. (continued)

The ML parameter estimates gave good visual results on several of the natural textures, with some of the synthesized images visually close to the original textures. Also, note that estimates were closer for binary images as compared to the sixteen gray level images. A possible reason may be due to the fact that the likelihood function for the gray level case involves a more complex search space than for the binary case. The statistical analysis of the parameter estimates for the binary image showed that the estimates were not within "acceptable" limits, and that a proper choice of α gives better estimates as shown in Table 5. So far, analysis of images generated using the binomial marginal to compute the gray level distribution have given the best results as shown in Table 4. More investigation needs to be done on the role of the parameter α in the POMM model, and its influence on the gray level distribution in the image. More important, a texture synthesis technique that does not limit the gray level distribution needs to be formulated. A better parameter estimation technique needs to be applied for this particular model. Since the likelihood function is not complex, this POMM model may be used with other, more efficient, estimation techniques.

In the next chapter, the problem of texture segmentation is investigated, where the parameter values of individual textures have been estimated in advance using ML estimation. The estimated parameters provide an initial segmentation of the unknown image. Then a maximum a posteriori technique is applied to obtain "better" segmentation according to a Bayesian criterion.

6. SIMULATED ANNEALING FOR TEXTURE SEGMENTATION AND TEMPLATE IDENTIFICATION

In the previous chapter, a parameter estimation technique was used to estimate the parameters of natural images modeled by a fourth-order POMM. When an image consists of several textures, region classification cannot be obtained merely by knowledge of the representative texture parameters. Regions must first be identified and then subjected to a classification process. In this chapter, the problem of texture segmentation is viewed as a *maximum a posteriori* (MAP) problem that may be solved iteratively by simulated annealing. The theory and background of simulated annealing as an optimization technique is given in Section 6.1. In Section 6.2, a texture segmentation technique using simulated annealing is detailed, and the results obtained on the segmentation of natural images is presented in Section 6.3. Section 6.4 presents the simulated annealing optimization procedure as a learning rule for a morphology neural network (MNN), a relatively new neural network. Results of applying the MNN to solve a template identification problem are given.

6.1. Simulated Annealing: Theory and Applications

An optimization problem is concerned with finding a good or even the best configuration or parameter set from a large set of feasible options. When the configurations are elements from a very large but finite set, the respective problem is called a combinatorial problem. *Simulated annealing* is a probabilistic algorithm for approximately solving large combinatorial optimization problems. All optimization problems have an associated *cost function* that must be maximized or minimized. The cost function is a function of many variables and represents the quantitative "goodness," with respect to a certain criteria, of the set to be optimized.

Iterative improvement is a well known heuristic method for solving combinatorial optimization problems. The algorithm starts with the system in a given state, and a corresponding cost, and iteratively proceeds to new states that have lower costs. The inherent limitation of the iterative improvement algorithm is that it usually gets stuck in a local optimum. It is therefore not suited for complex optimization problems.

Simulated annealing is an optimization technique that allows "cost increasing" changes in the state of the system, thereby enabling the algorithm to jump out of local optima. Simulated annealing (SA) [77] is a simulation of the physical process of annealing whereby a solid is slowly cooled starting at a high temperature so that its molecular structure becomes "frozen" at a "minimum energy configuration." The basic idea in this process of stochastic relaxation or statistical cooling is to change provisionally a single randomly selected part of the system at a time. If this change results in a lower global system energy or "cost," then the change is accepted as the new state. If the change results in an increase in global energy, then this change is accepted with a probability p given by

$$p = e^{\frac{-\Delta E}{T}} = e^{\frac{-\Delta g}{T}}.$$

This is known as the Metropolis acceptance criterion [78]. The quantity $\Delta E = E^{new} - E^{old} = \Delta g = g^{new} - g^{old}$ is the change in energy or "cost" and T is the temperature. The acceptance of deteriorations or possibly higher costs in the system prevents the system from getting stuck in a local minimum. As the process proceeds, the temperature value is decremented so that probabilities for accepting system deteriorations becomes smaller until the probability approaches zero.

The simulated annealing algorithm was designed to find an optimal solution to a continuous system of states. By making appropriate modifications to the problem statement, a discrete version of this algorithm can be obtained which can be applied to combinatorial optimization problems [79].

The algorithm can be mathematically described as the generation of a sequence of homogenous Markov chains [78]. Since the theory of homogenous Markov chains is well established, it is possible to analyze the convergence of this algorithm by studying the Markov chain describing the process. It has been proven [80] that if the "cooling schedule" or decrements in temperature is "properly" implemented, and the system transitions are done an infinite number of times, then the algorithm converges with uniform probability to the set of globally optimum states. Equivalently, the stationary distribution of the Markov chain describing the process is given as

$$\lim_{C_k \downarrow 0} P(\omega) = \begin{cases} |\mathcal{R}_{opt}|^{-1} & if\ \omega \in \mathcal{R}_{opt} \\ 0 & otherwise \end{cases},$$

where C_k is the temperature or *control parameter* that varies with the number of iterations. Note that the simulated annealing algorithm converges with probability 1 to a globally optimal configuration if the series of values of the control parameter or temperature converges to 0 and if the Markov chains generated at each value of the control parameter are of infinite length.

In the next two sections, a version of the simulated annealing algorithm is used for texture segmentation using a maximum a posteriori criterion, and to train a MNN model for template identification.

6.2. Texture Segmentation Using Simulated Annealing

Image segmentation is the partitioning of an image into a set of non-overlapping regions. The purpose of image segmentation is to decompose an image into parts that are meaningful with respect to a particular application. General segmentation procedures tend to obey the following rules [81]: (1) Regions in the image should be uniform and homogenous with respect to certain properties; (2) region interiors should be simple, without many holes; (3) region boundaries should be simple and not ragged. When the image is represented by multiple properties, the basic approach for extracting regions with similar properties should, more or less, be achieved by computing the multiple properties at each pixel or a "selected region" in the image, mapping every pixel or "selected region" to the multidimensional space and classifying the pixels or "selected regions" and finally labeling each pixel or "selected region" accordingly [82]. The crucial factors in this process are the properties to be used for classification.

A number of segmentation algorithms were developed in [81] to segment images by using neighboring average gray values. However, many complex images, including most natural images, cannot be segmented merely by average gray values. Just as single textures can be modeled stochastically, images consisting of various textures may also be represented by a model. When the individual textures in the images cover large regions, and do not contain holes, then the image regions can be modeled by a Gibbs distribution [43], [70], [83] which include prior contextual information. The contextual information of neighborhood pixels is important since spatially close pixels are likely to belong to the same region or texture. Once the region distribution is modeled effectively, a good segmentation algorithm should be selected. A variety of MRF based segmentation algorithms have been proposed in the literature, including using a *maximum a posteriori* criterion that is solved by simulated annealing [42], [43], *iterated conditional modes* (ICM) [41], and *maximum marginal probability* (MPM) [41] using a Gibbs Sampler [42]. These methods differ in the details of the type of prior model, in the model for noise corrupting the true image, in the type of optimization algorithm, and in the details of the algorithms themselves. Choices for the model for the region distributions and the individual textures are dictated by the problem at hand. The multi-level logistic model (MLL) has been found to be a good model for region distributions when using labelled images [43], since it considers differences in neighboring gray values or "pixel labels" to compute the probabilities. In the previous chapters the fourth-order POMM model was shown to model individual textures effectively. In this section, a hierarchal MRF model is proposed. The Gibbs distribution is used to model the region distributions and the fourth-order POMM with a binomial distribution is used for modeling the textures within the regions.

A brief description of the Gibbs distribution is given below, followed by details of the specific MLL model used to model the regions. Readers are referred to Chapter 3 for more details on the Gibbs distribution. The simulated annealing algorithm used to obtain the MAP estimate of an unknown image comprised of various textures with known parameters is outlined. Results on real and synthetic images are presented.

6.2.1. Hierarchical Image Model and Problem Formulation

6.2.1.1. Gibbs Random Field with a MLL for Region Distribution.

The Gibbs distribution with a MLL model was used for image restoration in [42] and for image segmentation in [83], [84]. For our purposes, we assume that the regions in the image have reasonable sizes, smooth boundaries, and no isolated small holes. The joint distribution of the region is represented by a Gibbs distribution (GD) of the form:

$$P(X) = \frac{1}{Z} exp(-U(X)),$$
$$U(X) = \sum_{c \in C} V_c(x),$$
$$Z = \sum_{X} exp(-U(X)).$$

Here Z is the partition function, $U(X)$ is the energy function that is the summation of the potential function $V_c(x)$ over all possible cliques $c \in C$, and X is the segmented image.

The MLL model is a particular type of Gibbs random field model that has been studied in [42]. In order to define the energy functions of the MLL model, the cliques associated with a neighborhood system η, and the potential functions associated with the cliques need to be defined. It has been proved experimentally in [83] that second-order neighborhood systems, η^2, are very flexible and powerful, and adequate to represent many region distributions [43]. The cliques of a second order neighborhood are shown in Fig. 37.

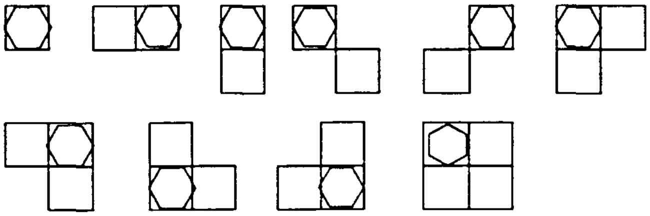


Figure 37. Cliques of a second order neighborhood system. The marked cell is location *(i,j)*.

For more details on cliques and potential functions, the reader is referred to Chapter 3. Potentials need to be assigned for each possible configuration. The idea is that cliques with the same gray levels should have a low potential, and cliques with different gray values have a high potential. The objective is to obtain a region configuration with a low potential. This effectively means that regions of very small sizes are not acceptable. The potentials assigned to some possible region configurations in the labelled image is shown in Fig. 38.

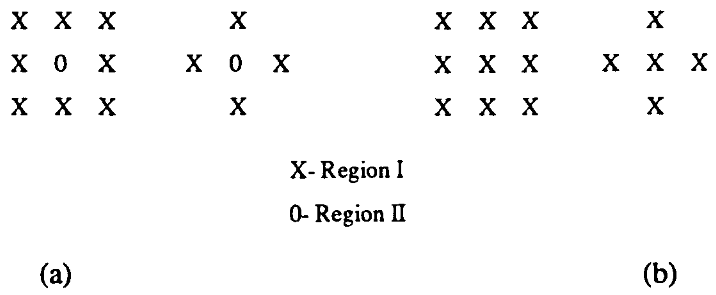


Figure 38. Region configurations with (a) high potential (b) low potential.

The models most commonly used for GDs are the pairwise interaction models, where only cliques of first and second order exist, or equivalently, the potential functions of cliques greater than second order are zero. The potential functions commonly used in literature [83], [84] are shown below:

- For single pixel cliques: $V_c\{x_s\} = e_k$ for $x_s = k$ where e_k is a function of k
- For 2 pixel cliques: $V_c\{x_s\} = \begin{cases} -d & \text{if all pixels in clique are equal} \\ d & \text{otherwise} \end{cases}$
- For cliques with more than 2 pixels: $V_c\{x_s\} = 0$

Since the Gibbs random field and the MRF are equivalent, the conditional probability of the region distribution can be written as:

$$P(X_s = i|\eta_s) = \frac{exp\left(-\sum_{c:s\in c} V_c(x_s = i)\right)}{\sum_{j=0}^{L-1} exp\left(-\sum_{c:s\in c} V_c(x_s = j)\right)}, \tag{31}$$

where η_s is a neighborhood of x_s. The summation in the exponent is over all cliques containing s.

A crucial factor is the choice of value for *d* for the two-pixel cliques. A larger value for *d* creates more clumping, while a negative value for *d* causes neighboring pixels to have different gray levels or a "repulsion" effect. When *d*=0, the pixels in the clique have no interdependence. The e_k's control the proportion of pixels at each gray level. Since no prior knowledge exists of the regions in the image, typically $e_k = 0$ for the single pixel cliques.

6.2.1.2. POMM Model For Textures.

Here, it is assumed that the texture inside the regions are homogenous and relatively large. The fourth-order POMM with a binomial distribution, proposed in Chapter 5 is used to model the textures. See Eq. 17 in Chapter 5, for a description.

6.2.2. Segmentation Using MAP Criterion

Bayesian modeling is a popular technique for formulating estimation and pattern classification problems [85]. A Bayesian model is a statistical description of an estimation problem which consists of two separate components. The first component, the *prior model, P(X)*, is a probabilistic *prior* description of the system. The second component, the *sensor model*, $P(A|X)$, is a description of the stochastic processes that relate the original, unknown state *X* to the sampled input image or sensor values *A*. These two probabilistic models can be combined to obtain a *posterior model*, $P(X|A)$, which is a probabilistic description of the current estimate of *X* given the data *A*. The posterior model can be obtained by using Bayes Rule

$$P(X|A) = \frac{P(A|X)P(X)}{P(A)}.$$

Bayesian modeling is used to compute the *maximum a posteriori* (MAP) estimate [42], a value of *X* which maximizes the conditional probability $P(X|A)$, or in calculating the *maximum marginal probability* that minimizes the expected proportion of misclassified pixels [41].

Here, the segmentation problem is formulated as a maximum a posteriori problem, where some representative textures are available, from which the parameters have been estimated. If *X* is the segmented, labelled image, *A* is the original image, let there be *i=1,2,...,L* possible regions with respective parameters ψ_i. Pixel $X_{mn} = i$ labels pixel A_{mn} as belonging to texture *i*.

Given the original image, A, the problem is to find a segmented image X^*, such that: $P(X^*|A) = \max_X P(X|A)$. By Bayes rule we have $P(X|A) = \frac{P(A|X)P(X)}{P(A)}$, and the probability of image A, $P(A)$, is a constant independent of changes in X. Thus we need to find an X^* such that

$$P(X^*|A) = \max_X P(A|X)P(X).$$

Here image X is modelled by a Gibbs distribution,

$$P(X) = \frac{1}{Z_X} exp(-U_X(X)),$$

where $P(A|X)$ is a Gibbs distribution of the form

$$P(A|X) = \frac{1}{Z_A(X)} exp(-U_A(A|X)).$$

Here, $Z_A(X)$ is the partition function,

$$Z_A(X) = \sum_A e^{-(U_A(A|X))}.$$

Then the joint conditional probability is written as

$$P(A|X)P(X) = \frac{1}{Z_A(X)} exp(-U_A(A|X)) \frac{1}{Z_X} exp(-U_X(X)).$$

The maximization of this product requires a large amount of computations, because $Z_A(X)$ needs computation of the exponentials of $U_A(A|X)$ over all A, while the $U_A(A|X)$ themselves are functions of X. This results in a huge number of computations. Since the criterion function $P(X|A)$ is nonlinear, it is extremely difficult to find the optimum solution X^* by any conventional method. Thus an iterative procedure that does not involve computation of the partition function $Z_A(X)$ is needed. A approach that requires computation of ratio of probabilities such as simulated annealing or ICM is adequate.

The following properties of MRFs involving computation of ratios of probabilities have been proven in [43] that state that under certain conditions, computation of the partition function $Z_A(X)$ is avoided. The segmented image, X, and the observed image, A, are represented as vectors for notational convenience, $X = \{X_1, X_2, ..., X_M\}$ and $A = \{A_1, ..., A_M\}$. If one pixel X_k in X is changed, and the new image is denoted as $\widehat{X} = \left\{X_1, X_2, .., \widehat{X_k}, .., X_M\right\}$, then the ratio of the conditional probabilities for the changed and original images and the ratio of the joint pdf of the region distribution for the changed and original images can be written in the form:

$$\begin{gathered}\frac{P\left(A|\widehat{X}\right)}{P(A|X)} = \frac{P\left(A_k|A_1, ., A_{k-1}, A_{k+1}, ., A_M, X_1, .., \widehat{X_k}, .., X_M\right)}{P(A_k|A_1, ., A_{k-1}, A_{k+1}, ., A_M, X_1, ., X_k, ., X_M)} \\ \frac{P\left(\widehat{X}\right)}{P(X)} = \frac{P\left(\widehat{X_k}|X_1, .., X_{k-1}, X_{k+1}, .., X_M\right)}{P(X_k|X_1, .., X_{k-1}, X_{k+1}, .., X_M)}.\end{gathered} \tag{32}$$

This form does not involve the normalizing constant, and hence is more efficient for calculations. Since we assume A to be a realization of a POMM, and X is modeled by a MRF due to the GD-MRF equivalence, the ratio of $P\left(\widehat{X}|A\right)$ and $P(X|A)$ reduces to a ratio of the product of local conditional probabilities of the chosen pixel X_k. Thus

$$\frac{P\left(A|\widehat{X}\right)P\left(\widehat{X}\right)}{P(A|X)P(X)} = \frac{P\left(A_k|N_k, \widehat{X_k}\right)P\left(\widehat{X_k}|\delta_k\right)}{P(A_k|N_k, X_k)P(X_k|\delta_k)} \tag{33}$$

where $N_k \equiv adj_{\prec}A_k$, is the causal neighborhood of A_k for the fourth-order POMM, and δ_k is the neighborhood of X_k for the second order MRF. The configurations of N_k and δ_k are shown below in Fig. 39. Since only a ratio of the product of two terms are involved, the maximization can be implemented iteratively by using a simulated annealing approach as follows.

(a) (b)

Figure 39. Neighborhood systems (a) $N_k \equiv adj_{\prec}A_k$ for fourth-order POMM, and (b) δ_k for second order MRF, represented by the shaded regions.

6.2.3. Simulated Annealing (SA) for Texture Segmentation

1. Given image A, obtain an initial segmented image $X^{(0)}$, and set iteration number, $n=0$.
2. Assign an initial "temperature" or control parameter as: $T_n = \frac{C_0}{ln(C_1+n)}$, where C_0 and C_1 are predetermined constants.
3. Choose a pixel at site k, $X_k^{(n)}$, in the segmented image $X^{(n)}$, and update its value through a predetermined procedure, either randomly or by selecting a pixel value from a neighborhood of the pixel. The new label of the pixel is $\widehat{X}_k^{(n)}$.
4. Compute
$$\rho = (\frac{P\left(A_k|N_k, \widehat{X_k}\right)P\left(\widehat{X_k}|\delta_k\right)}{P(A_k|N_k, X_k)P(X_k|\delta_k)})^{1/T_n}$$
If $\rho > 1$, $X_k^{(n)} = \widehat{X}_k^{(n)}$, otherwise $X_k^{(n)} = \widehat{X}_k^{(n)}$ with probability ρ.
5. Go back to step 3 and repeat until image A is scanned once.
6. Increment $n=n+1$, decrease temperature T_n as given in step 2, and repeat steps 3,4 and 5 until $T_n < T_e$.

The advantages offered by using SA to compute the MAP estimates are:

1. Iterative computation of the local conditional probabilities is required, thus avoiding large numerical computations;
2. Multi-site updating is possible when multiple processors are available.

The crucial factors affecting the algorithm are:

1. A method for the initial segmentation;
2. Value of d for potential function, which should not be too large or too small;
3. Order in which sites are updated. Visiting sites in sequential order might result in unwanted global effects and loss of compactness of segments;
4. Cooling schedule, and values of C_0 and C_1 that determine the initial temperature T_0;
5. Stopping criterion.

If the initial segmentation is a good approximation of the actual region distribution, then the cooling schedule, or the number of iterations at each temperature and the choice of the final temperature do not affect the final results. However, the neighborhood size used to classify each pixel and the magnitude of the potentials, i.e., the value *d*, assigned to each possible configuration, can affect the final results as shown below.

6.2.4. Initial Segmentation

The speed of convergence of the SA algorithm to the MAP estimate depends largely on the initial labeled image $X^{(0)}$. Hence, a crucial factor in this algorithm is the method chosen to compute the initial segmented image $X^{(0)}$. One method to label a pixel *k* in the image *X* as *j* is, if $P(A_k|N_k;\psi_j) = \max_i P(A_k|N_k;\psi_i)$, where N_k is the neighborhood $adj_{\prec}a_k$ of r.v. A_k for an *A* that satisfies a fourth-order POMM, and ψ_i is the set of parameters corresponding to texture *i* that has been predetermined from available texture samples. However, since a neighborhood criterion may be insufficient to yield good results, an alternate method is to use the joint local pseudo-likelihood to label the pixels. The joint local pseudo-likelihood is defined as $J(i) = \prod_{s\in B} P(A_s|N_s;\psi_i)$, *i=1,...,L*, where *B* is a set of pixels chosen by the user. If *J(k)* is the largest among *J(i), i=1,...,L*, then all pixels in *B* are assigned to texture *k*.

The above method of joint local likelihood as suggested in [43] may result in a blocky segmented image, which may not be accurate at the texture boundaries, and may not be compact. An alternative way to use local neighborhood information to segment the image is to compute the joint local pseudo-likelihood for a certain neighborhood around each pixel as follows: $J_{m,n}(i) = \prod_{s\in Y_{m,n}} P(A_s|N_s;\psi_i)$, *i=1,...,L*, where $Y_{m,n}$ is a neighborhood around $A_{m,n}$. Then $X^{(0)}_{m,n} = k$ if $J_{m,n}(k) = \max_i J_{m,n}(i)$.

The next section presents results of applying simulated annealing to segment natural images, using various neighborhood configurations for the joint local pseudo-likelihood to obtain the initial image $X^{(0)}$.

6.3. Texture Segmentation Results

Three sets of natural images were used to test the segmentation algorithm. In each case, two of the Brodatz images of size 128x128 were combined into a single 128x128 image by selecting their border at random. Images I and II contain Brodatz cloud and water images, and image III contains Brodatz tree and water textures, with the borders at different, randomly chosen pixel locations. For each segmented image, *the texture border is shown as a sequence of black pixels.* The above segmentation algorithm using SA to compute the MAP estimate was applied with different neighborhood sizes and magnitudes of the clique potential functions. In all cases the method of pixel updating in the SA algorithm was as follows: the updated pixel value was chosen at random from the set of all possible labels. It is mentioned in [43] that a faster convergence is obtained if the candidate pixel values are chosen from a set of all values appearing in the neighborhood of a pixel. Such a technique has not been implemented here, since the final result of the simulated annealing algorithm is not affected by the method chosen for updating the pixel labels. However, since the performance of the initial segmentation does affect the final results, various neighborhood configurations for computing the joint local likelihoods have been tested.

6.3.1. Initial Texture Segmentation Results.

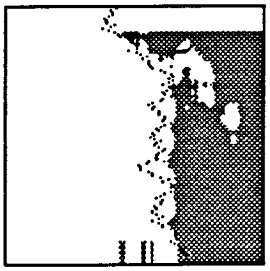
The initial texture segmentation results using the method outlined in Section 6.2 to obtain the initial labelled image $X^{(0)}$ is presented here. Various neighborhood configurations were used to compute the joint local likelihood at each instance, to estimate experimentally the size of the "optimum" neighborhood for initial segmentation. In all the results shown below, square shaped neighborhoods, $Y_{m,n}$ around each pixel *(m,n)* was used to compute the joint local pseudo-likelihood as follows: $J_{m,n}(i) = \prod_{s \in Y_{m,n}} P(A_s|N_s;\psi_i)$, $i=1,\dots,L$, where $Y_{m,n}$ is a neighborhood around $A_{m,n}$. Then pixel *(m,n)* is labelled as $X^{(0)}_{m,n} = k$ if $J_{m,n}(k) = \max_i J_{m,n}(i)$. A different neighborhood size $|Y_{m,n}|$ was used in each case.

(a) Image I: Combined Brodatz cloud and water images.

(b) Initial segmented image I by using local joint probability with neighborhood size 2x2.

(c) Initial segmented image I by using local joint probability with neighborhood size 16x16.

(d) Initial segmented image I by using local joint probability with neighborhood size 24x24.



(e) Initial segmented image I by using local joint probability with neighborhood size 32x32.

(f) Image II: Combined Brodatz water and cloud images.

(g) Initial segmented image II by using local joint probability with neighborhood size 16x16.

Figure 40. Initial segmentation of the Brodatz images.

(h) Image III: Combined Brodatz tree and water textures.

(i) Initial segmented image III by using local joint probability with neighborhood size 16x16.

Figure 40. (continued)

A neighborhood of size 16x16 for computing the joint local likelihood seems to yield good results among all the configurations, especially at the borders between textures. Also, the classification is better for homogenous looking textures as for image III shown in Fig. 40(i), where misclassification occurs only near the borders. Note that the initial segmentation results are not very good for the heterogeneous cloud textures in images I and II, shown in Fig. 40(b), (d), (e) and (g).

6.3.2. Final Segmentation Results.

After initial segmentation, simulated annealing (SA) was used to obtain a MAP estimate of images I and II using various values for the final temperature T_c, and clique potential d. The value of T_c controls the number of iterations of the SA algorithm, and the clique potential d controls the degree of clumping of similar gray values, and thus indirectly the size of textured regions in the segmented image. The parameter values of the SA algorithm were varied to observe the results on the MAP estimates.

It is observed that a value of d=1.0 yields the best segmentation results by using a neighborhood of size 16x16 for initial segmentation. The worst results are obtained by using a neighborhood of size 2x2, followed by a MAP segmentation with a clique potential d=2.5 as shown in Fig. 41(c). This shows that a larger value of the clique potential d causes more clumping in the segmented image.

(a) Final segmented image after 257 iterations on image in Fig. 40(c). Final temperature T_e = 0.18, clique potential d= 1.0.

(b) Final segmented image after 257 iterations on image in Fig. 40(c). Final temperature T_e = 0.18, clique potential d= 2.5.

(c) Final segmented image after 257 iterations on image in Fig. 40(b). Final temperature T_e= 0.18, clique potential d =2.5.

(d) Final segmented image after 257 iterations on image in Fig. 40(c). Final temperature T_e= 0.18, clique potential d =1.5.

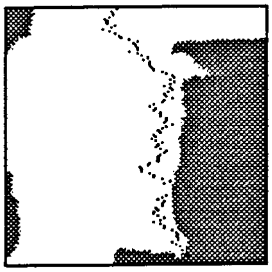
Figure 41. Final segmentation results obtained by application of simulated annealing with different parameters.

6.3.3. Modified Simulated Annealing Algorithm I

Pixel updating by scanning the image in a orderly manner as suggested earlier for the SA algorithm *might* result in unwanted global effects on the entire image, as compared to a random updating of sites. An alternative method for the updating is suggested here. One way to avoid such a snowball effect on the segmentation procedure would be to scan the whole image, store the interchanges as given by the simulated annealing algorithm discussed previously, and then change or update the pixel labels after one scan through the whole image. The results of applying this method on image I is shown below in Fig. 42.

(a) Final segmented image after 85 iterations on image in Fig. 40(b). Final temperature T_c= 0.225, clique potential, *d*= 1.0.

(b) Initial image segmentation using neighborhood of size 6x6 to compute joint local likelihood. Final temperature T_c= 0.215 (100 iterations or scans through image), clique potential, *d*= 1.0.



(c) Final segmented image after 100 iterations on image in Fig. 40(e). Final temperature T= 0.215, clique potential, *d*= 2.0.

Figure 42. Modified SA algorithm technique I.

This modified annealing algorithm results in more misclassified pixels than the original SA algorithm discussed earlier in Section 6.2.3. Even though this method is observed to give better results at texture borders as in Fig. 42(a), misclassification occurs at sites within the texture regions in Fig. 42(a), (b), and (c) resulting in patchy labelled images with a number of holes within the texture regions. Thus, this method does not offer any definite advantages over the original SA algorithm.

6.3.4. Modified Simulated Annealing Algorithm II

The SA algorithm suggested in Section 6.2.3 involves changes in pixel labels even when the labelled image $X^{(n)}$ at iteration *n* is "close" to the MAP estimate X^*. This results in slow convergence. An alternative procedure to speed up the convergence to the MAP estimate would be to update a pixel label only if an extra precondition is satisfied. In step 4 of the original SA algorithm given previously, a pixel X_k may be updated only if at least one pixel X_p in the neighborhood, M_k of X_k is different from X_k. That is, X_k is updated only if $X_k \neq X_p$, where $X_p \in M_k$. The results of applying this algorithm to the image in Fig. 40(c) are shown in Fig. 43. In Fig. 43(b), the pixels are updated only after one scan through the entire image, similar to the modified simulated annealing algorithm discussed previously.

(a) Final segmented image after 257 iterations on image in Fig. 40(c) with $M_k = 6 \times 6$, final temperature 0.18, Clique potential, *d*= 1.0.

(b) Final segmented image after 257 iterations on image in Fig. 40(c), $M_k = 8 \times 8$, update segmented image after 1 scan through whole image. Final temperature 0.18, clique potential, *d*= 2.5.

Figure 43. Modified SA algorithm technique II.

The image in Fig. 43(a) is similar to the one obtained by applying the original SA algorithm in Fig. 41(b), showing that this method gives the same results with a faster convergence rate. The image in Fig. 43(b) obtained by a combination of the two modified SA algorithms proposed here has a number of misclassified pixels in the texture regions, showing that the first modified SA algorithm in Section 6.3.3 does not have any potential applications in this segmentation technique.

The second modified SA algorithm suggested in Section 6.3.4, however, could be used to obtain faster convergence as compared to the original SA algorithm.

The segmentation technique using simulated annealing gave good, though not perfect results for the combined Brodatz images. A possible solution to obtain better results would be to fit a degradation model to the observed data, Z, and use a MAP technique on the perfect textured image and the segmented image given the observation. That is, if Y is the perfect textured image that has been degraded to observed image Z, and X is the segmented image, then we have to find a X^* and Y^* such that:

$$P(X^*, Y^* | Z) = \underset{X,Y}{Max} P(X, Y | Z)$$

Another disadvantage of the method suggested here is that the parameters of the representative textures should be known a priori. An unsupervised technique that involves parameter estimation from blocks in the given image and then using a clustering algorithm to find the representative parameters could be used. The performance of this technique on complex image data needs to be analyzed in further detail.

6.4. Simulated annealing as a Learning Rule for Morphology Neural Networks

The area of artificial neural networks (ANNs) has seen a large increase in activity since the 1980s, both in the theoretical and applications arena. Used mainly as a pattern recognition technique, ANNs have been applied to such diverse areas as robot control, forecasting in management and business, electric power systems, and image processing. ANNs have provided solutions to practical problems that are ill-posed and ill-conditioned, as well as to problems that have incomplete or ambiguous data. Here, a morphology neural network, a new type of artificial neural network, is presented, and applied to an image processing problem. Due to space limitations it is assumed that the reader is familiar with classical neural networks. For a good introduction, see [86].

Classical neural nets calculate the output at each node as given below, where w_{ji} is the weight from the lower node i to the upper node j, a_i is the lower neuron value at node i, and c_j is the upper neural value at node j. There are N nodes in the lower layer. The function f is the nonlinear activation function.

$$c_j = f(\sum_{i=1}^{N} a_i w_{ji}) \tag{34}$$

By replacing the algebraic operations of addition and multiplication with other operations, additional neural networks can be obtained that combine information in different ways [87]. Replacing the multiplication in Eq. 34 with addition, and the summation sign with a maximum operator, $\bigvee$, we have:

$$c_j = f\left(\bigvee_{i=1}^{N} a_i + w_{ji}\right). \tag{35}$$

Neural networks of this type are called *morphology neural networks* (MNN) [88], as this particular operation mimics a morphological image processing operation called *dilation*. Mathematical morphology is a nonlinear methodology for image processing which uses a template representing the shape of the object to be manipulated in the image. A dilation is exactly Eq. 35 where the weights $\{w_{ji}\}$ represent a shift invariant nonlinear transformation. For more information on mathematical morphology, consult [89]. While morphology operations can be emulated by classical neural networks [90], the morphology neural network is a network which directly emulates algebraically the morphology image transformations.

The image processing problem tackled here is one of transformation identification: given the input image a, and the dilated output b transformed by an ideal template $\{w_{ji}\}$ as in Eq. (35), we wish to approximate the template values by determining a set of weights producing b. An example of the dilation process is given in Fig. 44. The template t is an invariant template. The image is row-ordered as a vector, scanning from the top left-hand value to the bottom right-hand value. This particular MNN model has one input layer, one output layer, and limited interconnections. The network is trained using a set of gray valued images and the computed (dilated) output is compared with the desired output obtained by dilating the input image with the ideal template. The weights of the MNN are then modified by using simulated annealing as a learning rule.

The 4–step SA algorithm for training of the MNN model is given below [91]:

1. Initialize the control parameter C_0 and choose an initial state of values for the weights $\{w_{ji}\}$.
2. One weight value w_{ji} is chosen at random and changed by adding a constant ±k to it, $k \geq 0$. We used k=1. Call this set of weights $\hat{w}$. Then the new cost value $g(\hat{w})$ is calculated using this new weight value, and the cost difference, Δg, between the previous set of weights w^{old}, and the weights with the new value, $\hat{w}$, is calculated. If $\Delta g = g(\hat{w}) - g(w^{old}) < 0$, then $\hat{w}$ becomes the new set of weights w^{new}, otherwise $\hat{w}$ becomes the new set of weights with probability $p = e^{\frac{-\Delta g}{C_k}}$ (Metropolis Criterion). This last step is achieved by arbitrarily choosing $\beta \epsilon$ [0,1] uniformly and randomly, and if $\beta < p$, then $w^{new} \leftarrow \hat{w}$: otherwise if $\beta > p$, then $w^{new} \leftarrow w^{old}$.
3. Repeat step 2 l_k number of times at each temperature, C_k; see [79], [80] for details.
4. Decrement temperature C_k [78] and go back to steps 2 & 3 until the stop criterion is satisfied. The stop criterion is a constant, ϵ_f, chosen by the user.

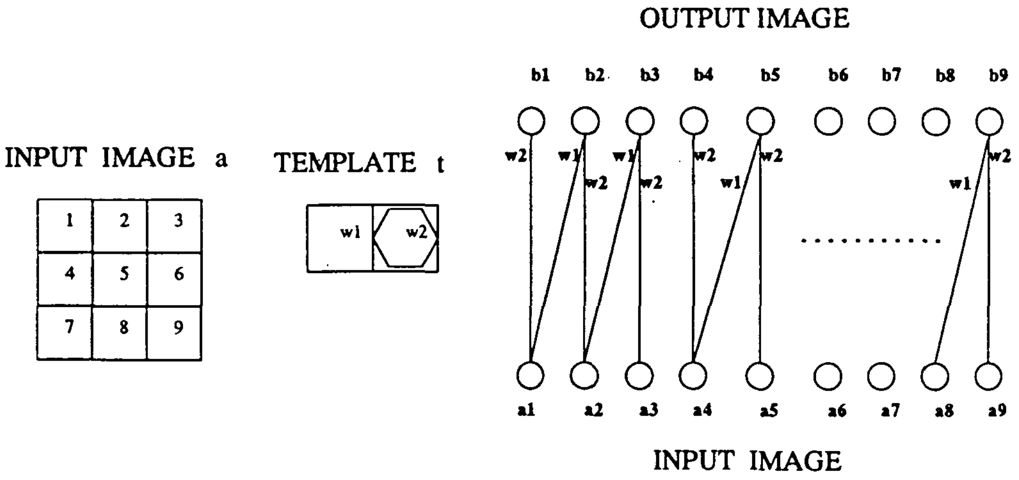


Figure 44. (a) Input image a, template t, and mapping operation to a morphological neural net.

A crucial choice is the choice of the cost function which is to be minimized. The goal is to iteratively compute a set of weights $\{w_{ji}\}$ so that given a set of training pairs $\{a^i, d^i\}_{i=1}^{M}$, the weights w satisfy $a^i \boxed{\vee} w = d^i, i = 1, \ldots\ldots, M$. For our application, possible cost functions are:

$$1. \quad g(w) = \sum [a \boxed{\vee} w - d] \tag{36a}$$

$$2. \quad g(w) = \sum |a \boxed{\vee} w - d| \tag{36b}$$

$$3. \quad g(w) = \sum \left|(a \boxed{\vee} w)^2 - d^2\right|^{1/2} \tag{36c}$$

$$4. \quad g(w) = \sum [\chi_{\geq 0}(d - a \boxed{\vee} w) + \lambda\chi_{>0}(a \boxed{\vee} w - d)] \tag{36d}$$

$$5. \quad g(w) = \sum [\chi_{\geq 0}(a \boxed{\vee} w - d) + \lambda\chi_{>0}(d - a \boxed{\vee} w)]. \tag{36e}$$

The choice of parameters for the simulated annealing algorithm applied to this problem are discussed next.

The initial value C_0 of the cooling control parameter should be large enough so that initially all configurations are accepted with equal probabilities. The value of C_0 is chosen as:

$$C_0 = \rho \bigvee_{r \epsilon R_s} \Delta g_{sr},$$

where s is a fixed state of the system (in this case a particular set of values assigned to all weights, $\{w_{ji}(s)\}$, ρ is a fixed constant, and R_s is the subset of all possible states of the system that are determined by allowing exactly one weight value w_{ji} to change by its permissible amount. In any case, if $\{w_{ji}(s)\}_{i,j=1}^{N}$ is the fixed state, then

$$R_s = \{\{w_{ji}(r)\}_{i,j=1}^{N} : w_{ji}(r) = w_{ji}(s) \text{ except at one}$$

$$\text{location } (m, n), \text{ where } w_{mn}(r) = w_{mn}(s) \pm k\} \ .$$

An alternate method has been proposed in [79] to compute the initial parameter, C_0.

The rule for decrementing the control parameter is given by

$$C_{k+1} = \frac{C_k}{(1 + C_k ln(1 + \delta)/3\sigma_k)} \ ,$$

where σ_k is the standard deviation in cost values in the k-th Markov chain. Alternatively, the control parameter can be decreased by a constant factor of 0.8–0.9 [79], as long as the Markov chains are long enough to allow the system to arrive at a stationary distribution.

The finite chain length l_k may be set at the maximum vector length of all possible states,

$$l_k = \bigvee_{s} \left\{ vector\ length\ of\ \{w_{ji}(s)\}_{i,j=1}^{N} \right\}.$$

The set of weights can be viewed as a vector in N^2 dimensional Euclidean space.

One choice of value for the stopping criterion is a value ϵ_f satisfying $\epsilon_f < \frac{g(C_k) - g_0}{g_{init} - g_0}$, where $g(C_k)$ is the cost at temperature C_k at iteration k, g_o is the optimal cost for the entire system, and g_{init} is the cost for the initial values chosen for $\{w_{ji}\}$.

Simulation results. The proposed technique was implemented on a computer and tested on 16 x 16 and 64 x 64 images. A set of training data, i.e., input images a and their corresponding desired outputs d generated by the desired template, was used to train the neural network. The sum of absolute differences between computed output and desired output in Eq. 36(b) was used as a cost function.

The values of the five parameters used were chosen from the range:

Initial temperature, C_0= 30 – 300

$2 \leq \rho \leq 10$

$1500 \leq l_k \leq 2500$

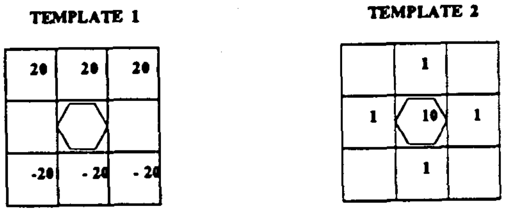
$10^{-7} \leq \epsilon_f \leq 10^{-5}$

$C_{k+1} = \frac{C_k}{(1+C_k ln(1+\delta)/3.928)}$

It was observed that almost similar results were obtained when the parameter values were within these ranges.

The weights of the neural network, which correspond to the template weights, were randomized uniformly to integers in the interval [0,255], and the network was trained on a set of ten 64 x 64 images. The desired templates used were the two shown in Fig. 45. The network converged to the correct set of weights for these templates after 5– 7 changes in temperature, corresponding to 12500

– 17500 iterations or changes in state, as compared to 4.7233 $\times 10^{21}$ iterations which an exhaustive search through all possible states would have taken. See Fig. 46 for the output images resulting from applying learned and ideal template weights shown in Fig. 45(a). The algorithm was found to perform better as compared with the iterative improvement method both in terms of effectivity, or the quality of solution returned by the algorithm, and the efficiency, that is the computation time taken by the algorithm.



TEMPLATE 1

20	20	20
-20	-20	-20

TEMPLATE 2

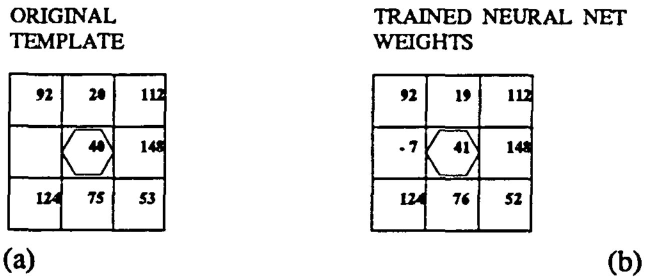
	1	
1	10	1
	1	

Figure 45. Original templates used for dilation. Neural net weights converged to exact weights after training.

Figure 46. (a) Input image dilated with initial random weights; (b) Output after 2000 iterations; (c) after 6000 iterations; (d) after 10000 iterations, i.e., weights converged.

The network converged to the correct template when only a single 64 x 64 image was used as training data, but the convergence time was longer with 8–9 changes in temperature, as compared to 5–7 changes in temperature when ten training data sets were used.

To further test the efficacy of the algorithm, the desired template weights were chosen at random and the network was trained on a set of training data of 10 64 x 64 images. The random weights chosen are shown in Fig. 47(a).



ORIGINAL TEMPLATE

92	20	112
	40	148
124	75	53

(a)

TRAINED NEURAL NET WEIGHTS

92	19	112
-7	41	148
124	76	52

(b)

Figure 47. Random template. (a) Original template weights; (b) trained neural net weights.

The minimum cost obtained after 40 changes in temperature was 17.88, at which point the algorithm was terminated as the stopping criterion was not yet satisfied. The corresponding neural net weights are as shown in Fig. 47(b).

It is observed that the higher-valued template weights corresponded exactly with the neural net weights, whereas the smaller valued neural net weights did not converge to the correct values. This is probably due to the fact that higher template weights will influence the output images more than the smaller values due to the operation of maximum, hence not much information was available in the training data regarding the lower valued weights. The computed weights were tested on a 64 x 64 array cloud data and compared with the perfectly dilated output as shown in Fig. 48(a) and 48(b). The results are shown in Table 6 below.

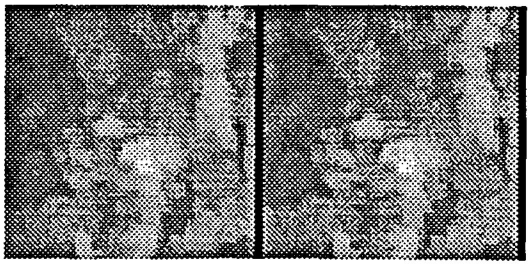


Figure 48. Dilated cloud image: (a) Ideal output, (b) input image dilated with trained neural net weights

Table 6. Results of template learning.

Template Number (Figure Number)	Number of pixels out of 4096 in which the ideal dilation and N.N dilation differ	Average absolute diff. between ideal dilation and N.N dilation	Gray level range of image		Column 3 as percentage of gray level range
			Minimum gray value	Maximum gray value	
Fig. 45(a)	0	0	0	255	0 %
Fig. 45(b)	0	0	0	255	0 %
Fig 47(a) (tested on cloud data)	868	1.351	132	2675	0.06755 %

The training algorithm for the MNN was found to give computationally good solutions, even for random templates, in reasonably good CPU time. The weights converged exactly to the nicely symmetric ideal templates in Fig. 45, and very close in the random template trial, Fig. 47. These are excellent results. The training may possibly be made more efficient by a different choice of cost functions or an alternate cooling schedule for the annealing algorithm. A modified annealing algorithm has been suggested for accelerated solutions. Applying the training for translation variant templates is also promising. The approach is flexible and may be applied to other MNN problems. The results could also be extended to multilayer morphology networks to solve more complex mapping problems, for example, in texture identification using templates.

This effort was also presented as a paper at the second Midwest Electro-Technology Conference, April 1993 in Ames, Iowa.

7. CONCLUSIONS AND FUTURE RESEARCH

7.1. Conclusions

In this research, a statistical model-based approach using a new class of models, POMMs, has been described for texture synthesis and image segmentation. The advantages of the POMMs as compared to other model-based approaches are the factorability of the joint distributions into a product of conditionals, and a lack of the partition function that makes the computation of the joint distribution much easier.

A new fourth order POMM has been developed and used for texture synthesis of binary, four, and sixteen gray level images. The parameters that describe the POMM model control the strength and direction of clustering of gray values in the image, and together can model a number of spatial dependencies. The Flinn's algorithm yields a sample from the POMM distribution in reasonable time with a reduction in computational complexity, as compared to generating a sample from a MRF. The synthesized images look very similar to those generated in [3], [92], and [93], using other Markov random field approaches. Spatially varying textures have also been generated using spatially varying parameters.

The model has been found to give good results when applied to the inverse problem, that is, the estimation of parameters representing unknown textures that are assumed to have an underlying POMM. The synthetic images generated from the ML estimates of natural Brodatz textures were visually close to their natural counterparts in three out of five cases. One attractive feature of the POMM model is the feasibility of using a gradient descent approach for ML parameter estimation due to the product form of the joint distribution of the r.v.s. The parameter estimation problem is further simplified since only five parameters are involved for the fourth order POMM, thus reducing the search space for the ML problem.

An image composed of several textures is modeled by a MLL MRF model, with the third and higher order clique pairs assumed to be zero. The underlying individual textures within the image are modeled by a fourth order POMM, each having a unique set of parameters. The images are segmented by a MAP criterion using simulated annealing to obtain the MAP estimates. One advantage of using SA for optimization is that it can be implemented in parallel to speed up the convergence. This segmentation technique was observed to give good segmentation results on Brodatz images. The segmentation results were better for images composed of homogenous textures.

The advantages of the fourth order POMM model as models of texture are summed up below.

1. The model parameters are sufficient to generate a variety of random and structured textures. This offers a definite advantage over feature-based approaches that can be used merely for classification. The model parameters control the directionality, sharpness, and coarseness in the image.
2. The local spatial interactions enforced by the POMM parameters create a global effect on the entire image.
3. Realistic looking patterns can be generated with the fourth order POMM model.
4. The parameter estimation procedure is simplified as compared to an MRF where a coding of disjoint sets of pixels is required; this results in a reduction in sample size and a loss of efficiency in estimation.
5. The MAP problem is simplified since the joint distribution can be written as a product of the conditional probabilities.
6. A hypothesis test on the parameter estimates of the POMM is proposed that can be used to test the feasibility of the estimates.

The POMM model and the segmentation algorithm proposed have the following drawbacks.

1. Large images are required to obtain good estimates.
2. All theoretical properties of POMMs are not known at present.

3. The segmentation algorithm proposed is a supervised technique, that is, the actual parameters of the representative textures needs to be known in advance.

7.2. Future Research

As a result of the investigation of the POMMs in this thesis, several questions leading to future research areas have arisen. The questions and suggested solutions are presented below.

The focus of this research was on the application of a new fourth order POMM as a texture model. Even though this model was found to be sufficient for modeling *most* textures, it is most probably not a comprehensive one that can simulate *all* natural textures. The issue of model selection is crucial to the successful application of POMMs. Are there other POMMs, specifically, other partial orders on $\mathbf{Z} \times \mathbf{Z}$, that might provide "better" solutions to the inverse (classification or parameter) problem? How can one pick a "better" model? Could a nonlinear system describe a set of models, and use the data to optimize a system cost function to select the best-fitting POMM? The use of the co-occurence matrix of the data might be of some use in defining such a system. For example, information from the gray level co-occurence matrices could be used to determine the nature of spatial dependencies in a texture. This knowledge could then help choose a "better" POMM. Perhaps an artificial neural network could be designed to give a solution to the model selection problem, incorporating other texture features including the co-occurence matrix, first and second order statistics, etc.

Parameter estimation is another crucial factor in the problem of model-fitting. Currently, only parameter estimation of homogenous POMMs have been attempted; estimation and more modeling of heterogenous models require further investigation. Can visualization techniques be helpful in the parameter estimation and model selection problems? Even though the ML method yields estimates with desired properties, the estimates may not be the "best" for the POMM model with regard to a different objective function. A suggested alternative is to use a Bayesian parameter estimation technique by using information from the co-occurence matrix to fit a prior distribution on the possible parameter values. The concept of a loss function different from 0–1 may also be incorporated in the estimation technique that may provide better fits of the POMM model. A different optimization technique tried here could be used to obtain better and faster estimates.

Image restoration is another area to which POMMs could be applied. MRFs and a Bayesian approach have already been used for image restoration. Some types of POMMs could offer improved results. Unsupervised image segmentation may be carried out by simultaneous parameter estimation and image segmentation. An additional criterion to model noise may be included to reduce the effects of noise in the image. In many cases, highly correlated POMM model parameters may result in misclassification of data. A principal component analysis could be carried out to decorrelate the parameter estimates for more accurate classification.

Flinn's algorithm used for texture synthesis has a limitation due to the constraint of a fixed gray level distribution for the image throughout the synthesis process. Possible improvements are:

1. A technique that does not assign a fixed gray level distribution to the image;
2. One that attains swift convergence to an image satisfying the POMM distribution.

One method to speed up the synthesis process is to pick an initial image that is close to the desired distribution for the specified POMM parameters. This could be achieved by assigning the pixel values in the initial image by scanning the image in increasing order of the level sets of the POMM model. The gray value at each pixel could then be assigned based on the probability of each possible gray value at that site conditioned on the "previous" neighbors. This would give an initial image that is statistically "close" to the desired distribution.

It is known that Markov mesh models are a special kind of POMM, which in turn are a special kind of MRF. What are necessary and sufficient conditions for a MRF to be a POMM? Can graph theory be used to help solve this latter problem? Can a unilateral-type process like the POMM be used to approximate a bilateral process like MRFs? Recall that the use of a *dilation* in the mathematical morphology sense was necessary to derive Theorem 2, Section 4.2. A result relating mathematical morphology and a type of co-occurence matrix for texture analysis has been introduced [94]; what implications do these results have for POMMs? Finally, how can the POMMs be incorporated into Image Algebra? The structure of the lower adjacency sets, $adj_{\prec}a_{ij}$, is similar to that of a template, while the dilation $dil\ a_{ij}$ is related to template convolution under the operation of additive maximum. The additive maximum, $\boxed{\vee}$, and additive minimum, $\boxed{\wedge}$, operations are generalizations of the mathematical morphology operations of dilation and erosion, respectively. These are by no means the only open questions on this topic.

The POMM models developed here have potential applications to other nonimaging areas. Other spatial problems where there exist a particular "ordering" of r.v.s could be modeled by POMMs. A thorough analysis of the theoretical properties of POMMs would give a better idea of the possible wide ranging applications of POMMs.

REFERENCES

[1] B. Julesz, "Visual pattern discrimination," *IRE Trans. Inf. Theory*, vol. IT-8, pp. 84–92, Feb. 1962.

[2] B. Schacter, A. Rosenfeld, and L. Davis, "Random mosaic models for textures," *IEEE Trans. SMC*, vol. SMC-8, pp. 694–702, Sept. 1978.

[3] G. Cross and A. Jain, "Markov random field texture models," *IEEE Trans. on Patt. Analys. and Mach. Intell.*, vol. PAMI-5, pp. 25–39, Jan. 1983.

[4] J. Davidson and N. Cressie, "Markov pyramid models in image analysis," in *SPIE Symp. on Image Alg. and Morph. Image Proc. IV* (P. Gader, E. Dougherty, and J. Serra, eds.), vol. 2030, (San Diego, CA), pp. 179–190, Jul. 1993.

[5] K. Abend, T. Harley, and L. Kanal, "Classification of binary random patterns," *IEEE Trans. on Info. Theory*, vol. IT-11, pp. 538–544, Oct. 1965.

[6] R. Gonzalez and P. Wintz, *Digital image processing*. Reading, MA: Addison-Wesley, 1987.

[7] R. Duda and P. Hart, "Use of the Hough transformtion to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, pp. 11–15, 1972.

[8] A. Jain and S. Ranganath, "Image restoration and edge extraction based on 2-D stochastic models," *Proc. ICASSP-82*, pp. 1520–1523, May 1982.

[9] R. Kashyap and R. Chellapa, "Stochastic models for closed boundary analysis: representation and reconstruction," *IEEE Trans. Inf. Theory*, vol. IT-27, pp. 627–637, Sept. 1981.

[10] H. Samet, "Region representation: Quadtrees from boundary codes," *Commn. ACM*, vol. 23, pp. 163–170, Mar. 1980.

[11] A. Rosenfeld and J. Pfaltz, "Sequential operations in digital picture processing," *Journal of the Association for Computing Machinery*, vol. 13, pp. 471–494, 1966.

[12] J. Hawkins, *Textural properties for pattern recognition*, pp. 347–370. New York: Academic Press, 1970.

[13] A. Rosenfeld and B. Lipkin, *Texture synthesis*. New York: Academic Press, 1970.

[14] A. Rosenfeld and M. Thurston, "Edge and curve detection for visual scene analysis," *IEEE Trans. on Computers*, vol. C-20, pp. 562–569, 1971.

[15] R. Sutton and E. Hall, "Texture measures for automatic classification of pulmonary disease," *IEEE Trans. on Computers*, vol. C-21, pp. 667–676, 1972.

[16] G. Lendaris and G. Stanley, "Diffraction pattern sampling for automatic pattern recognition," *Proc. IEEE*, vol. 58, pp. 198–216, Feb. 1970.

[17] L. Kirvida, "Texture measurements for the automatic classification of imagery," *IEEE Trans. on Electromagnetic Compatibility*, vol. 18, pp. 38–42, 1976.

[18] R. Conners and C. Harlow, "A theoretical comparison of texture algorithms," *IEEE PAMI*, vol. PAMI-2, pp. 204–222, 1980.

[19] Haralick, M. Shanmugham, and I. Dinstein, "On some quickly computable features for textures," *Proc. 1972 symp. on Comp. Image Proc. and Recognition*, vol. 2, pp. 12.2.1–12.2.10, 1972.

[20] M. Nagao, H. Tanabe, and K. Ito, "Agricultural land use classification of aerial photographs by histogram similarity method," *Proc. 3rd Int. Joint Conf. Pattern Recognition*, pp. 669–672, 1976.

[21] Haralick, M. Shanmugham, and I. Dinstein, "Textural features for image classification," *IEEE Trans. SMC*, vol. 3, pp. 610–621, 1973.

[22] J. Tou and Y. Chang, "Picture understanding by machine via textural feature extraction," in *Proc. Third Int. Conf. on Pattern Recognition and Image Processing*, (Troy, NY), pp. 1–8, 1977.

[23] C. Dyer, T. Hong, and A. Rosenfeld, "Textural classification using gray level coccurence based on edge maxima," *IEEE Trans. SMC*, vol. 10, pp. 158–163, 1980.

[24] J. Bacus and E. Gose, "Leukocyte pattern recognition," *IEEE Trans. SMC*, vol. 2, pp. 513–526, 1972.

[25] M. Galloway, "Textural analysis using using gray run lengths," *Computer Graphics and Image Processing*, vol. 4, pp. 172–179, 1975.

[26] O. Faugers and W. Pratt, "Decorrelation methods of texture feature extraction," *IEEE Trans. PAMI*, vol. PAMI-2, no. 4, pp. 323–332, 1980.

[27] B. Ashjari, "Textural features for image classification," *IEEE Trans. SMC*, vol. 3, pp. 610–621, 1973.

[28] K. Laws, "Goal directed textured-image segmentation," *SPIE-Applications of Artificial Intelligence II*, vol. 548, pp. 19–26, 1985.

[29] M. Unser, "Local linear transforms for texture analysis," in *Proceedings of the Seventh International Conference on Pattern Recognition*, (Montreal), pp. 1206–1208, 1984.

[30] F. Ade, "Characterization of textures by eigenfilters," *Signal Processing*, vol. 5, pp. 451–458, Sept. 1983.

[31] B. Julesz and T. Caelli, "On the limits of Fourier decompositions in visual texture perception," *Perception*, vol. 8, pp. 99–73, 1979.

[32] W. Pratt, O. D. Faugeras, and A. Gagalowicz, "Visual discrimination of stochastic texture fields," *IEEE Trans. SMC*, vol. SMC-8, pp. 796–804, Nov. 1978.

[33] A. Gagalowicz, "A new method for texture field synthesis: Some applications to the study of human vision," *IEEE Trans. PAMI*, vol. PAMI-3, pp. 520–533, Sept. 1981.

[34] Mandelbrot, *Fractals-form, Chance and Dimension.* San Francisco, CA: Freeman, 1977.

[35] K. Deguchi and I. Morishita, "Texture characterization and texture-based partitioning using two-dimensional linear estimation," *IEEE Trans. on Computers*, vol. C-27, pp. 739–745, 1976.

[36] J. Read and S. Jayaramamurthy, "Automatic generation of texture feature detectors," *IEEE Trans. on Computers*, vol. C-21, pp. 803–811, 1972.

[37] R. Chellappa and R. Kashyap, "Texture synthesis using 2-d noncausal autoregressive models," *IEEE Trans. on Acous., Speech, and Sig. Proc.*, vol. ASSP-33, pp. 194–203, Feb. 1985.

[38] L. Xu and M. Azmi-Sadjadi, "Two-dimensional modeling of image random field using artificial neural networks," in *Proceedings ICASSP*, vol. 3, (Minneapolis), pp. I581–I584, 1993.

[39] R. Yokoyama and R. Haralick, "Texture pattern image generation by regular Markov chain," *Pattern Recognition*, vol. 11, pp. 225–234, 1979.

[40] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *J. R. Stat. Soc. B*, vol. 36, no. 1, pp. 192–236, 1974.

[41] J. Besag, "On the statistical analysis of dirty pictures," *J. R. Stat. Soc. B*, vol. 48, pp. 259–302, 1986.

[42] S. Geman and D. Geman, "Stochastic relaxation, Gibbs Distributions, and Bayesian restoration of images," *IEEE Trans. PAMI*, vol. PAMI-6, pp. 721–741, Nov. 1984.

[43] R. Hu and M. Moustafa, "Texture segmentation based on a hierarchical Markov random field model," *Sig. Proc.*, vol. 26, pp. 285–305, Mar. 1992.

[44] J. Woods, "Two-dimensional discrete Markov random fields," *IEEE Trans. Information Theory*, vol. IT-18, pp. 232–240, March 1972.

[45] R. Dobrushin, "The description of a random field by means of conditional probabilities and conditions of its regularity," *Theory Prob. Appl.*, vol. 13, pp. 197–224, 1968.

[46] E. Wong, "Recursive causal filtering for two-dimensional random fields," *IEEE Trans. Information Theory*, vol. IT-24, pp. 50–59, Jan. 1978.

[47] M. Hassner and J. Sklansky, "The use of Markov random fields as models of texture," *Comp. Vis., Graphics, and Image Proc.*, vol. 12, pp. 357–370, 1980.

[48] F. Spitzer, "Markov random fields and Gibbs ensembles," *Amer. Math. Mon.*, vol. 78, pp. 142–154, Feb. 1971.

[49] B. Prum and J. Fort, *Stochastic Processes on a lattice and Gibbs measure.* London: Kluwer Academic Press, 1991.

[50] R. Kindermann and J. Snell, *Markov random fields and their applications.* Providence, RI: American Mathematical Society, 1980.

[51] J. Hammersley and P. Clifford, "Markov fields on finite graphs and lattices," *(unpublished)*, 1971.

[52] P. Flinn, "Monte Carlo calculation of phase separation in a 2-dimensional Ising system," *J. Statist. Phys.*, vol. 10, pp. 89–97, 1974.

[53] F. Cohen and D. Cooper, "Real-time textured image segmentation based on non-causal Mar kov random field models," in *Proc. of SPIE Conf. Intel. Robot.*, vol. 449, (Cambridge, MA), pp. 17–28, Nov. 1983.

[54] P. Devijver, *Image Segmentation using Causal Markov Random Field Models*, pp. 131–143. Pattern Recognition: Lecture Notes in Computer Science, Berlin: Springer-Verlag, 1988.

[55] V. Lacroix, "Pixel labeling in a second-order Markov mesh," *Sig. Proc.*, vol. 12, pp. 59–82, 1987.

[56] W. Qian and D. Titterington, "Multidimensional Markov chain models for image textures," *J. R. Stat. Soc. B*, vol. 53, no. 3, pp. 661–674, 1991.

[57] J. Goutsias, "Unilateral approximation of Gibbs random field images," *Comp. Vis., Graphics, and Image Proc.: Graph. Models and Image Proc.*, vol. 53, pp. 240–257, 1991.

[58] J. Moura and N. Balram, "Recursive structure of noncausal Gauss-Markov random fields," *IEEE Trans. Info. Theory*, vol. 38, pp. 142–154, March 1992.

[59] D. Pickard, "Unilateral Markov fields," *Adv. Appl. Prob.*, vol. 12, pp. 655–671, 1980.

[60] J. Davidson and N. Cressie, "Partially ordered markov model and their use in texture analysis," *to be submitted to IEEE Trans. Info. Theo.*, 1994.

[61] N. Cressie and J. Davidson, "A theory of image analysis with partially ordered markov models," *in preparation.*

[62] G. Chartrand and L. Lesniak, *Graphs and Digraphs.* CA: Wadsworth, 1986.

[63] J. Blinn and M. Newell, "Texture and reflection in computer generated images," *Commun. Ass. Comput. Mach.*, vol. 19, pp. 542–560, 1976.

[64] W. Dungan, "A terrain and cloud computer generation model," *Comput. Graphics*, vol. 13, no. 2, pp. 143–150, 1979.

[65] L. Onural, "Generating connected textured fractal patterns using Markov random fields," *IEEE Trans. on Patt. Analys. and Mach. Intell.*, vol. 13, pp. 819–825, Aug. 1991.

[66] M. Hassner and J. Sklansky, "Markov random field models of digitized image texture," in *Proc. Int. Joint Conf. Pattern Recognition*, (Kyoto, Japan), pp. 522–540, Nov. 1978.

[67] A. Bortz, M. Kalos, J. Lebowitz, and M. Zendejas, "Time evolution of a quenched binary alloy: Computer simulation of a two-dimensional model system," *Phys. Rev. B*, vol. 10, pp. 535–541, 1974.

[68] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, pp. 1087–1091, 1953.

[69] J. Hammersley and D. Handscomb, *Monte Carlo methods.* London: Metheun and Co., 1964.

[70] C. Won and H. Derin, "Unsupervised segmentation of noisy and textured images using Markov random fields," *Comp. Vis., Graphics, and Image Proc.*, vol. 54, pp. 308–328, 1992.

[71] S. Geman and C. Graffigne, "Markov random field image models and their applications to computer vision," in *Proc. of the Int'l. Congress of Mathematicians*, vol. 2, (Berkeley, CA), pp. 1496–1517, 1986.

[72] L.Younes, "Estimation and annealing for Gibbsian fields," *Ann. Inst. Henri Poincare*, vol. 24, no. 2, pp. 269–294, 1988.

[73] C. Won and H. Derin, *On estimating the parameters of Gaussian Markov random fields and a class of Gibbs Distribution: Technical Report.* University of Massachusets, Amherst, MA, 1989.

[74] J. Besag, "Efficiency of pseudo likelihood estimation for simple Gaussian fields," *Biometrika*, vol. 64, no. 3, pp. 616–618, 1977.

[75] N. Cressie, *Statistics for Spatial Data.* New York: Wiley and Sons, 1991.

[76] P. Brodatz, *Textures.* New York: Dover, 1966.

[77] S. Kirkpatrick, C. Gellet, and M. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May 1983.

[78] E. Aarts and J. Korst, *Simulated annealing and Boltzmann machines.* J. Wiley, 1989.

[79] E. Aarts and P. Laarhoven, "Statistical cooling: A general approach to combinatorial optimi zation problems," *Phil. J. Res.*, vol. 40, pp. 193–226, 1985.

[80] P. Laarhoven and C. Boender, "A Bayesian approach to simulated annealing," *Prob. in the Eng. and Inf. Sc.*, vol. 3, pp. 453–475, 1989.

[81] R. Haralick and L. Shapiro, "Image segmentation techniques," *Comp. Vis. Graph. Image Proc.*, vol. 29, pp. 100–132, 1985.

[82] F. Tomita and S. Tsuji, *Computer analysis of visual textures.* Boston: Kluwer Academic Publishers, 1990.

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

[84] S. Lakshmanan and H. Derin, "Simultaneous parameter estimation and segmentation of Gibbs random fields using simulated annealing," *IEEE Trans. PAMI*, vol. 11, no. 8, pp. 799–813, 1989.

[85] R. Duda and P. Hart, *Pattern classification and scene analysiss.* New York: J. Wiley, 1973.

[86] R. Lippmann, "An introduction to computing with neural nets," *IEEE Mag. Acoust. Speech, Signal Proc.*, vol. 4, pp. 4–22, Nov. 1987.

[87] G. Ritter, D. Li, and J. Wilson, "Image algebra and its relationship to neural networks," *Proc. of SPIE Tech. Symp. Southeast on Optics, Elec-Opt. and Sensors*, pp. 90–101, March 1989.

[88] J. Davidson and F. Hummer, "Morphology neural nets: An introduction with applications," *IEEE Trans. Circuits Sys., Signal Proc.*, vol. 12, no. 2, pp. 177–210, 1993.

[89] J. Serra, *Image analysis and mathematical morphology.* London: Academic Press, 1982.

[90] S. Wilson, "Morphological networks," *Proc. of the 1989 SPIE Vis. Comm. and Image Proc.*, vol. 4, pp. 483–494, Nov. 1989.

[91] J. Davidson, "Simulated annealing and morphological neural nets," *SPIE - Image algebra and Morph. Image Proc. 3*, vol. 1769, pp. 119–127, 1992.

[92] J. Goutsias, "Mutually compatible Gibbs random fields," *IEEE Trans. on Info. Th.*, vol. 35, pp. 1233–1249, Nov. 1989.

[93] W. Qian and D. Titterington, "On the use of Gibbs Markov chain models in the analysis of images based on second-order pairwise interactive distributions," *Journal of Applied Stat.*, vol. 16, pp. 267–281, 1989.

[94] A. Loui, A. Venetsanopoulos, and K. Smith, "Morphological autocorrelation transform: A new representation and classification scheme for two-dimensional images," *IEEE Trans. on Image Processing*, vol. 1, pp. 337–354, May 1992.

ACKNOWLEDGMENTS

The work in this thesis owes its genesis and success to the support of many colleagues and friends. In particular, I would like to thank my thesis advisor, Dr. Jennifer Davidson, for her ideas, encouragement, and guidance. I also thank Dr. Noel Cressie for his comments and for providing me with many interesting ideas, insights and suggestions. Special thanks to Rashmi, my friend and research mate, whose friendship and support made my stay in Ames an unforgettable one.